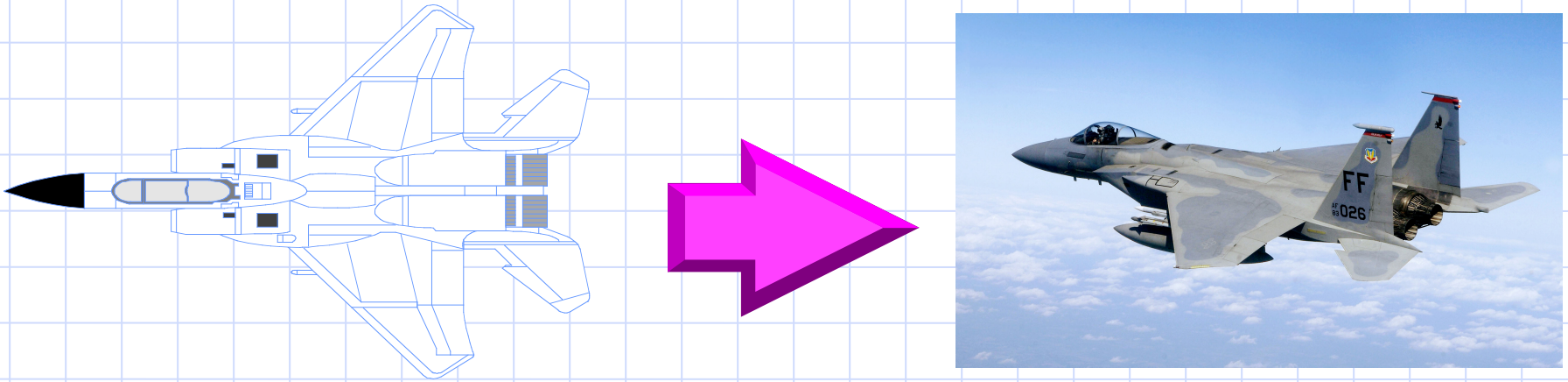
ENSF 614

More on UML and Application-Level Design

- A model is a simplification of reality.

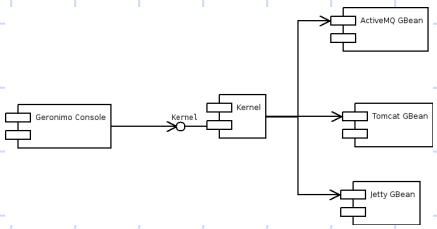


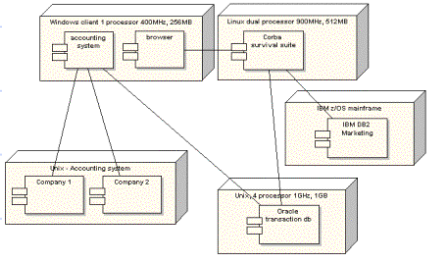
Why Build Models?

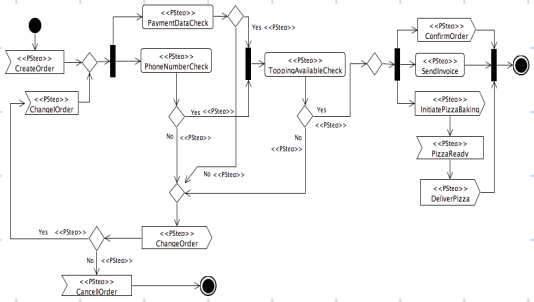
Make low-cost low risk changes and corrections.
Verify understanding.
Document decisions.
Educate / communicate knowledge or decisions.
Focus on important systems features.
Downplay less important features

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.







