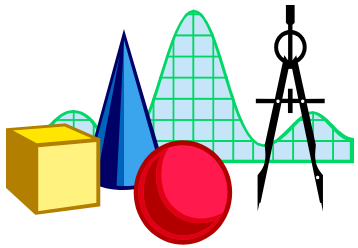
Brief Introduction to Object Technology

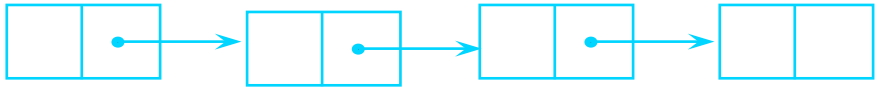


M. Moussavi, 2019

What is an Object?

- From human cognition perspective, an object is any of the following:
  - A tangible and/or visible thing (a physical entity)
  - Something that may be apprehended intellectually (a conceptual entity)
  - Something toward which a thought or action is directed.

Chemical Process



Linked List

Characteristics of objects

- Every object has the following three characteristics: a state, a behavior, and an identity.
  - Object = State + behavior + identity
- Considering a person as an object:
  - Person's identity
  - Persons behavior(s)
  - Persons state

Another Formal Definition

- An object is an entity with a well-defined boundary and identity that encapsulates *state* and *behavior.*
- An object has two key components: attributes and operations.
  - Attributes represent an object's state
  - Operations represent the behavior of the object.

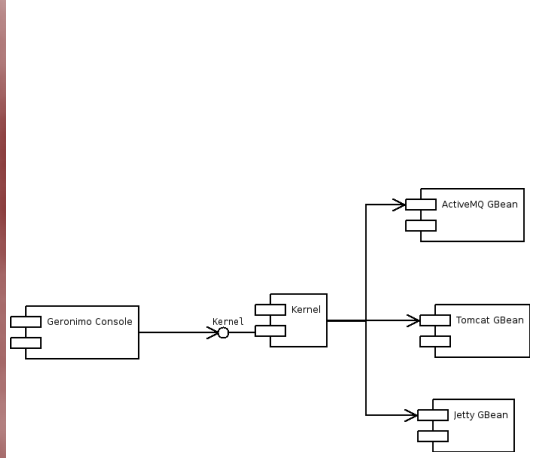
Introduction to Object Modelling

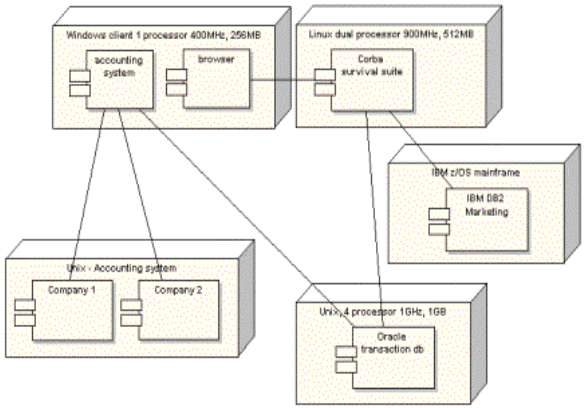
What is UML?

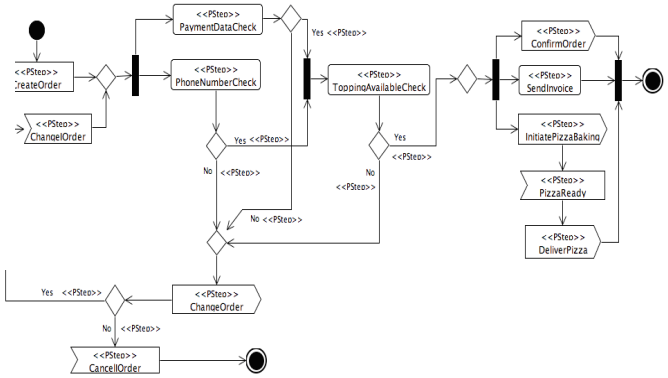
- UML, the Unified Modeling Language is a fusion of the notations of Booch, OMT, OOSE and others.
- UML is a modeling language not a method.
- The modeling language is the notations that methods use to express designs.
- The modeling language is the key part of communication.
- Permits forward engineering (i.e., generation of code from a UML model to a programming language)
- Permits reverse engineering (i.e., reconstruction of a model from an implementation back into the UML.

Choice of Models

- No one type of model can show everything about a system. Eg:
- Several models are used to give different views of the same system.
- Use any model or combination of models that works for the current situation.
- Most systems require multiple models.







UML 2.5 Models

- Behavioral Models
  - Use case diagram
    - Interaction with environment and external world
  - Interaction diagrams
    - Sequence diagram
    - Collaboration diagram
    - Interaction overview diagram and special type of interaction diagram.
    - Timing diagram
    - More…
  - State transition diagrams
  - Activity diagrams
- Static Models
  - Class diagram
  - Object diagram (an instance of class diagram)
  - Component diagram
  - Deployment diagram

Basic Diagrams

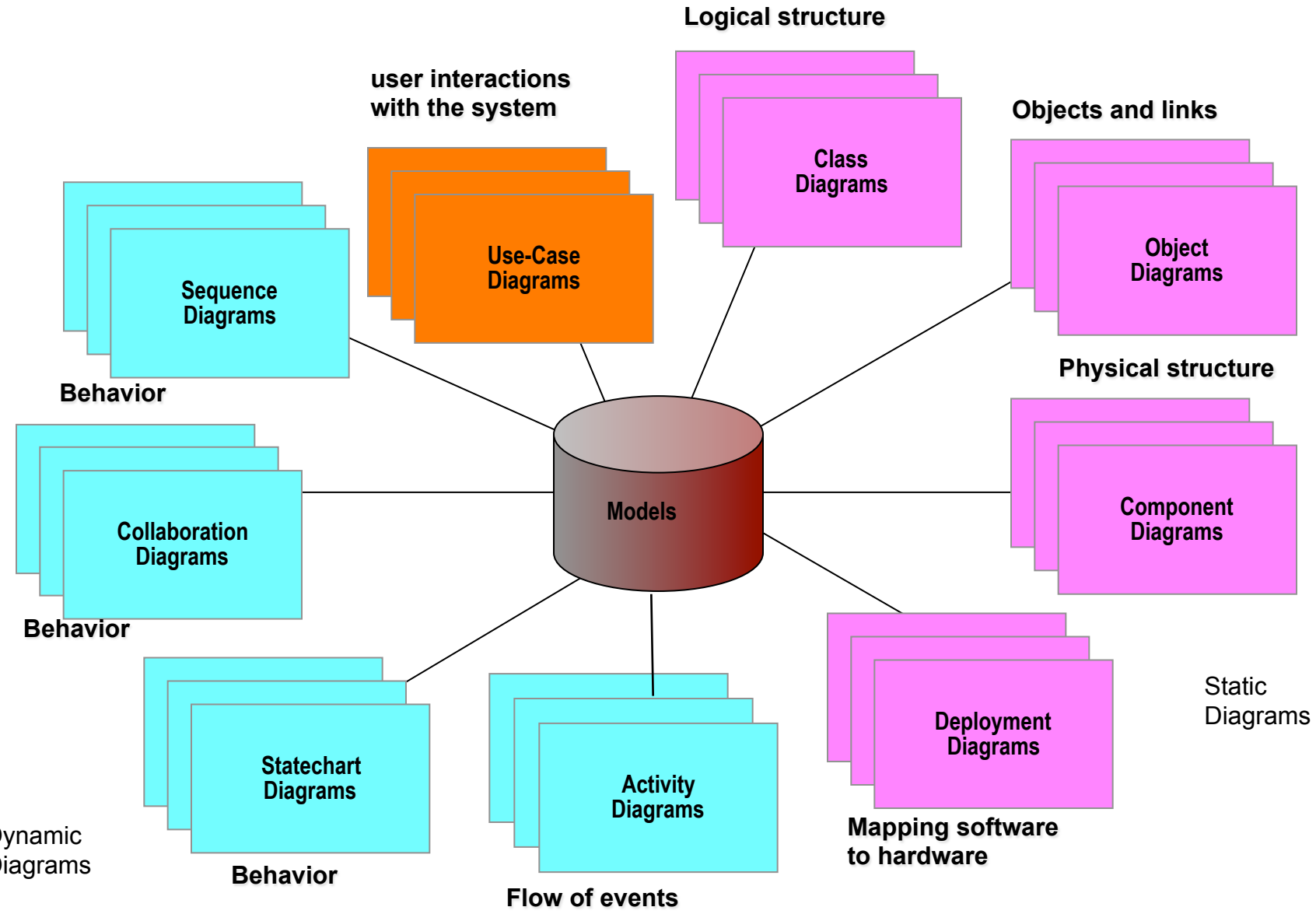


Figure From Rational Rose

UML Notation for Classes and Objects

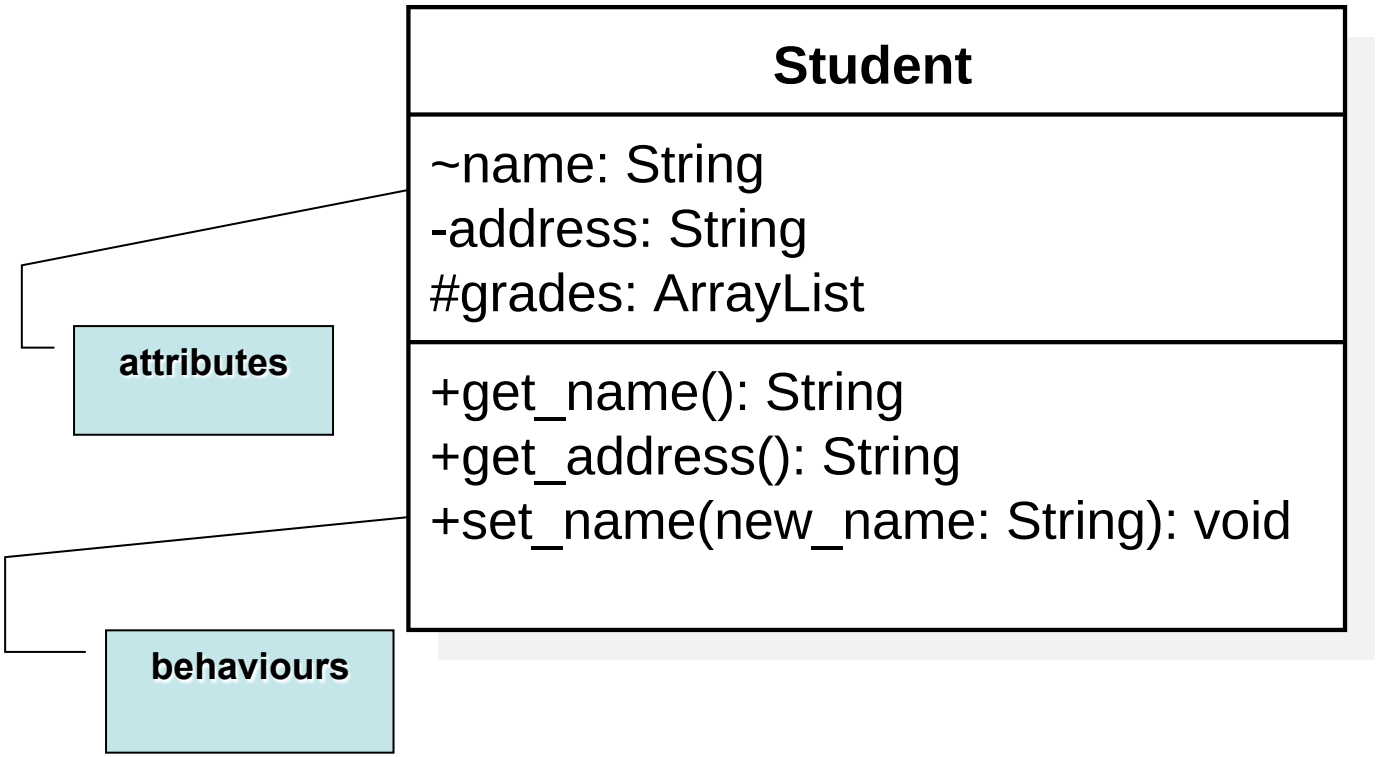
What Is a Class?

- A class is a description of a set of objects that share the same *attributes*, *operations*, *relationships*, and semantics.
  - An object is an instance of a class.
- Recognizing the commonalties among the objects and defining classes helps us deal with the potential complexity.

Classes and Object the UML

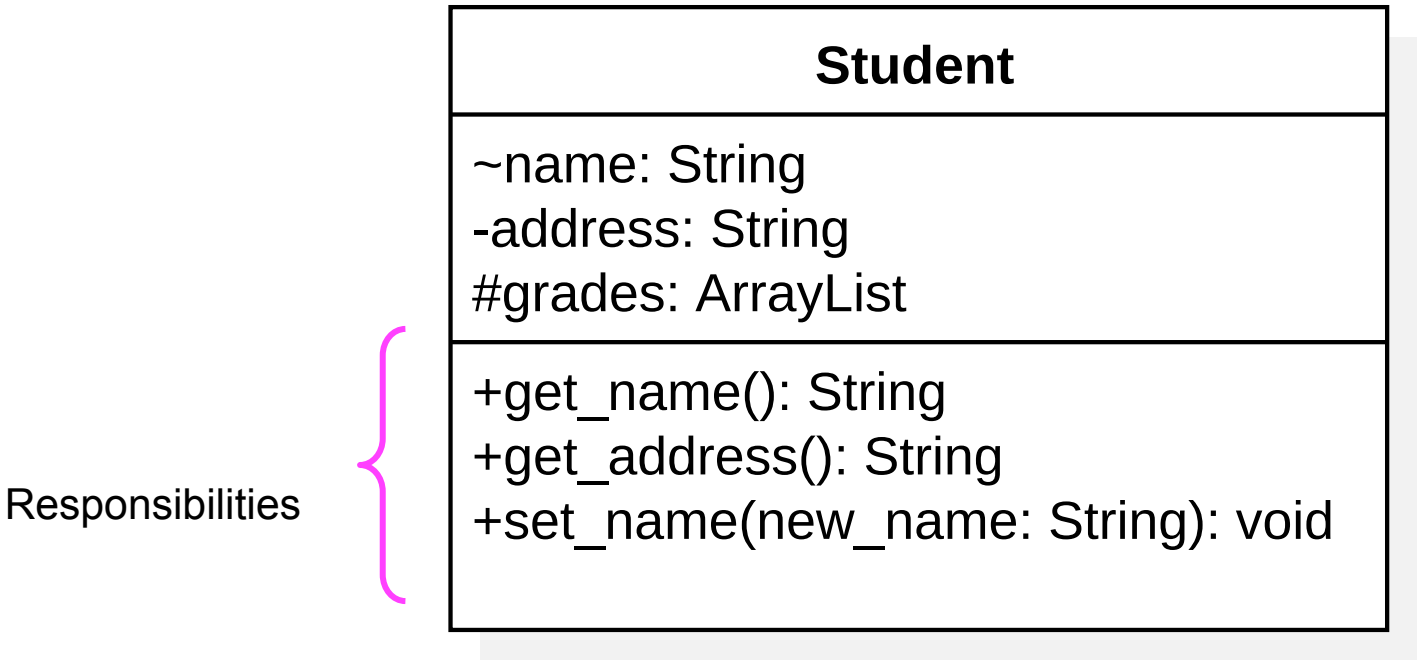
- A class is represented using a rectangle with compartments.