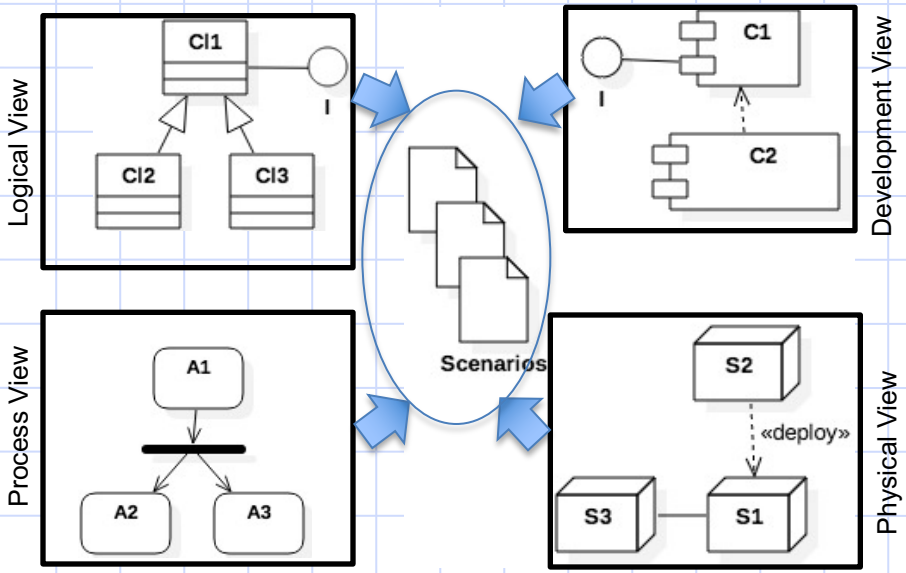
UML 2.5 Models

- 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

Use Case Modelling

Importance of Use Case Model

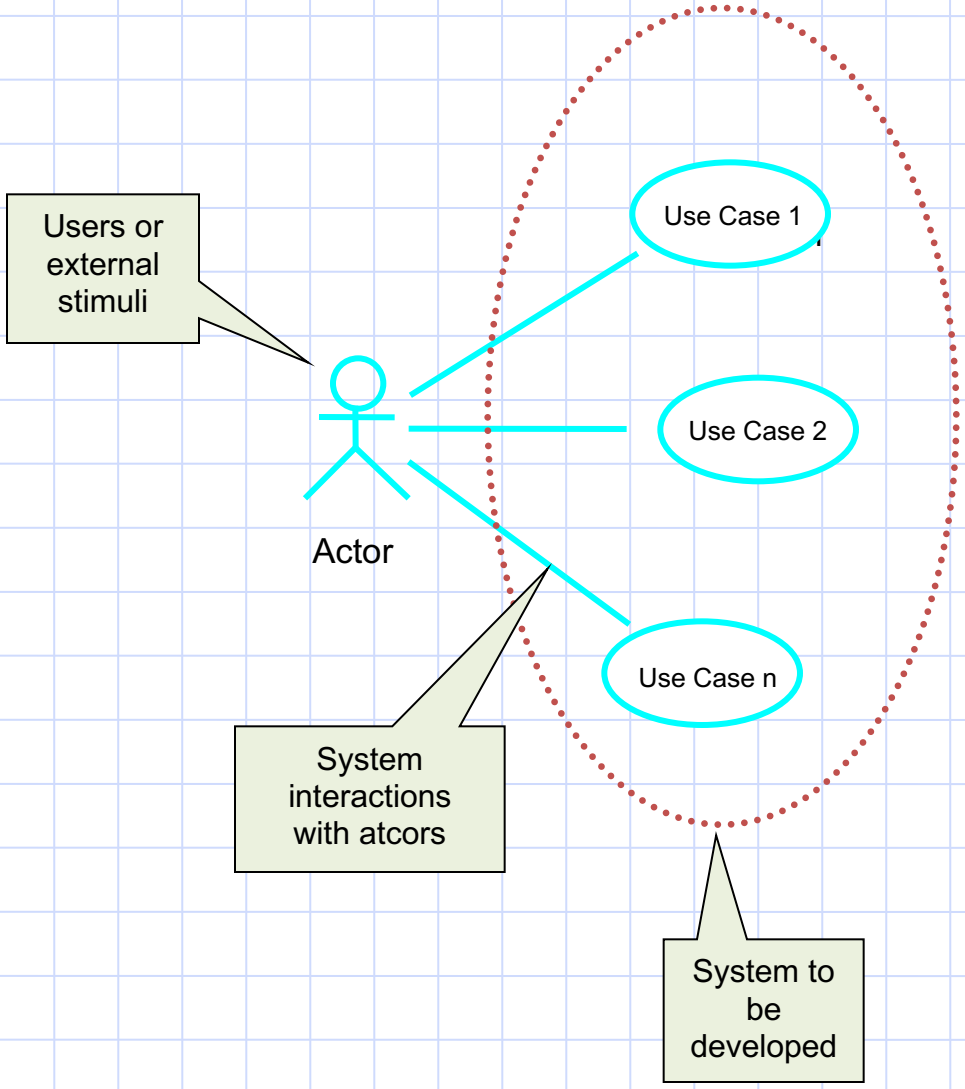
- Use case diagram are central to many activities and other artifacts of a system during the analysis and design period. The following figure from Wikipedia is good illustration of this claim :