Class Notation in UML



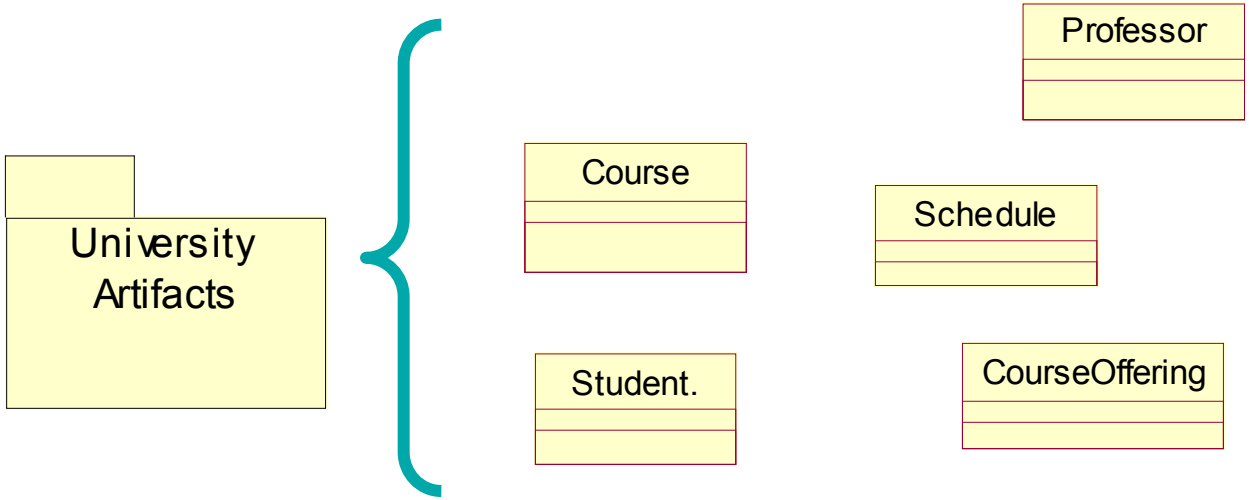
What Are Class Responsibilities?

- A responsibility is a contract or obligation of a class.
- It is a statement of something the class can be expected to provide.



Representing Packages in UML

- A package is a general purpose mechanism for organizing elements into groups.
- It is a model element that can contain other model elements.

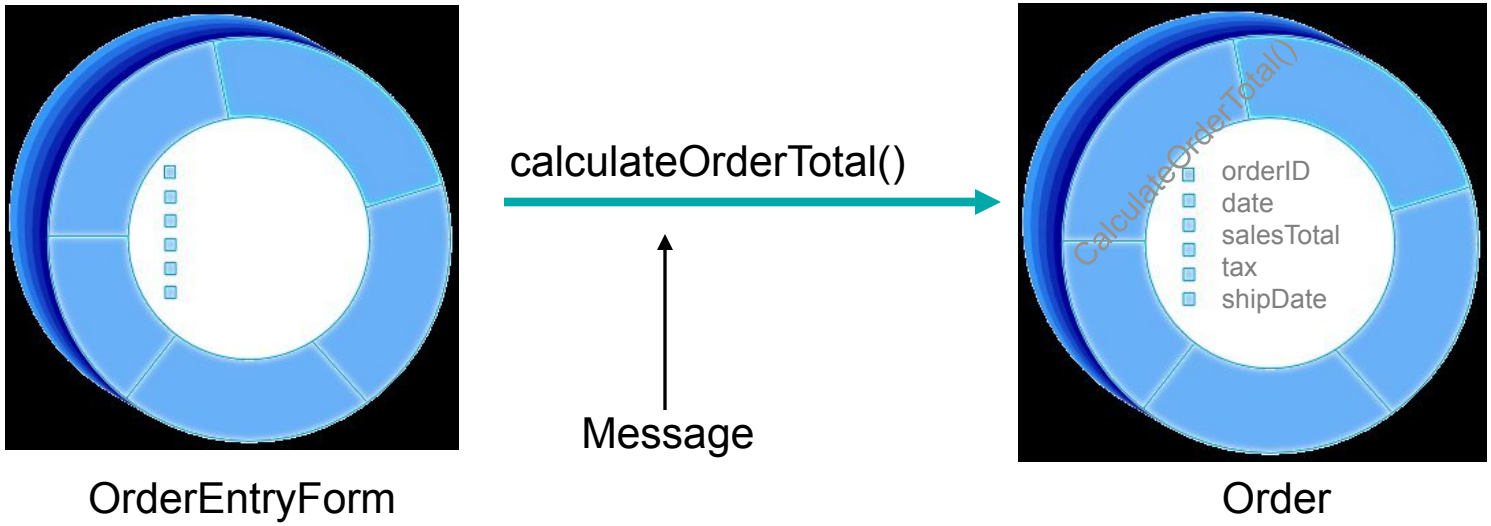


A package owns its elements and can even own other packages.
If the package is destroyed, the element is destroyed, too.

Object Collaborations and Responsibilities

How to Discover Responsibilities Object Interaction

- The OrderEntryForm wants Order to calculate the total dollar value for the order.

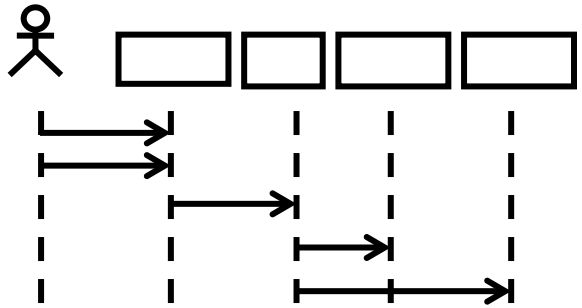


The class Order has the *responsibility* to calculate the total dollar value.

Figure From Rational Rose

What Is a Sequence Diagram?

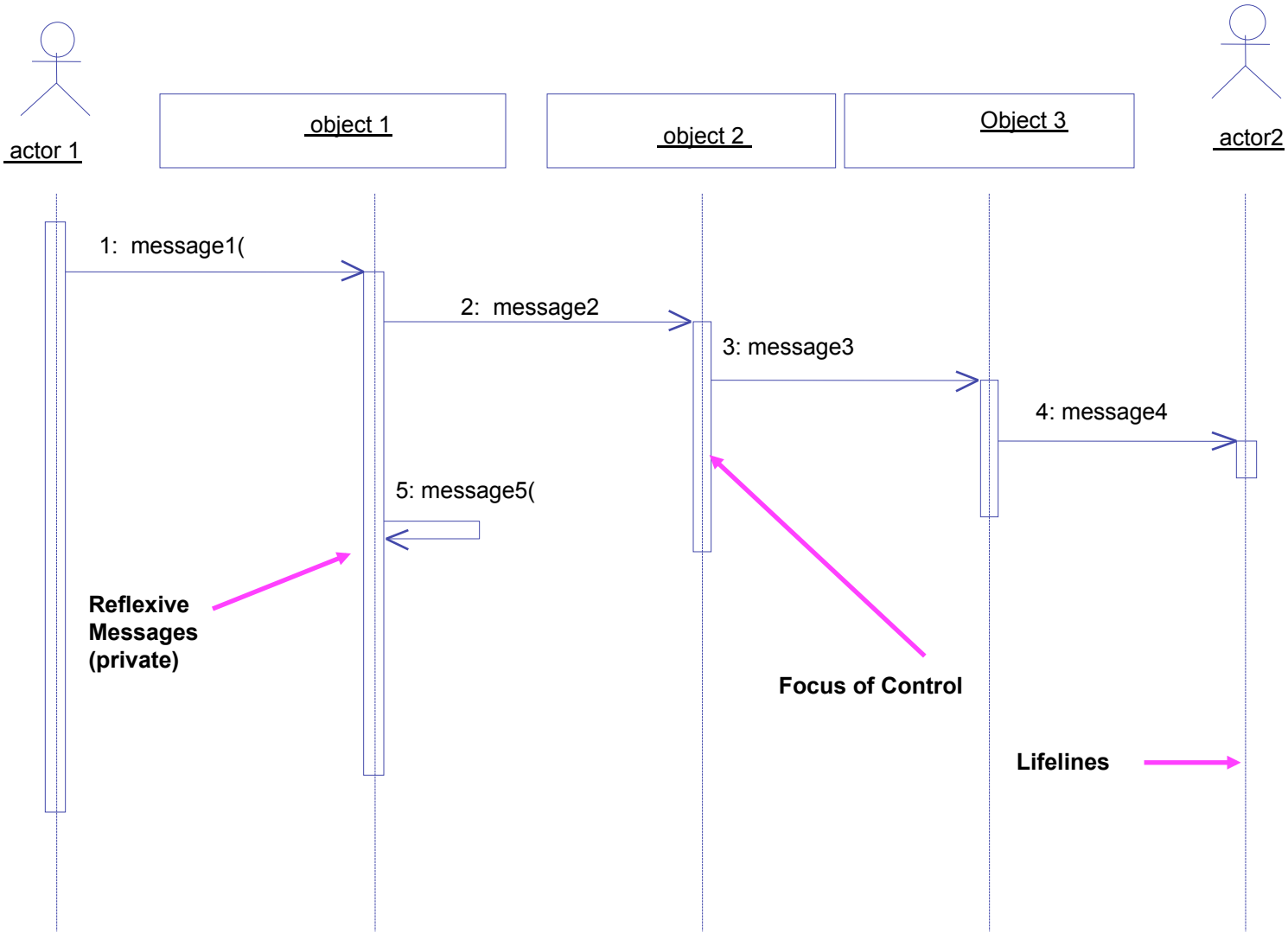
- It models the dynamic aspects of a system.
- is an interaction diagram that emphasizes the time ordering of messages.
- The diagram shows
  - The objects participating in the interaction.
  - The sequence of messages exchanged.



Sequence Diagrams

Figure From Rational Rose

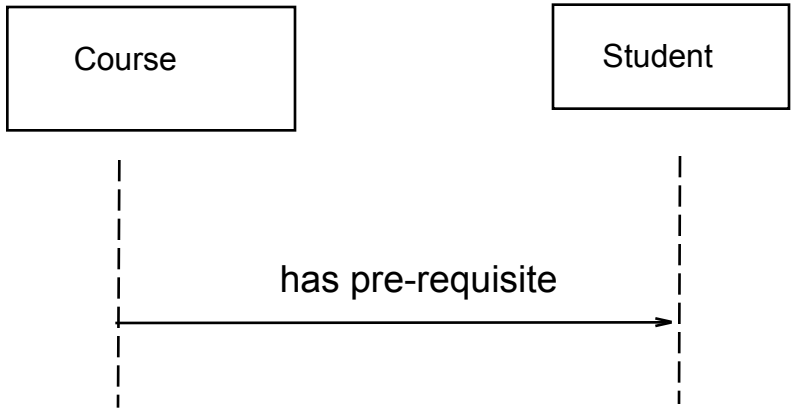
Example: Sequence Diagram

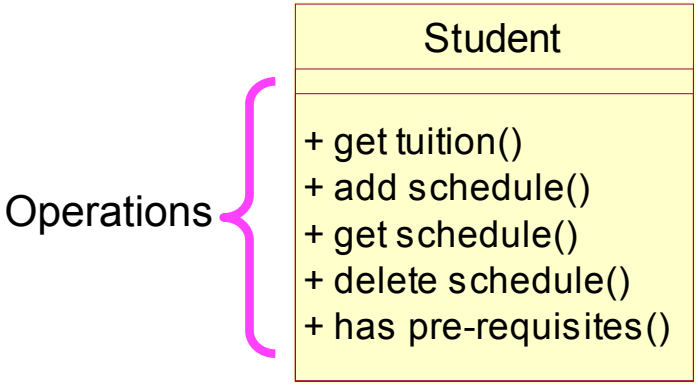


The **focus of control**: a thin rectangle that shows the period of time during which an object is performing something

Extracting Information from Sequence Diagrams

- An operation is the implementation of a service that can be requested from any object of the class to affect behavior.
  - Messages displayed in interaction diagrams

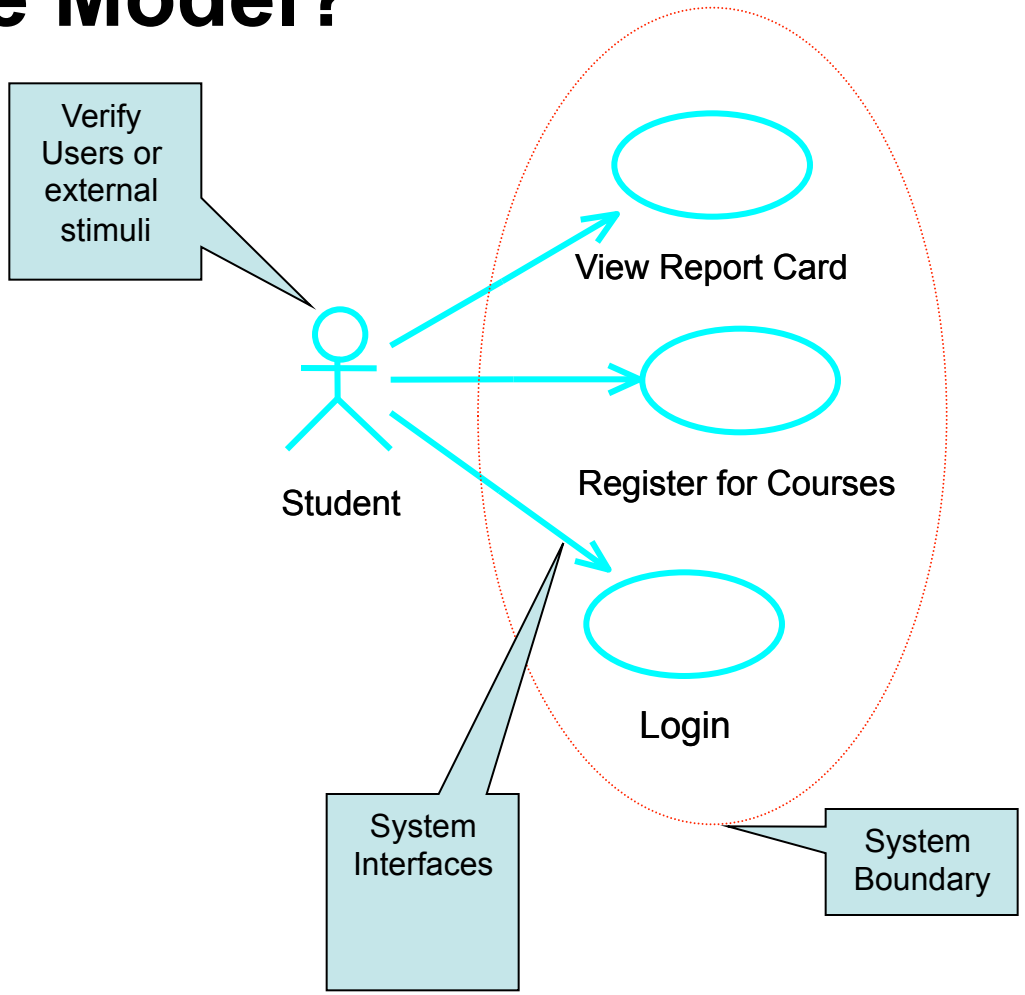




Use-Case Model

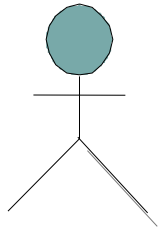
What Is a Use-Case Model?

- 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)
- Used to communicate with the end users and domain experts

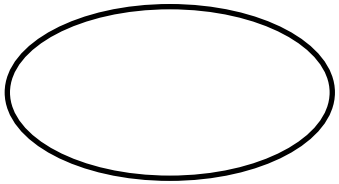


Major Concepts in Use-Case Modeling

- An actor represents anything that interacts with the system.
- A use case is a sequence of actions a system performs that yields an observable result of value to a particular actor.
  - A use case describes *what* a system does, but it does not specify *how* it does.



Actor



Use Case

Focus on the Roles

- An actor represents a role that a human, hardware device, or another system can play.
- The difference between an actor and an individual system user is that an actor represents a particular class of user rather than an actual user.
- Several users can play the same role.

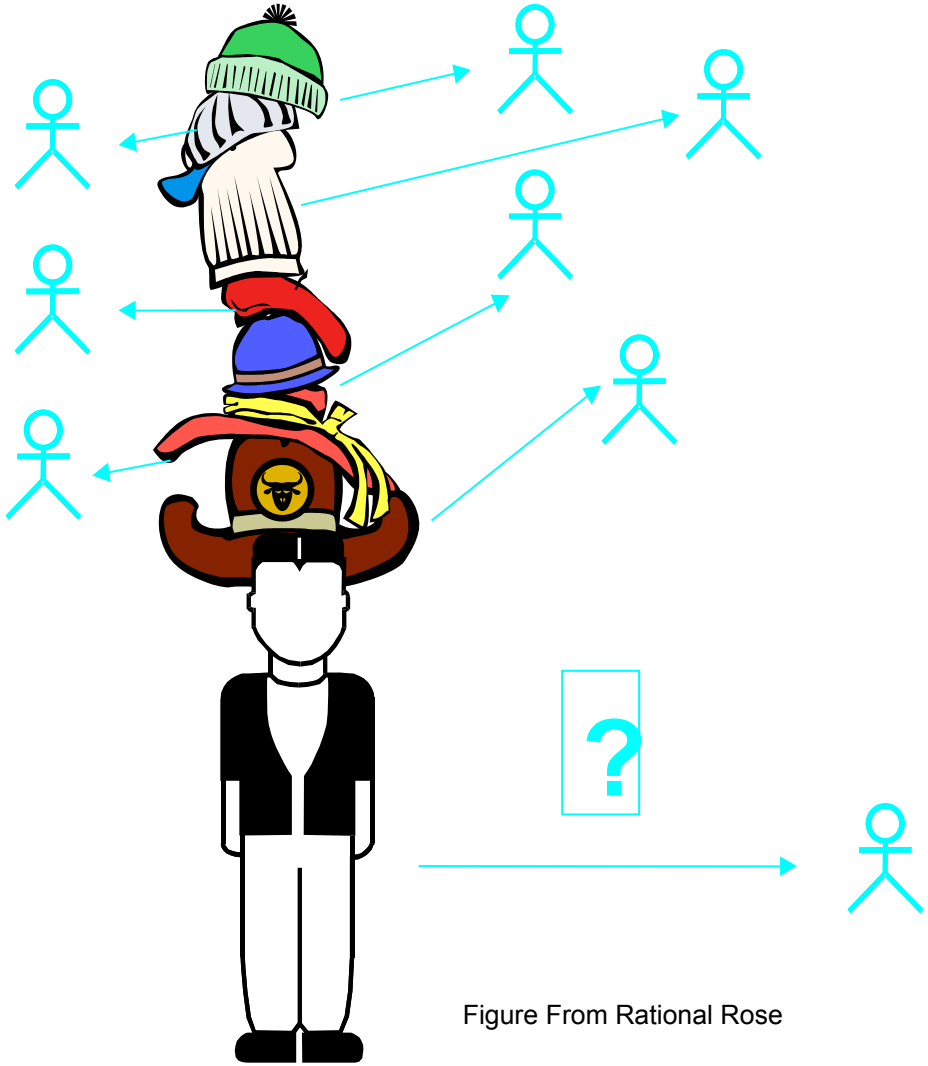
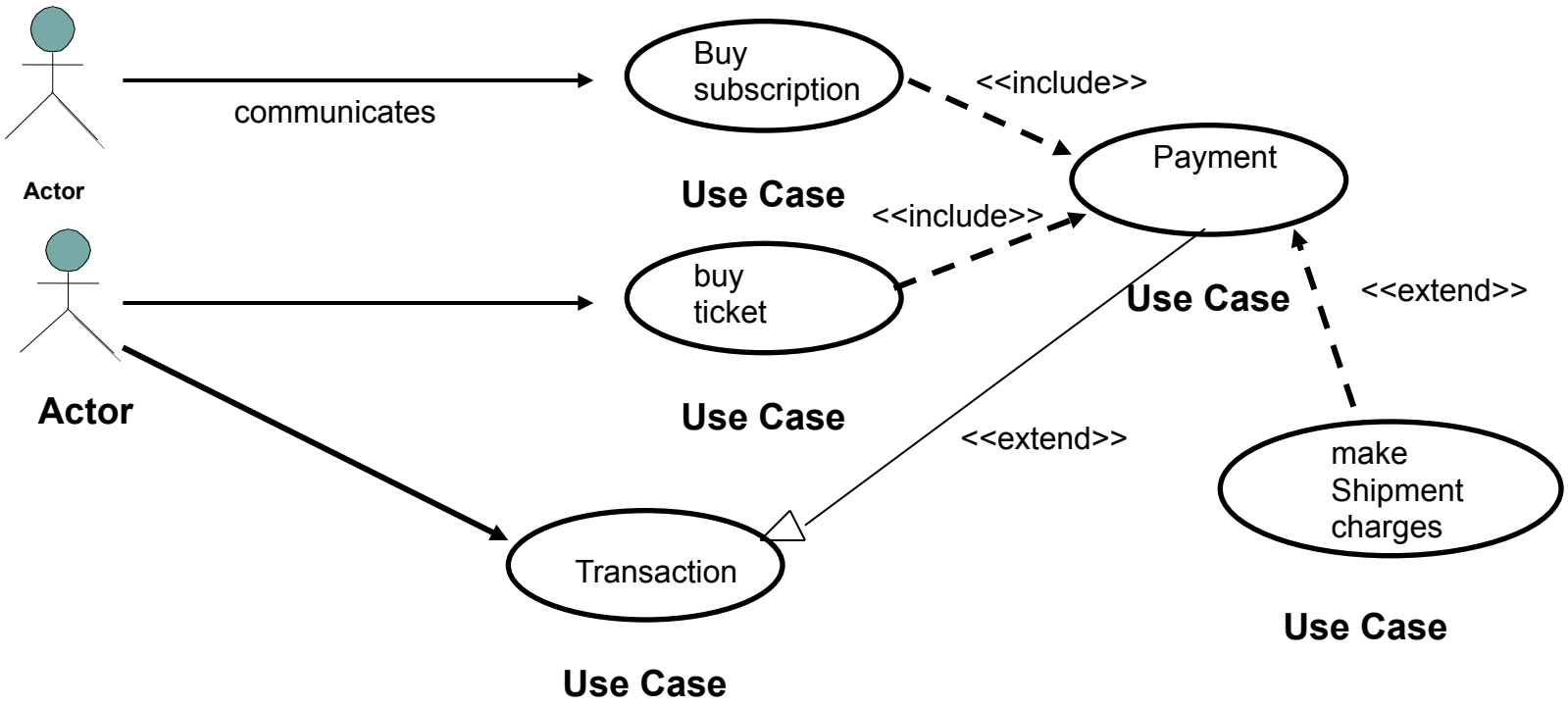


Figure From Rational Rose

Use case diagram

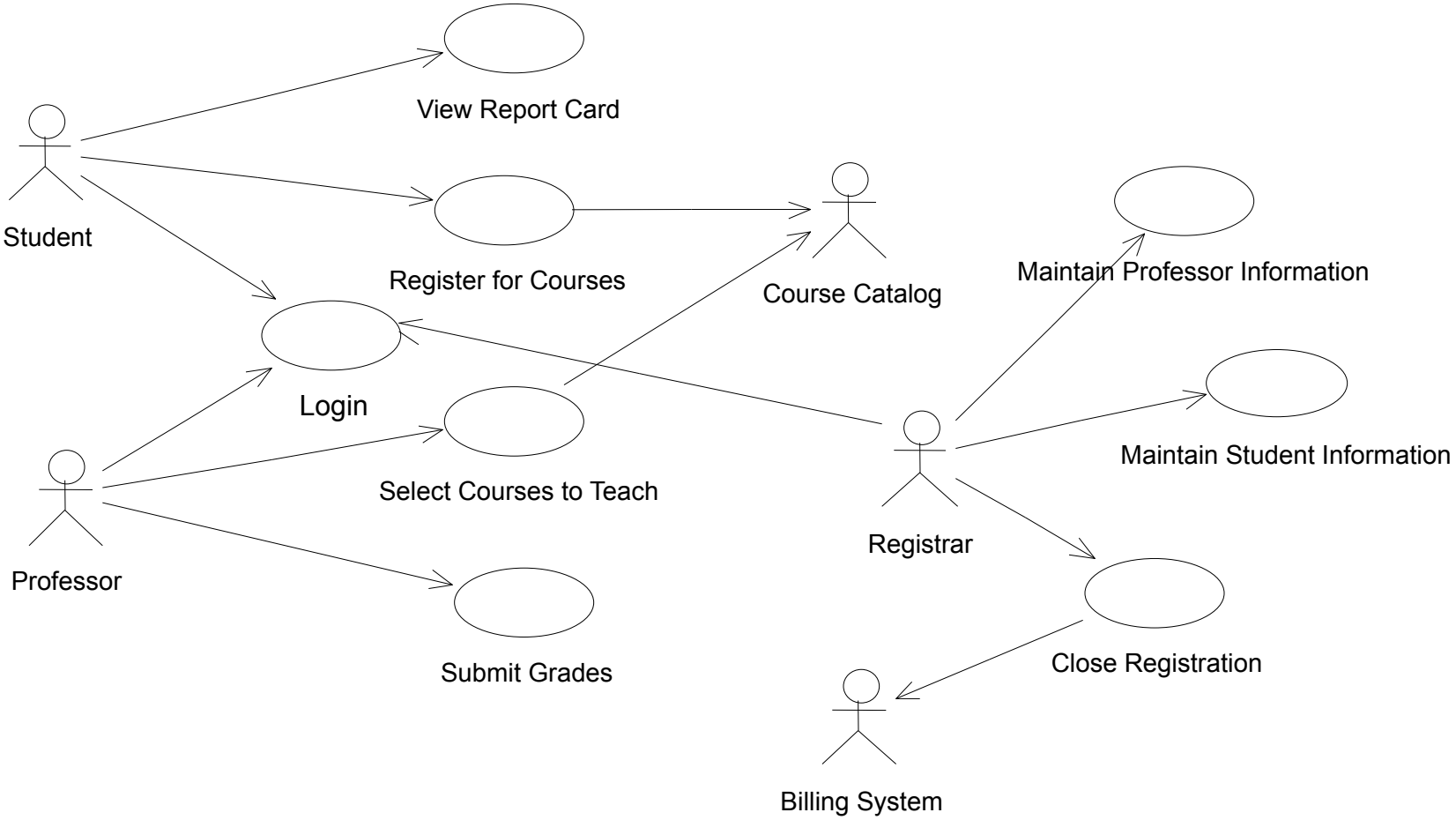
- A use case diagram is a dialog between actors and the system.
- Remember a use case diagram is not a flow chart



Discussion:

- Identify some possible actors and use cases in the following systems:
  - A greenhouse to grow vegetables
  - An automated teller machine

How Would You Read This Diagram?



Answer the following questions:

Describe the functionality of this system.

Describe the actor relationships for the Close Registration and Select Courses To Teach use cases.

What doesn't this model say?

Use-Case Model/Specifications

- Name
- Brief description
- Flows of events
- Relationships
- Activity and state diagrams
- Use-Case diagrams
- Special requirements
- Pre-conditions
- Post-conditions
- Other diagrams

Use-Case Flow of Events

- Has one normal, *basic flow* ("Happy Path")
- Several *alternative flows*
  - Regular variants
  - Odd cases
  - **Exceptional flows** handling error situations
- A scenario is an instance of a use case. It is one flow through a use case.

Finding Classes

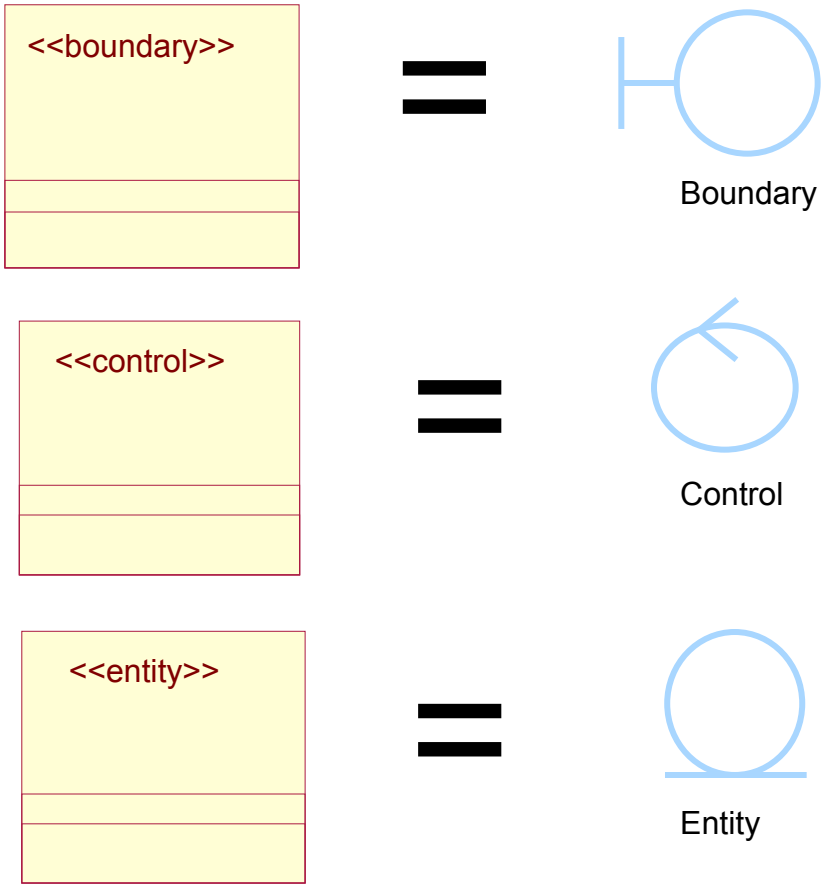
Where Do We Find Classes?

- Use-case realization documents
  - Flow of events (scenarios)
  - Interaction diagrams
  - Sequence diagrams
- Business documents
  - Forms
  - Receipts
  - etc
- Stakeholder documents
- Any project documentation

Guidelines for Class Discovery

- Classes should reflect the business.
- Classes should have descriptive names.
- Classes should have a clear description.
- Classes should have a set of related responsibilities.

What Are Analysis Classes?



What Is a Boundary Class?