Use-Case Modelling

What Is a Use-Case Model?

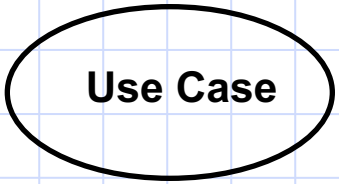
- A model that describes a system's functional requirements in terms of use cases
- A model of the system's intended functions (use cases) and its environment (actors)
- Used to communicate with the end users and domain experts



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

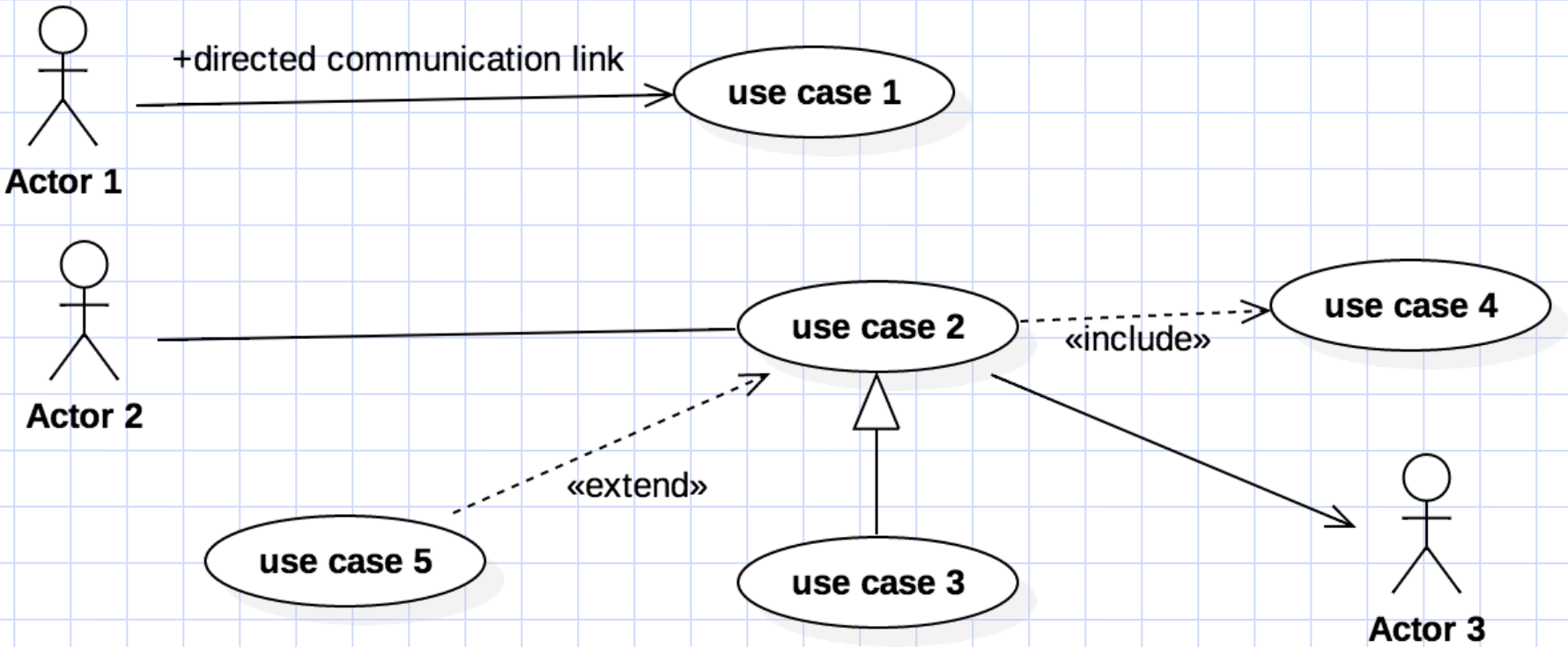


Focus on the Roles

- 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.