- Models the interaction between the system's surroundings and its inner workings
  - User-interface classes
  - Device-interface classes
  - System-interface classes
- Environment dependent
  - GUI dependent
  - Communication protocol dependent

The Role of a Boundary Class

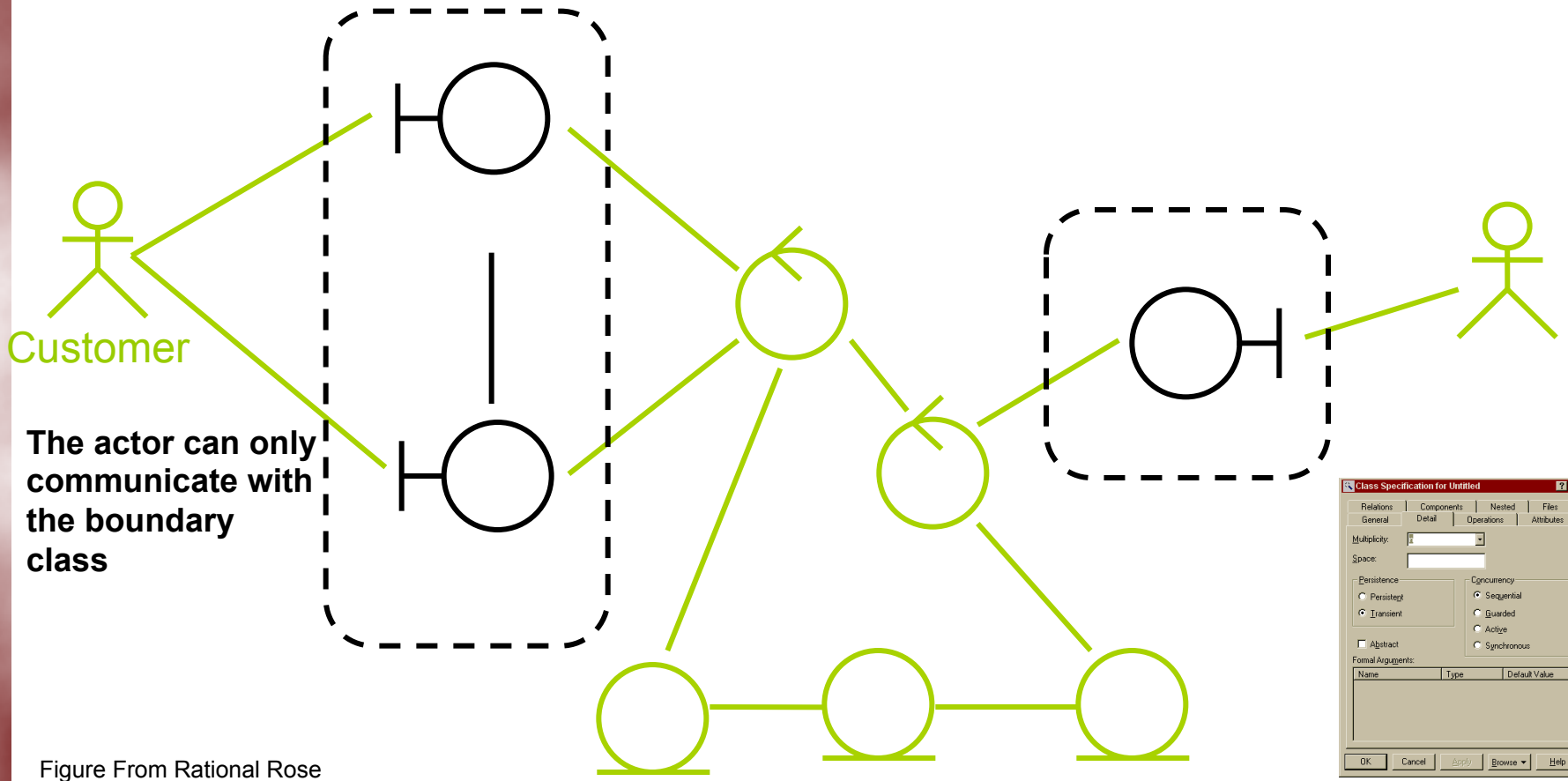


Figure From Rational Rose

Model interaction between the system and its environment.

Finding Boundary Classes

- There should be at least one boundary object for each actor/use-case pair.

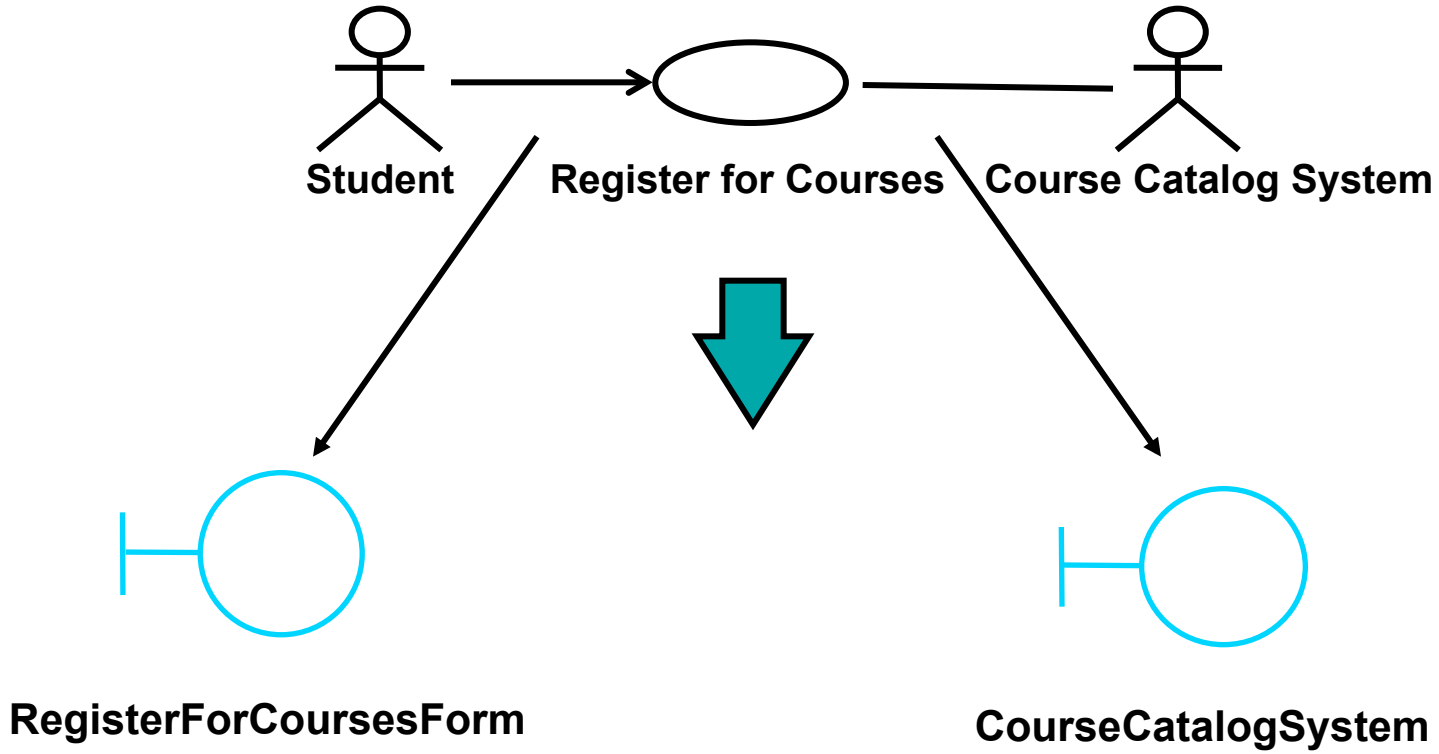


Figure From Rational Rose

What Is a Control Class?

- Controls the behavior of a use case
- Delegates the work of the use case to other classes
- Use case dependent, environment independent

Control

A control class is dependent on the use case to which it belongs. If the use case changes, its flow of events changes. The control class will likely need to change, too.

Unlike a boundary class, a control class is environment independent. The same control class can be used to coordinate many different boundary classes. When the system performs the use case, a control object is created. Control objects usually die when their corresponding use case has been performed.

What Is the Role of a Control Class?

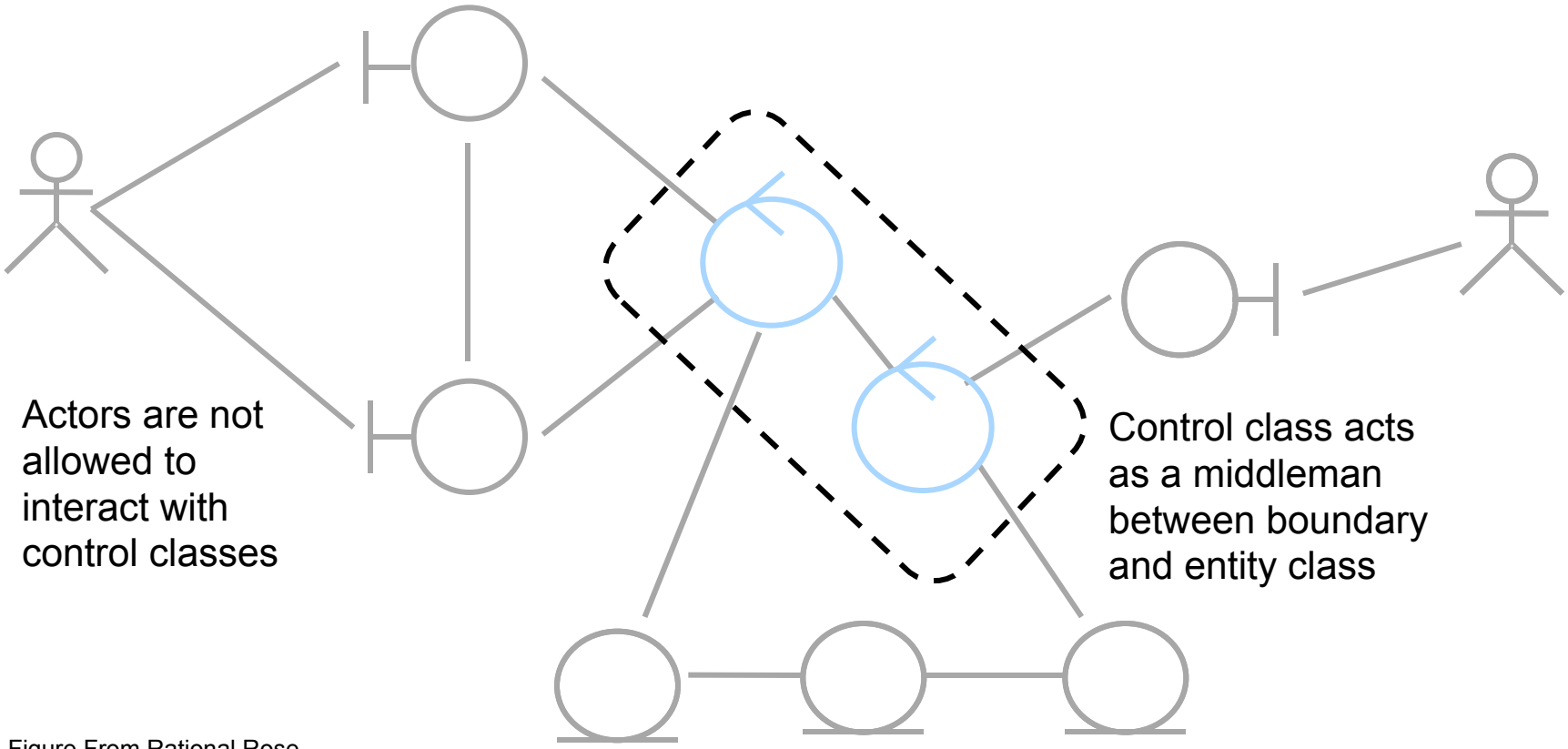


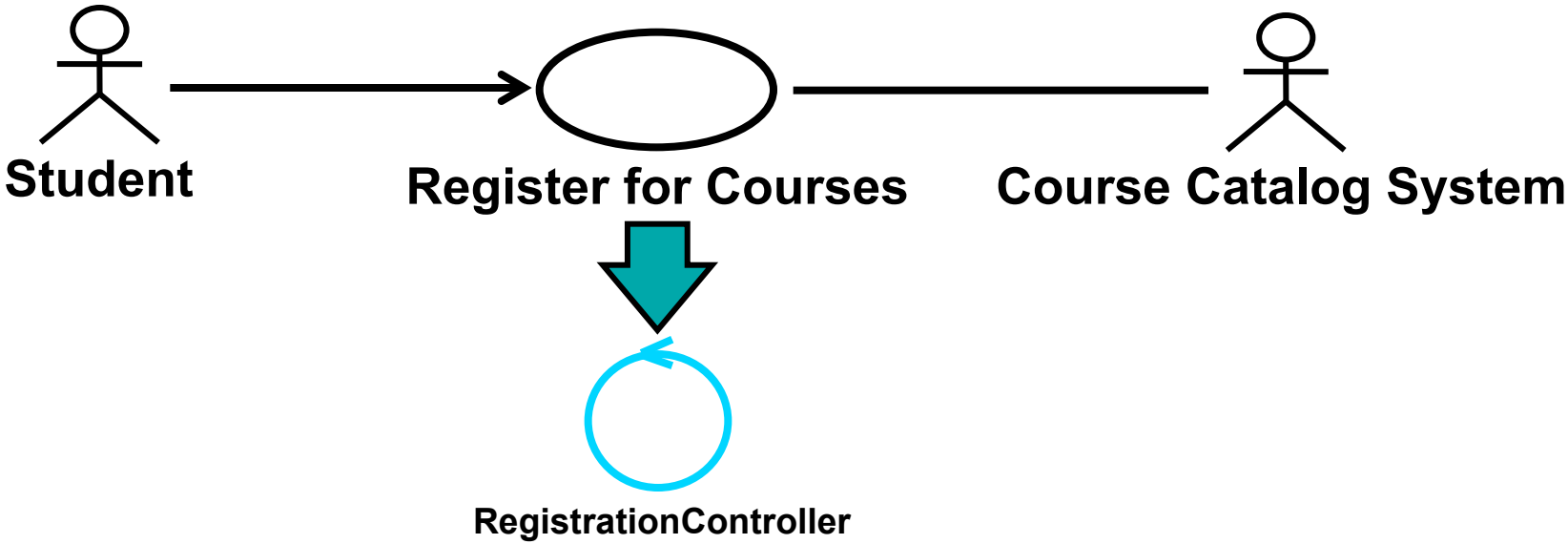
Figure From Rational Rose

Coordinate the use-case behavior.

Hints for Using a Control Class

- Control classes should only do sequencing.
- Be “bossy”: A control class should tell other classes to “do something” and should never “do anything” except for directing.
- A control class should only say “what to do” and leave the “how to do” to the other classes. (poor design may mix these two)
- Use control classes to decouple boundary and entity classes.

Finding Control Classes



- There might be one control class per use case
  - For Smaller use cases you may have a control class for several use cases
  - For larger use cases you may want to have two or more control classes

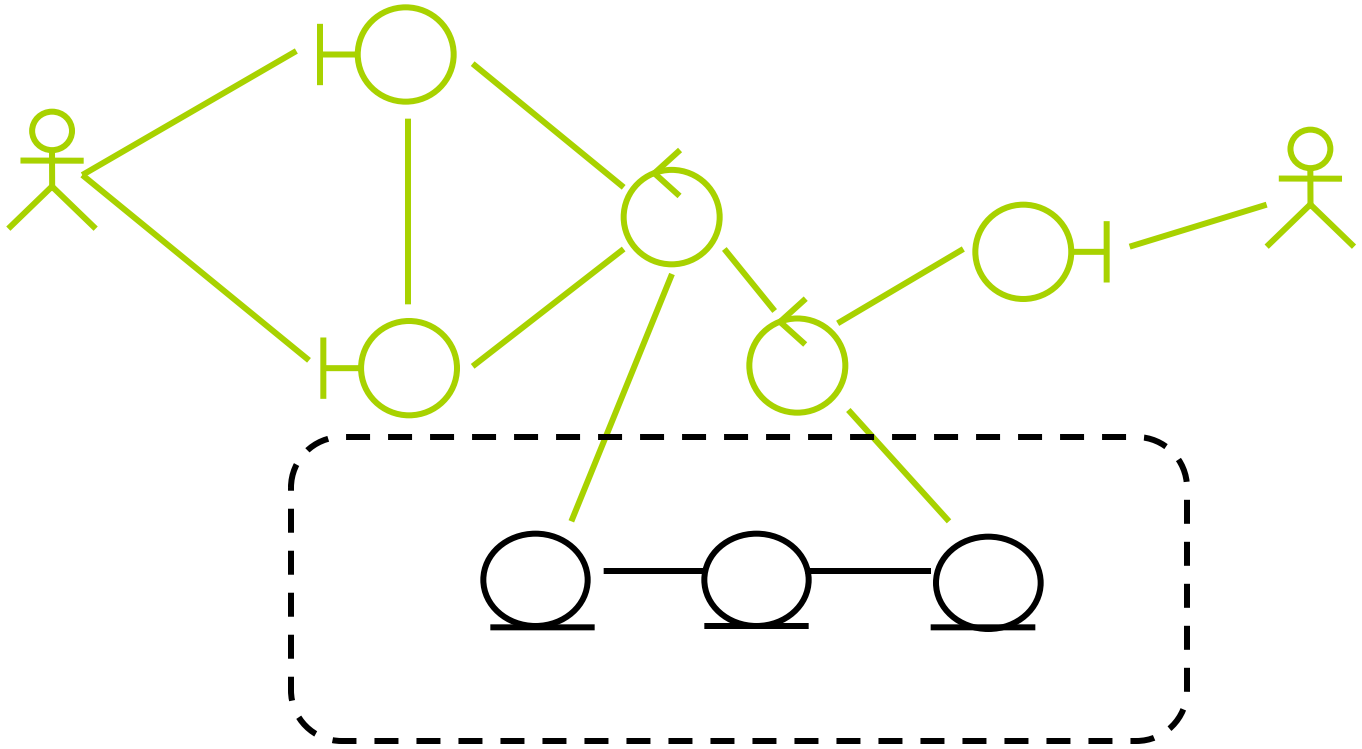
Figure From Rational Rose

What Is an Entity Class?

Entity

- Models the key concepts of the system
- Usually models information that is long lived (persistent)
- Environment independent
- Can be used in multiple use cases

The Role of an Entity Class



Store and manage information in the system. e.g., Account and Customer in a banking system.

Entity objects identified by examining the nouns and noun phrases in use cases

Figure From Rational Rose

A Sample Use-Case Scenario

The "Create A Schedule Scenario"

John enters the Student ID number 369-523449 and the system validates the number. The system asks which semester. John indicates the current semester and chooses create a new schedule.
From a list of available courses, John selects the primary courses: English 101, Geology 110, World History 200, and College Algebra 110. He then selects the alternate courses: Music Theory 110 and Introduction to Java Programming 180.
The system determines that John has all the necessary prerequisites by examining the student record and adds him to the course rosters. The system indicates that the activity is complete. The system prints the student schedule and sends billing information for four courses to the billing system for processing.

- Filter the nouns from this description.

Figure From Rational Rose

Filtering Decisions (only some of the noun)

Noun	Filtering decision
John	filtered (actor)
Student ID number	filtered (property of a student)
System	filtered (what is being built)
Number	filtered (same as student ID num)
Semester	filtered (state – when the selects apply)
Current Semester	filtered (same as semester)
New schedule	candidate object
List of available courses	candidate object
Primary courses	filtered (state of a selected course)
English 101	candidate object
Geology 110	candidate object
World History 200	candidate object
College Algebra 110	candidate object

Figure From Rational Rose

Example: Entity Classes

- Catalogue: List of all courses being taught in a semester
- Schedule: A list of courses for a semester for a student
- StudentRecord: List of previously taken courses
- Course: An offering for a semester
- BillingInformation: Information needed by the billing system actor
- CourseRoster: List of students for a specific course offering

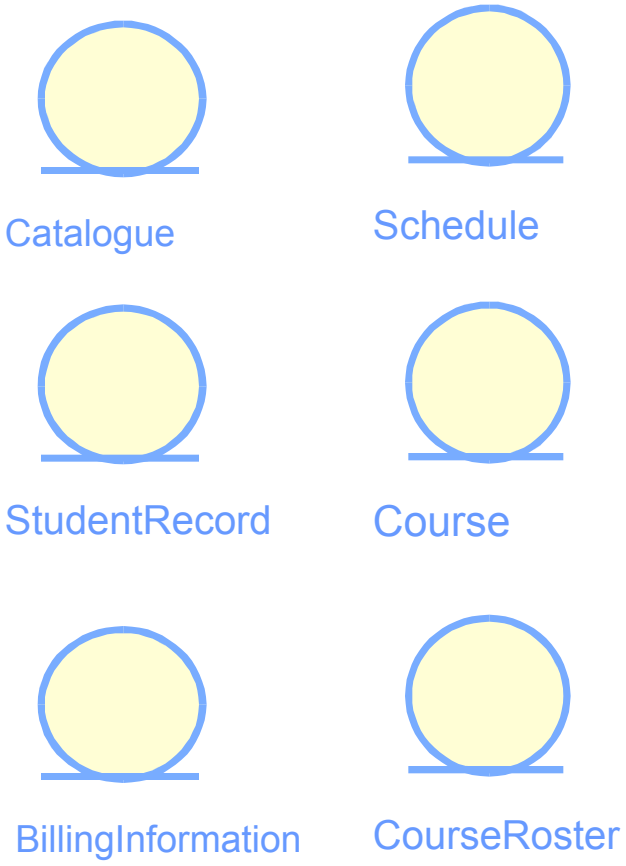


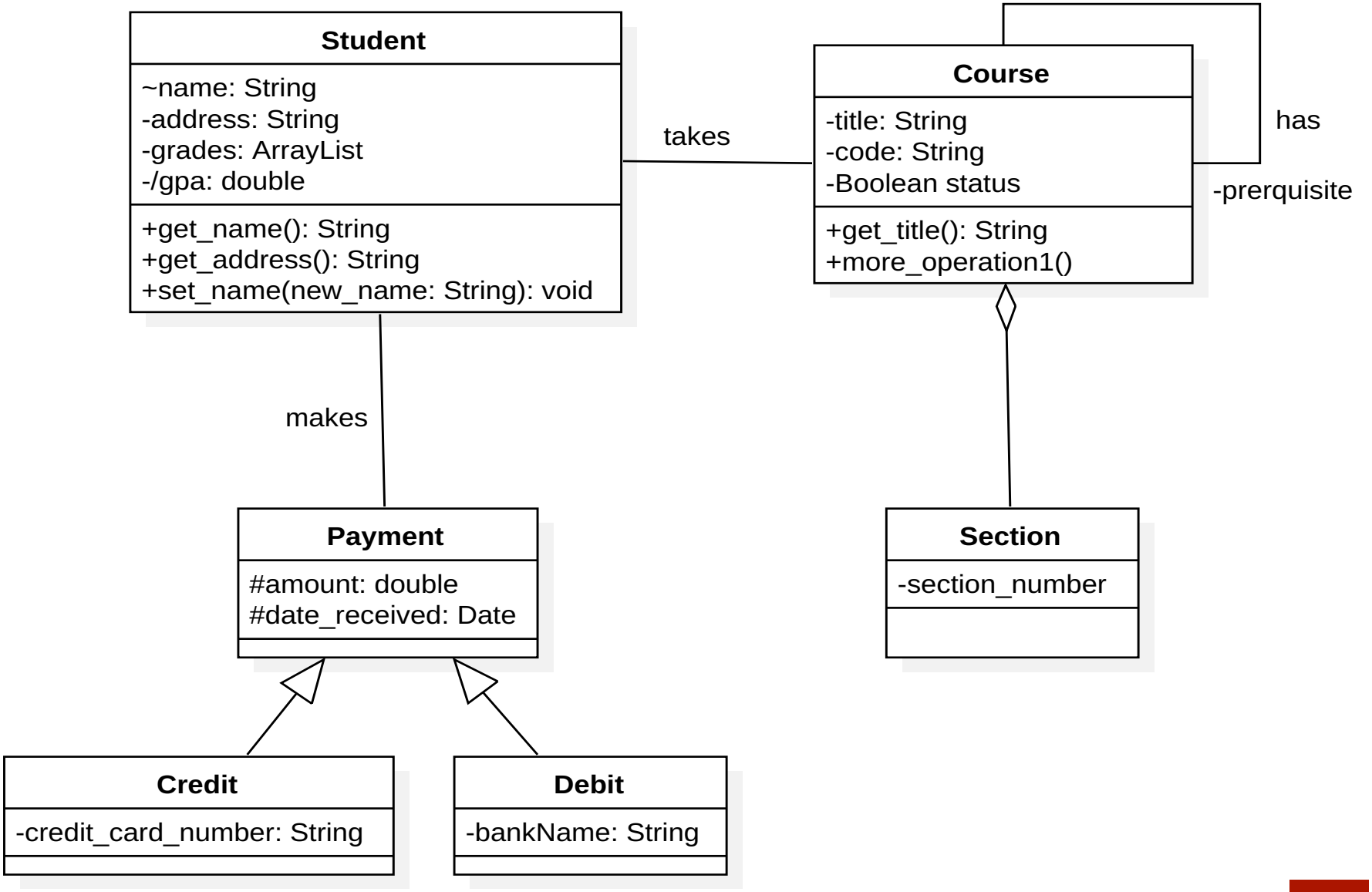
Figure From Rational Rose

Closer Look at the Class Diagrams (Relationship among Classes)

What is Class Diagram

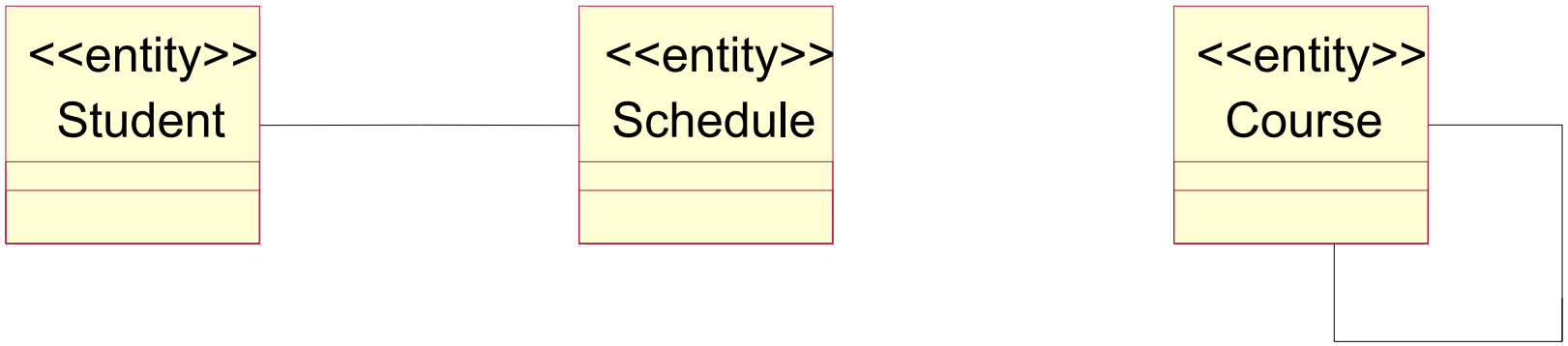
- A Static view of a system that shows the collection of classes involved to build a system or sub-system.
- The following simple class diagram shows several types of associations among several classes

What Is a Class Diagram



Class Relationship (Simple 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



How Do I Find Association?

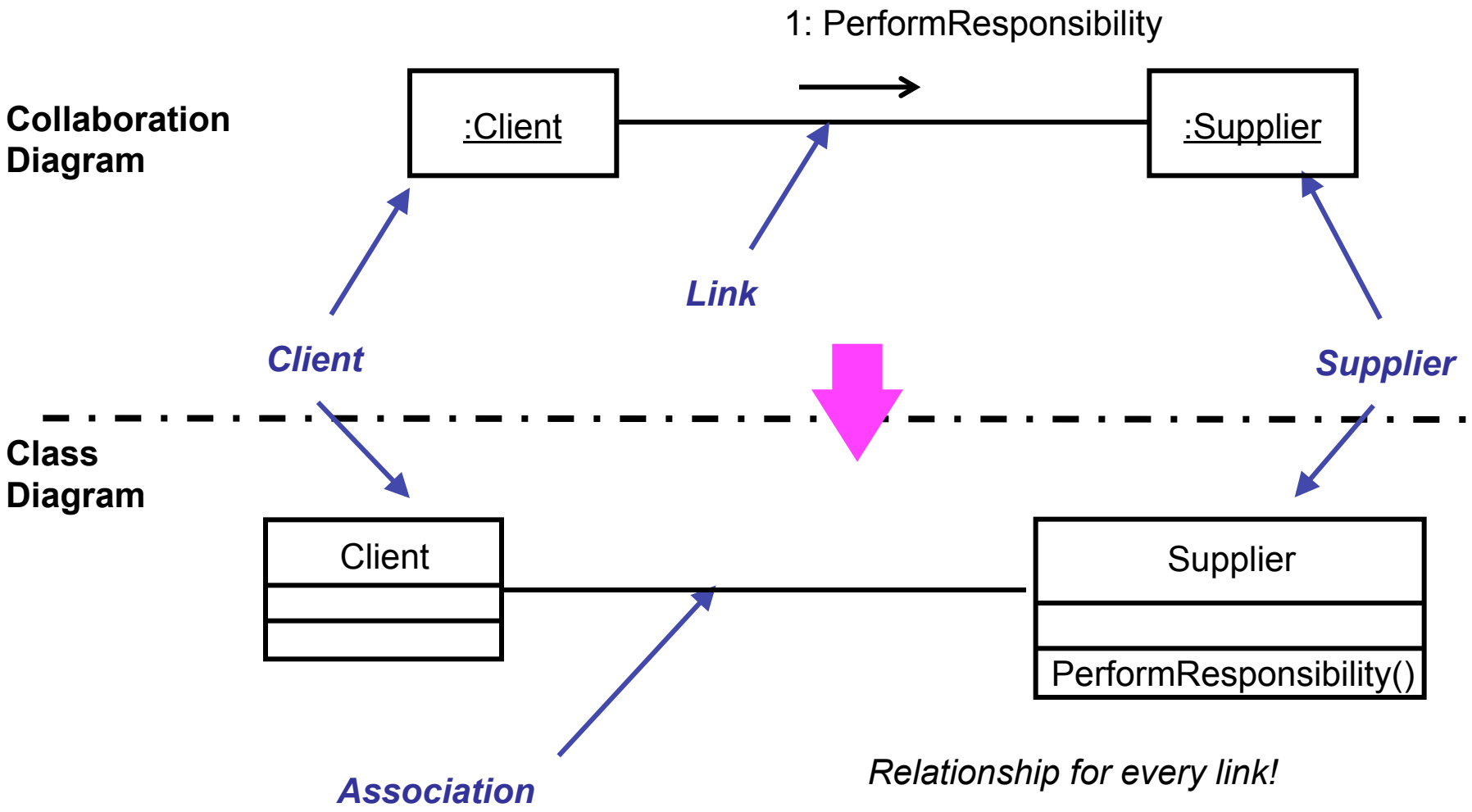


Figure From Rational Rose

What Is a Role?