Use Case Diagram Details

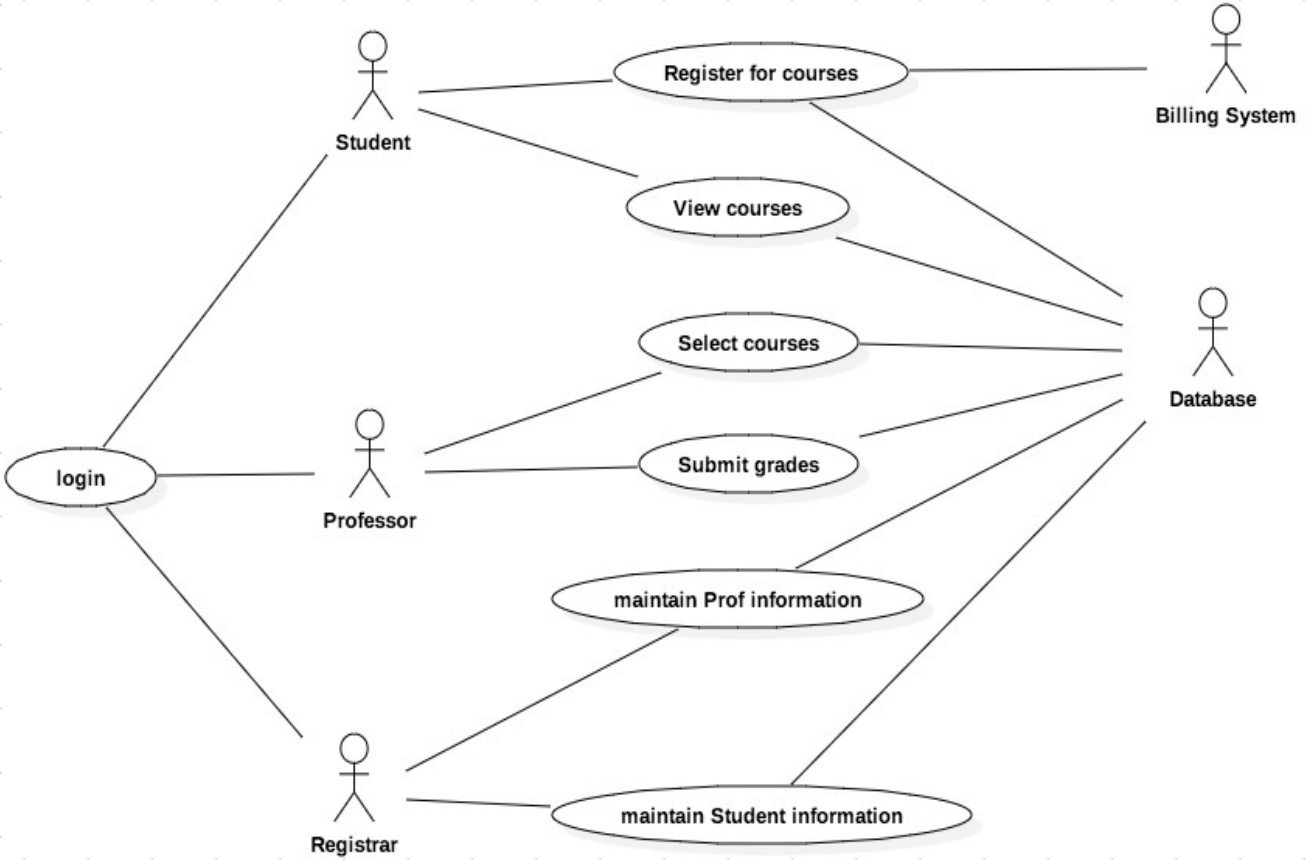
- A use case diagram is a dialog between actors and the system.
- Remember a use case diagram is not a flow chart



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:

1. What doesn‘t this model say?
2. Describe the functionality of this system.
3. Describe the actor relationship for the Maintain Student Information.
4. Describe the actor relationship for Select Courses To Teach use case.
5. Which use case needs to run first, Register for Courses or View Courses?