- A role specifies the face that a class plays in an association.
- Role names are typically nouns or noun phrases.
- A role name is placed along the association line close to the class it modifies.
  - One or both ends of an association may have role names.

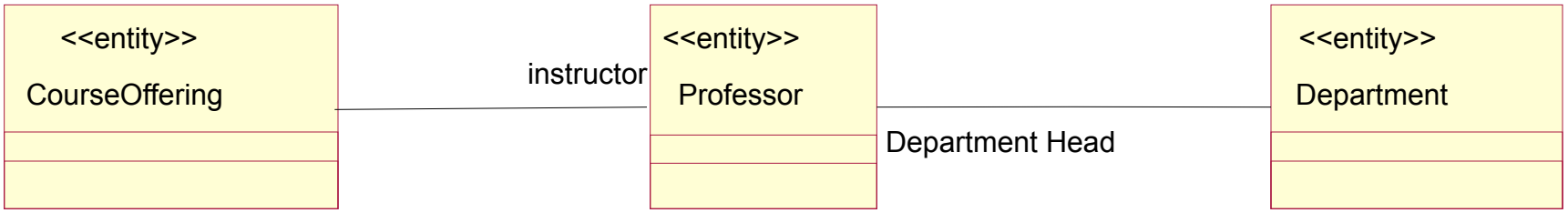
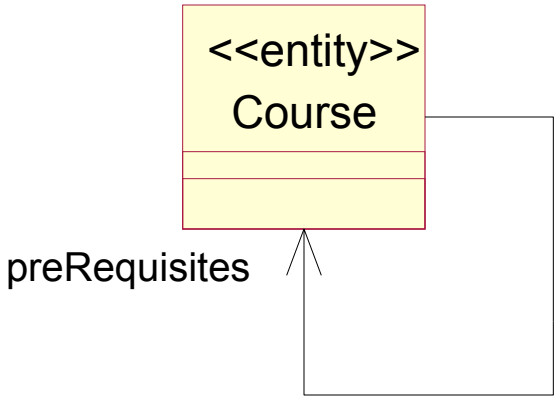


Figure From Rational Rose

Reflexive Associations and Roles

- In a reflexive association, objects in the same class are related.
  - Reflexive associations indicate that multiple objects in the same class collaborate together in some way.
  - Role names must be used in a reflexive association.



What is Multiplicity?

- Multiplicity or cardinality answers two questions.
  - Is the association mandatory or optional?
  - What is the minimum and maximum number of instances that can be linked to one instance?
- The following example expresses each student "MAY" register for many courses:

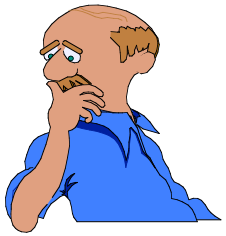
More Examples of Multiplicity

- Example of 1 to 1: Each Invoice MUST have a Date:

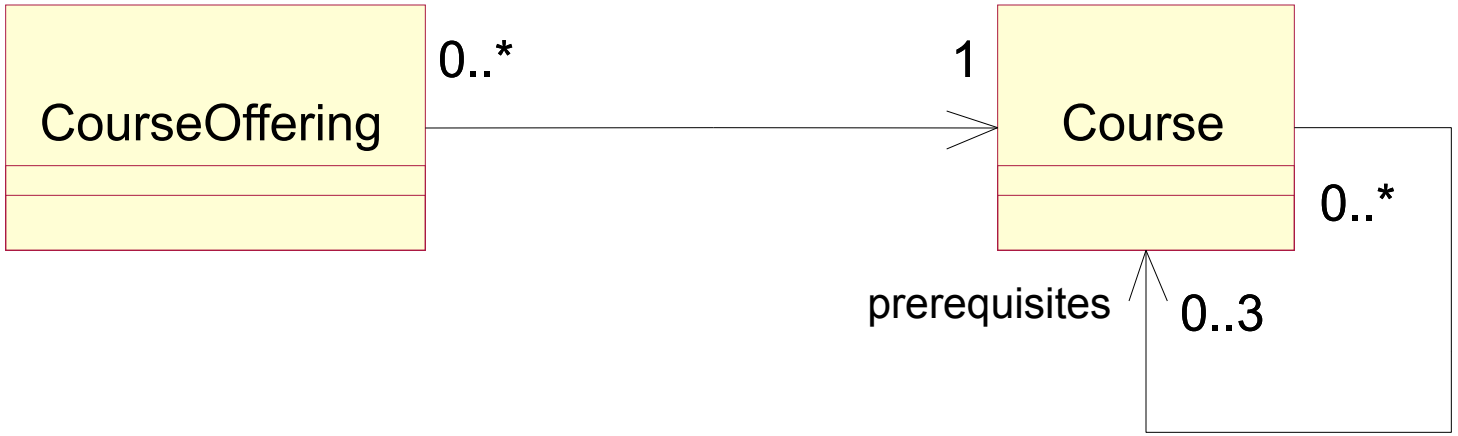
- Example of 1 to 1: Each Computer MAY be attached to a printer:

- Example of 1 to many: Each Computer May use several printers:

- Example of many to many o many Student May register for many courses:

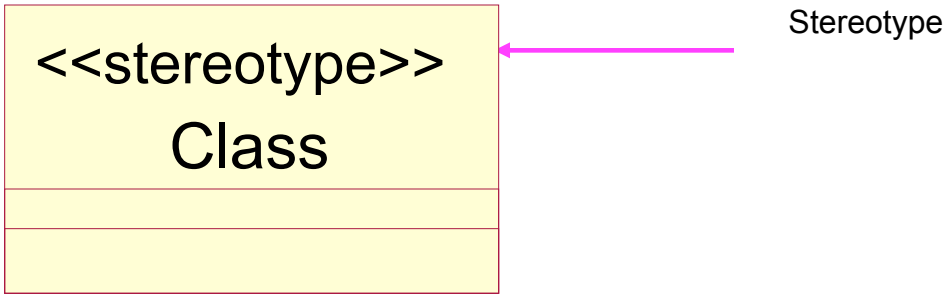


What Does Following Diagram Tell You?



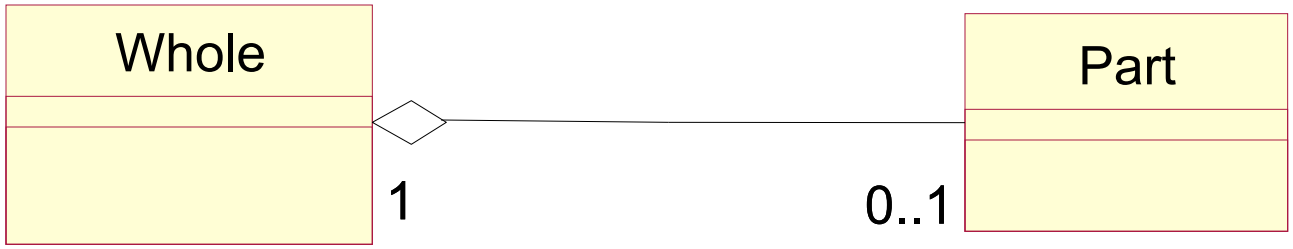
What Are Stereotypes?

- Stereotypes define a new model element in terms of another model element.
- Sometimes, you need to introduce new things that speak the language of your domain and look like primitive building blocks.



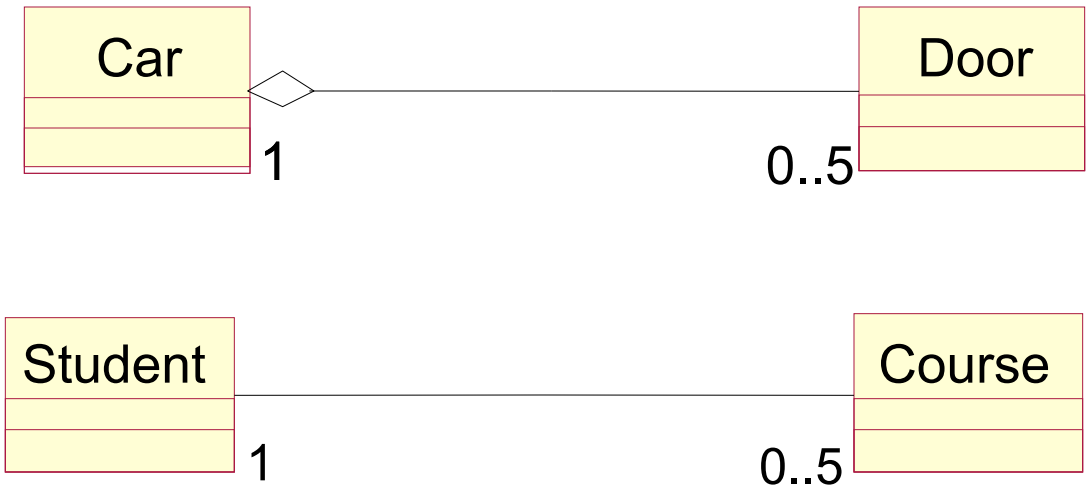
What Is Aggregation?

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

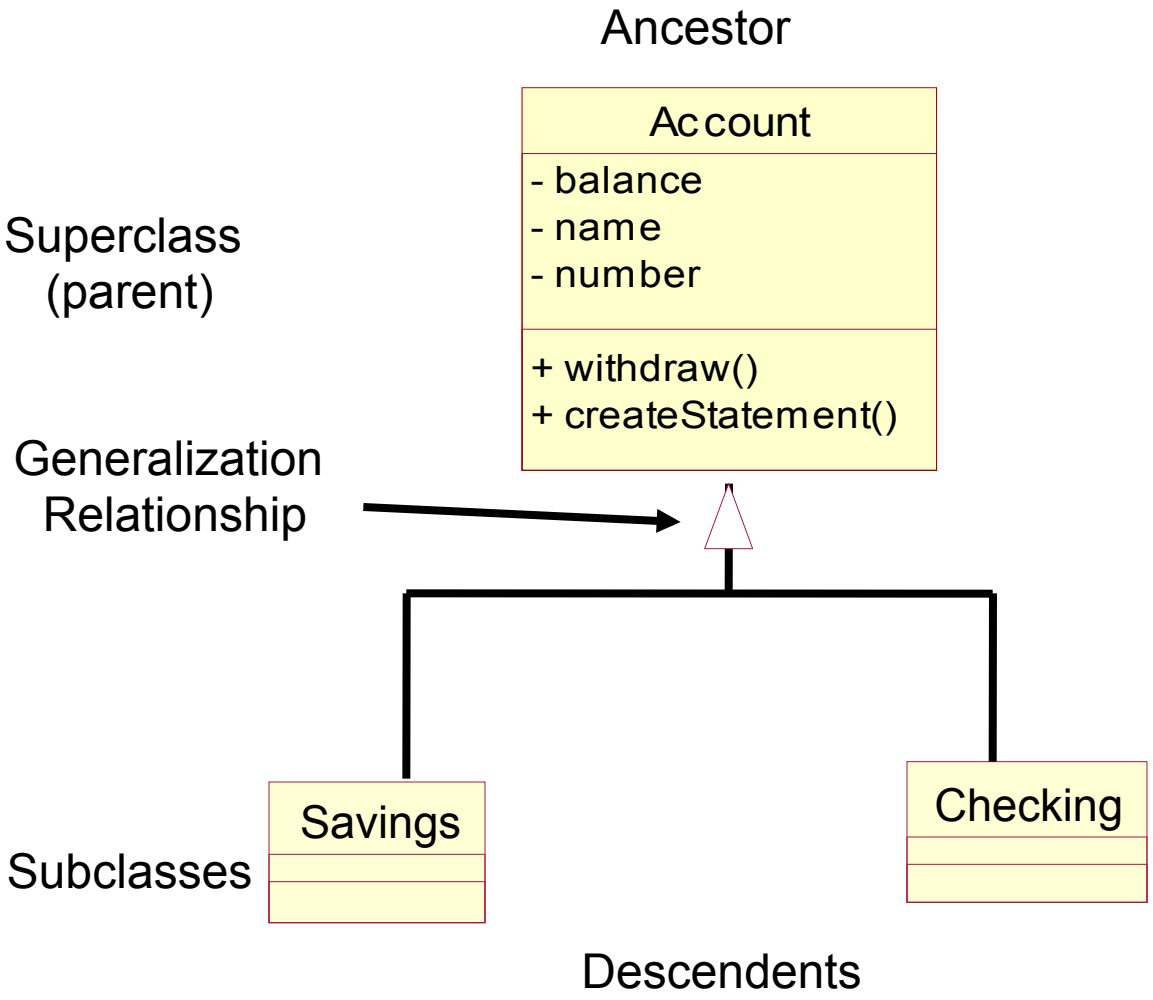


Association or Aggregation?

- If two objects are tightly bound by a whole-part relationship
  - The relationship is an aggregation.
- If two objects are usually considered as independent, although they are often linked
  - The relationship is an association.



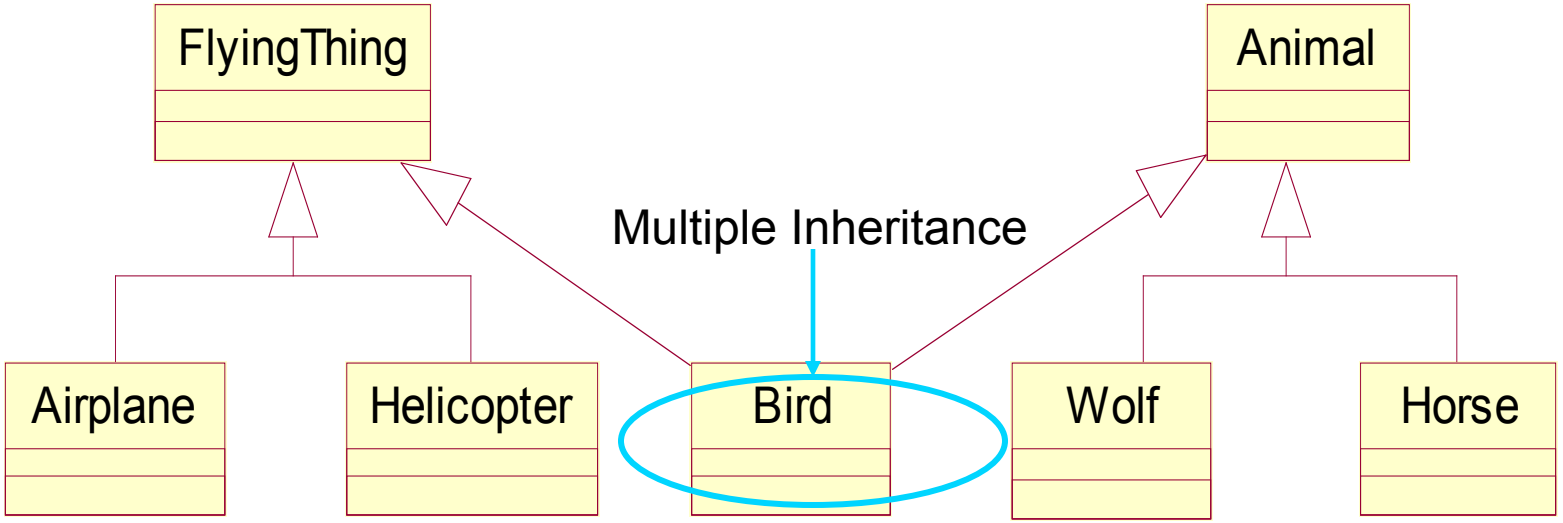
Class Relationship - Single Inheritance



The generalization is drawn from the subclass class to the superclass/ parent class.

The terms “ancestor” and “descendent” may be used instead of “superclass” and “subclass.”

Example: Multiple Inheritance

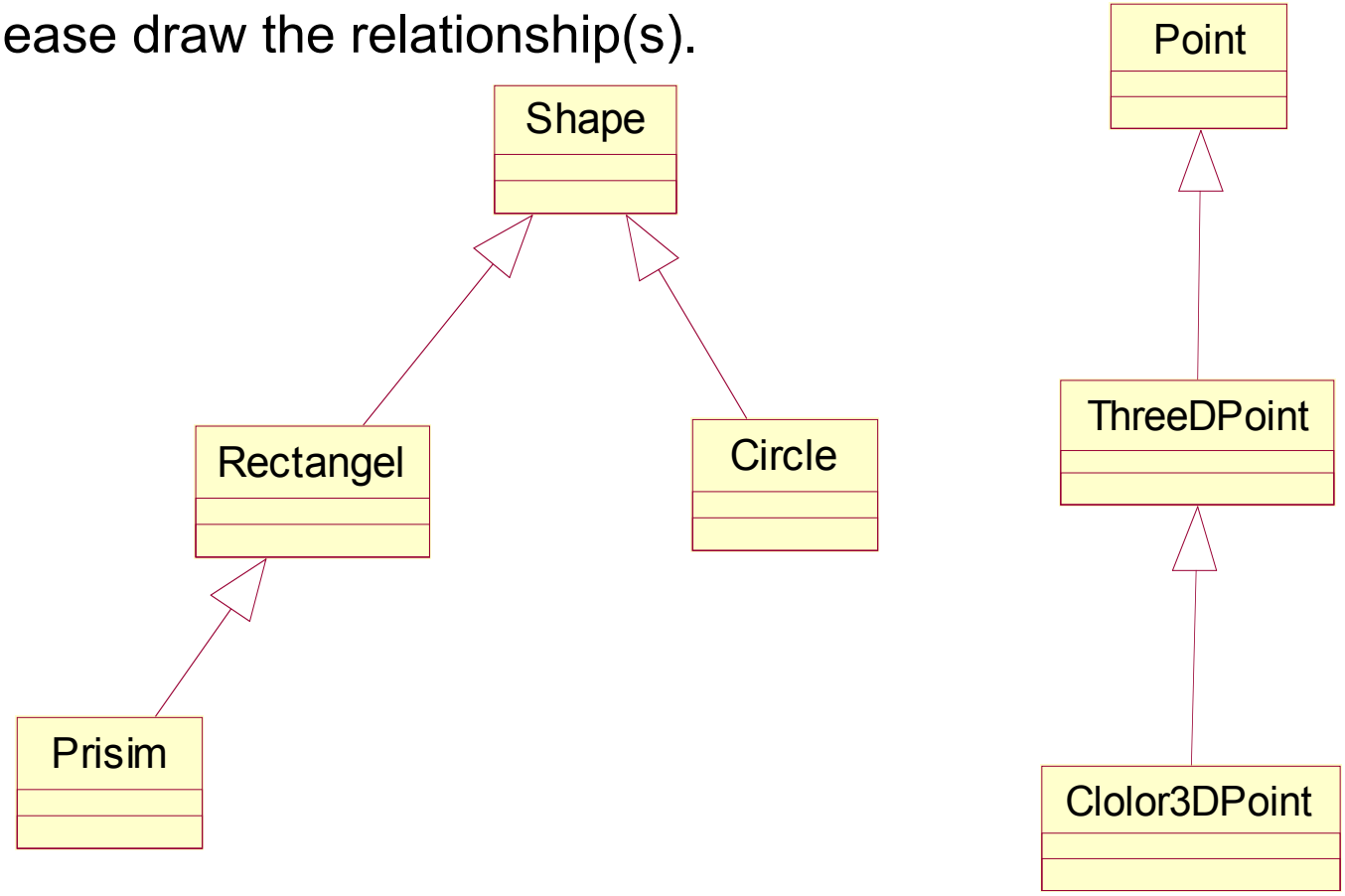


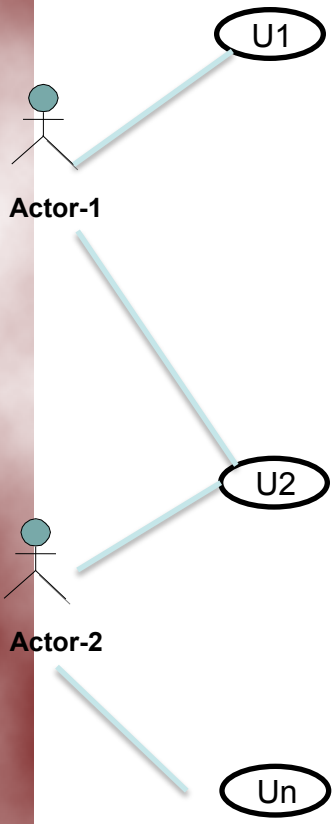
Use multiple inheritance only when needed and always with caution!

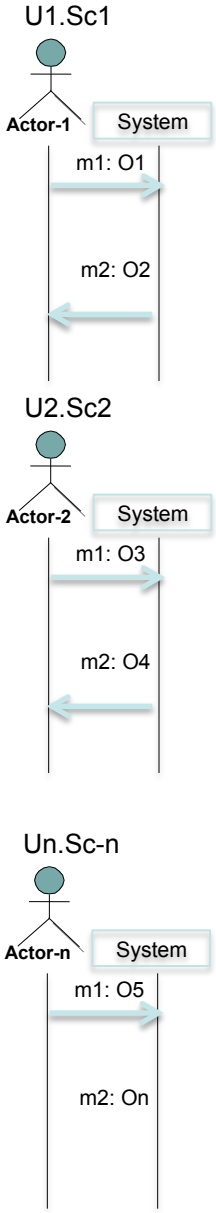
Figure From Rational Rose

Discussion: Aggregation and Inheritance

- We want each shape (rectangle, prism, and circle) to have a either a point or a 3-D point or a color-3D-Point.
- Please draw the relationship(s).



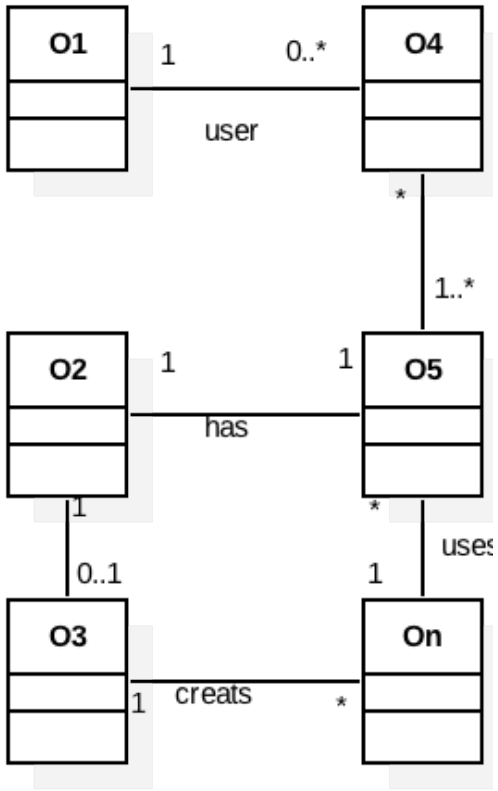




From behavior models. If behavior models are developed carefully with a domain expert, most of the domain concepts must have been explored.

At this stage we should extract all nouns from model. Then from list of extracted nouns we should select the concepts that are good candidates to be used in our conceptual/domain model.

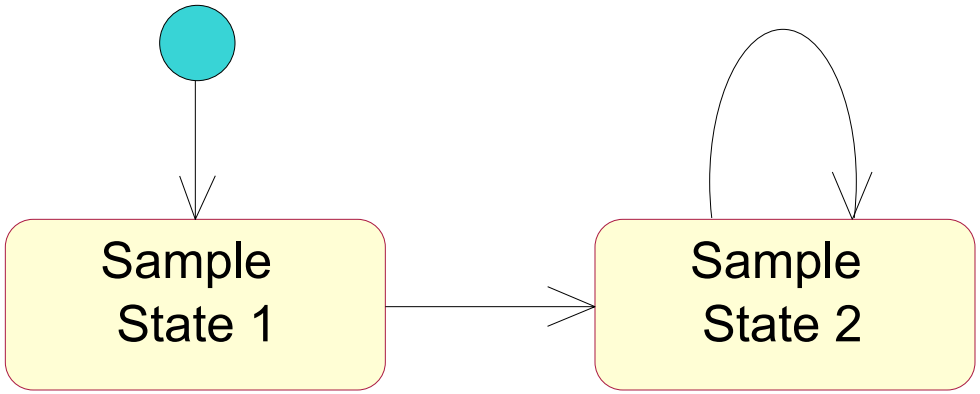
Good candidates are those concepts/nouns that have a structure and are not just simple nouns that can be implemented by int or string..



Activity and State Diagrams

What Are Statechart Diagrams?

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

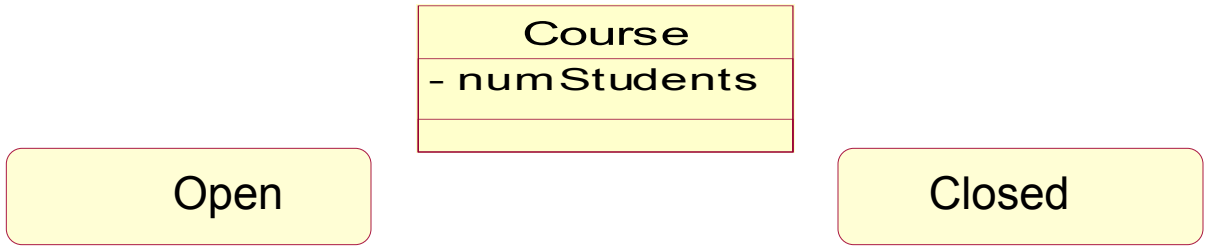


Where to Start When Determining State

- Initially, concentrate on the behavior of the classes with significant dynamic behavior.
- For a given class, look for possible states by
  - Evaluating attribute values.
  - Evaluating operations.
  - Defining the rules for each state.
  - Identifying the valid transactions between states.
- Examine message trace or interaction diagrams that involve the class being modeled.
  - The interval between two operations may be a state.

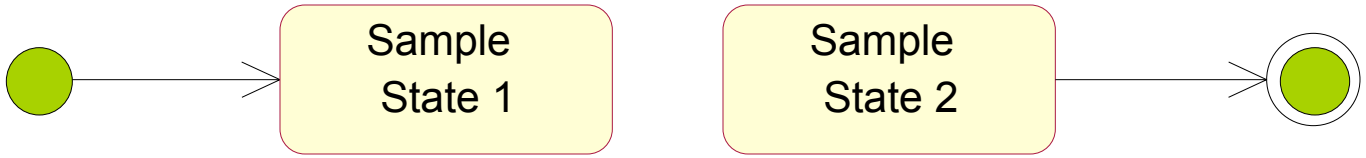
State and Attributes

- States may be distinguished by the values of certain attributes.
  - Example: If the maximum number of students
- States may be also distinguished by the links
  - Teaching when a link to a course exists.
  - On sabbatical when no link exists.



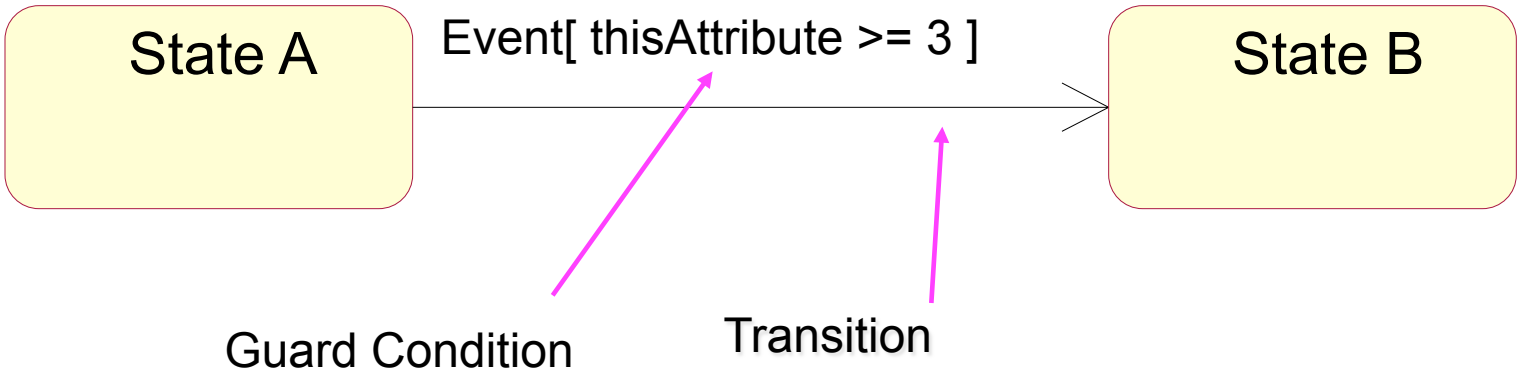
Special States

- 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.
  - More than one final state may exist.
  - A final state is indicated by a bull's eye.



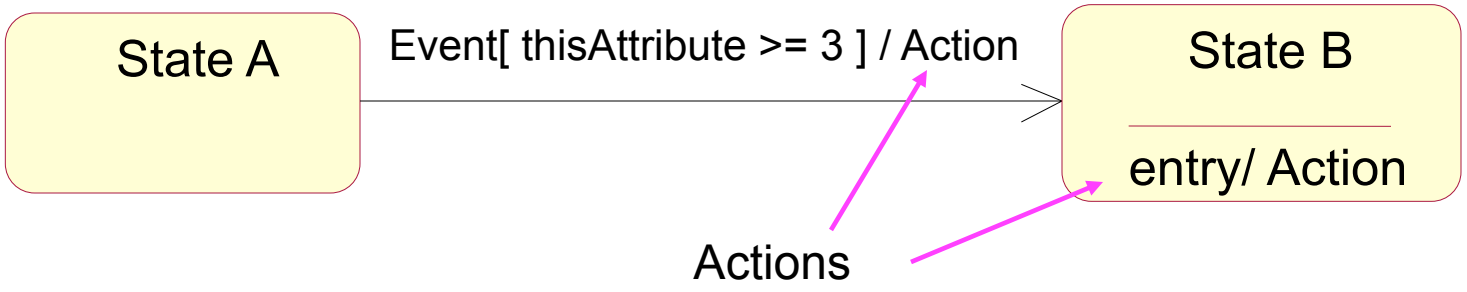
What Are Events and Guard Conditions?

- A Boolean expression that is evaluated when the transition is triggered by the reception of the event trigger. If the expression evaluates True, the transition is eligible to fire. If the expression evaluates to False, the transition does not fire.



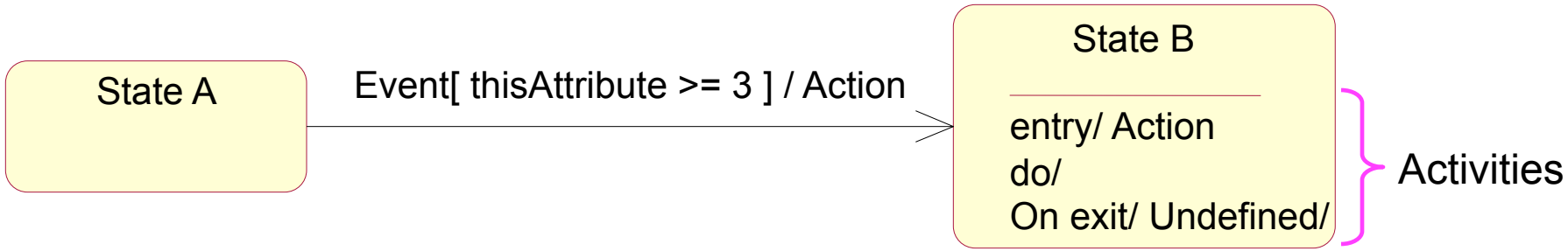
What Are Actions?

- Executable atomic computations that results in a change in state of the model or the return of a value
  - Associated with a transition
  - Take an insignificant amount of time to complete
  - Non-interruptible



What Are Activities?

- **Activity:** Ongoing, non-atomic executions within a state machine
  - An operation that takes time to complete
  - Starts when the state is entered
  - Can run to completion or can be interrupted by an outgoing transition



What is Activity Diagram?

Activity Diagram

- Activity diagram also describe the dynamic aspects of the system.
- Activity diagram similar to a flowchart illustrates the flow from one activity to another activity.
  - The activity can be described as an operation of the system.
  - The control flow is drawn from one operation to another.
- Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc
  - The flow can be:
    - sequential, branched, or concurrent.

What Is an Activity Diagram?

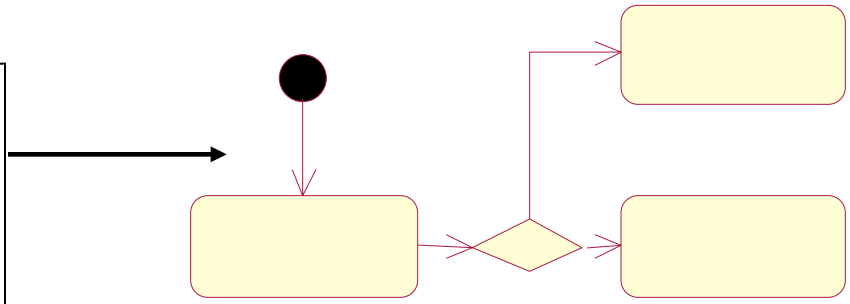
- An activity diagram in the use-case model can be used to capture the activities in a use case.
- It is essentially a flow chart, showing flow of control from activity to activity.

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.



The workflow of a use case consists of a sequence of activities that, together, produce something for the actor.
The structure of the workflow can be described graphically with the help of an activity diagram.

Figure From Rational Rose

Example: Activity Diagram

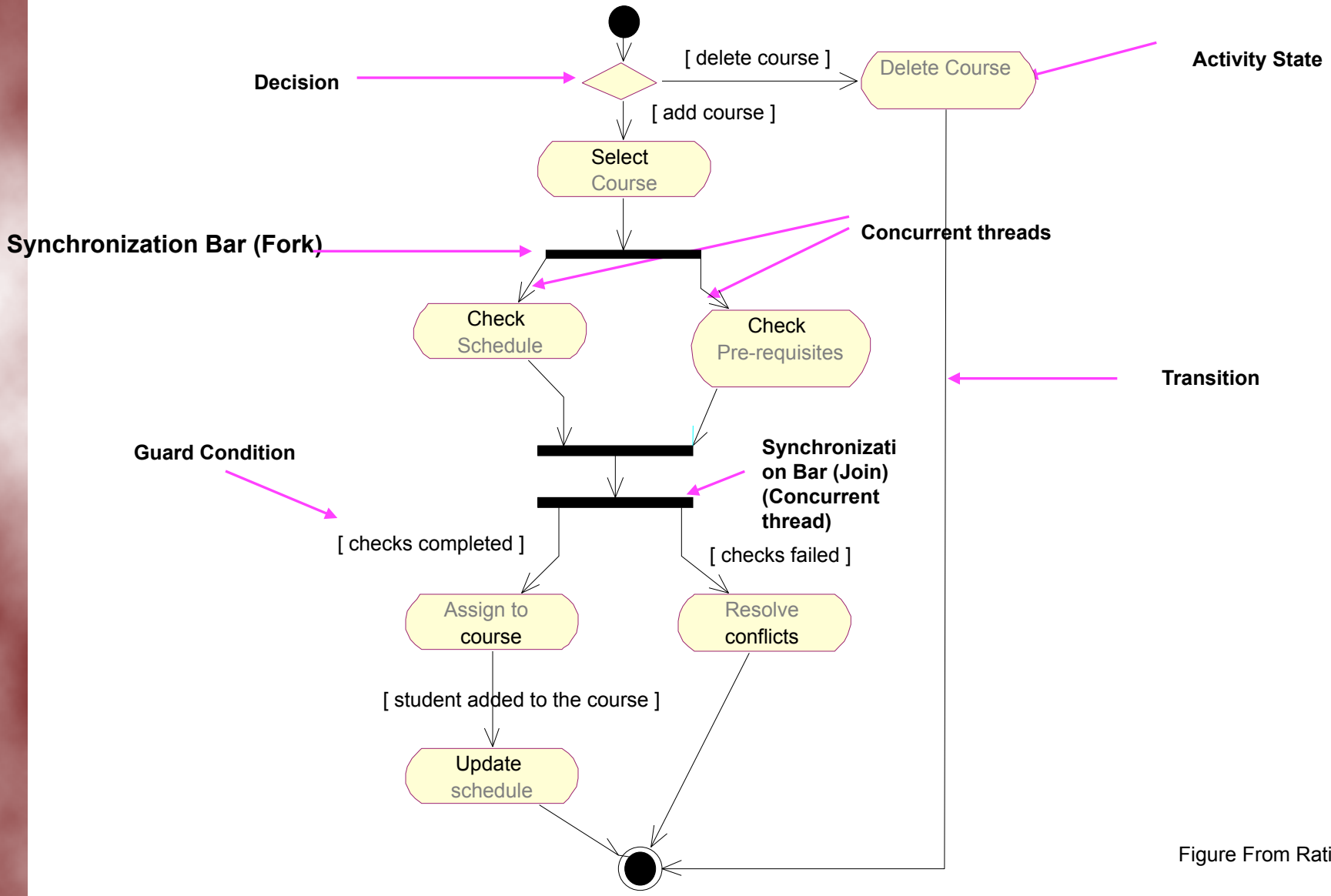
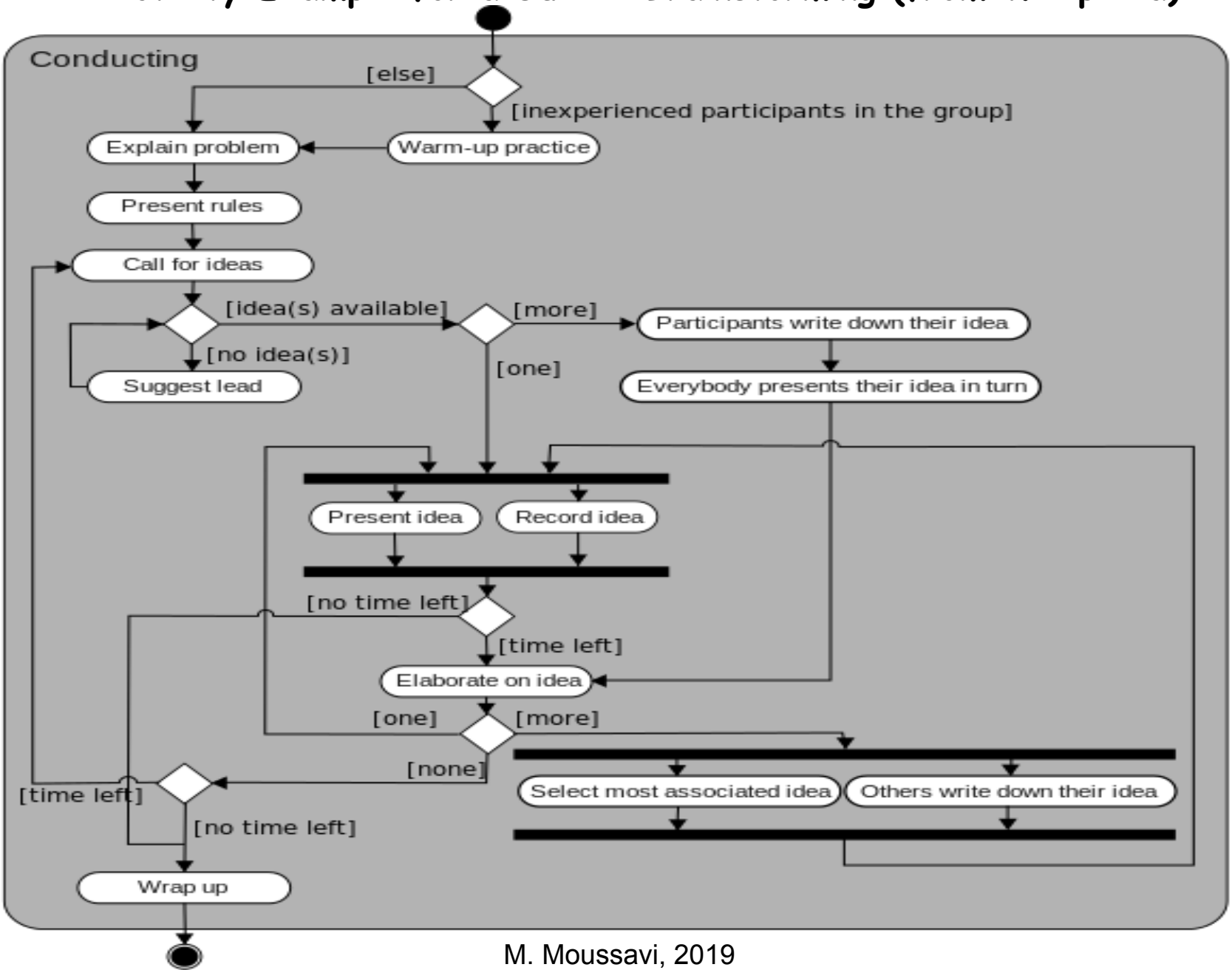
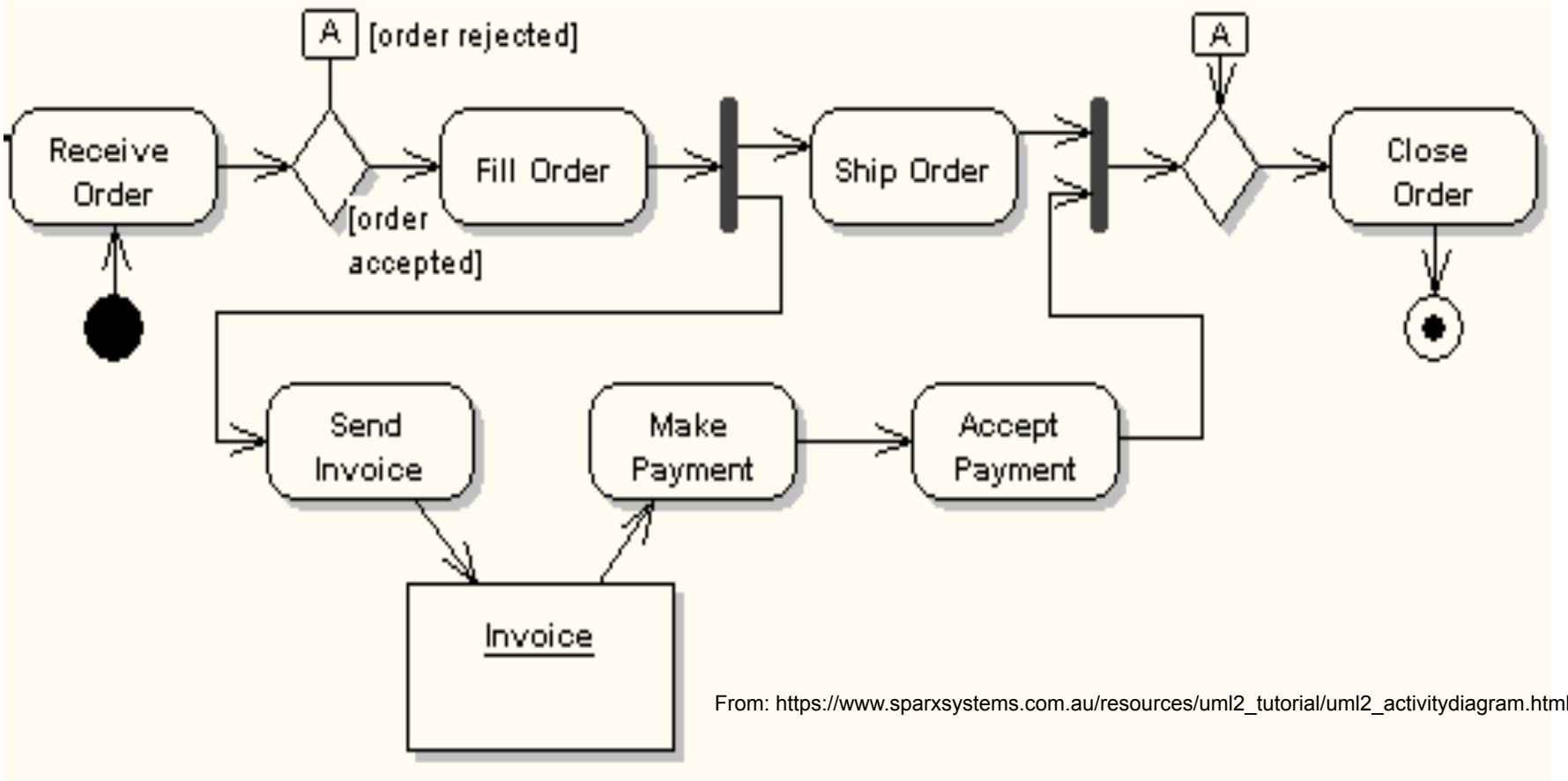


Figure From Rational Rose

Activity Example for a Guided Brainstorming (from Wikipedia)



Another Example: Placing Order



From: https://www.sparxsystems.com.au/resources/uml2_tutorial/uml2_activitydiagram.html