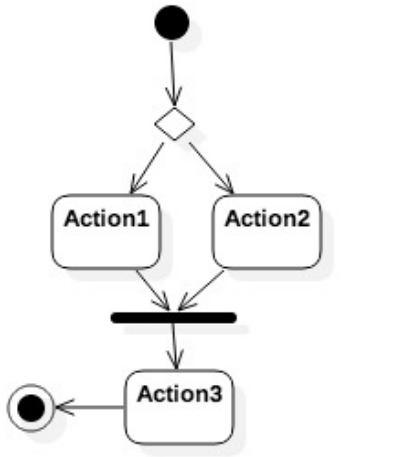
Use-Case Model/Specifications

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

Using Activity Diagram in Use Cases

The following diagrams can be used as additional documentation in a use-case model to to capture the sequence of activities and/or the possible states of system.

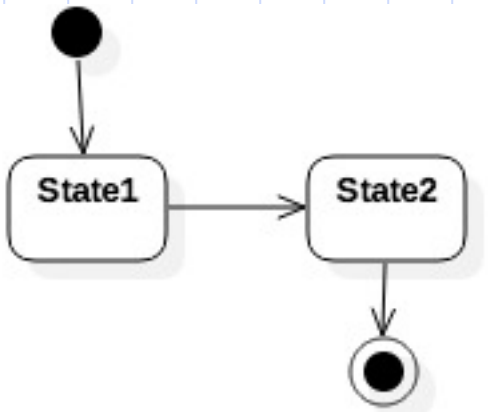
- An activity diagram: which is more like a flow chart, showing flow of control from one action to another action.
- State transition diagram.



Example of Flow of Events:

1. **This use case starts when the Student requests to register for a course**
2. **The system checks to see if:**
   - **course is being offered**
   - **course is not full**
3. **If all conditions are satisfied, system asks student to pay for the registration fee.**
4. **If payment succeeds, course will be added to the student's schedule.**

The workflow of a use case consists of a sequence of action that produces something for the actor.
This workflow can be described graphically with the help of an activity diagram.



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

A Sample Use-Case Scenario

The "Create A Schedule Scenario"

John enters the Student ID number 300-0000111 and the system validates the number.
System asks select 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, Programming 110, History 200, and Algebra 210.
John then selects the optional courses: Music 101.
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 John's schedule and sends billing information for five courses to the billing system for processing.

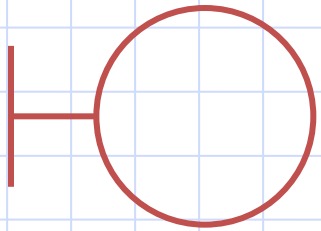
- Filter nouns from this description.

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
Programming 110	candidate object
History 200	candidate object
Algebra 210	candidate object
Music 101	candidate object

Different Types of Classes

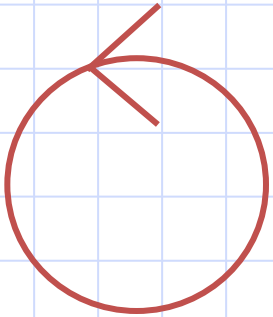
What Is a Boundary Class?



<<boundary>>

- Models the interaction between the system's surroundings and its inner workings
  - User-interface classes
  - Device-interface classes
  - System-interface classes

What Is a Control Class?



<<control>>

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

Control Class Dependency

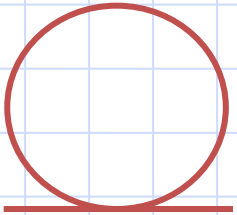
A control class is dependent on the use case to which it belongs. If the use case changes, its flow of events changes. The control class will likely need to change, too.

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.

Hints for Using a Control Class

- Control classes should only do sequencing.
- A control class should tell other classes to "do something" and should never "do anything" except for directing.
- Use control classes to decouple boundary and entity classes.
- 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)

What Is an Entity Class?



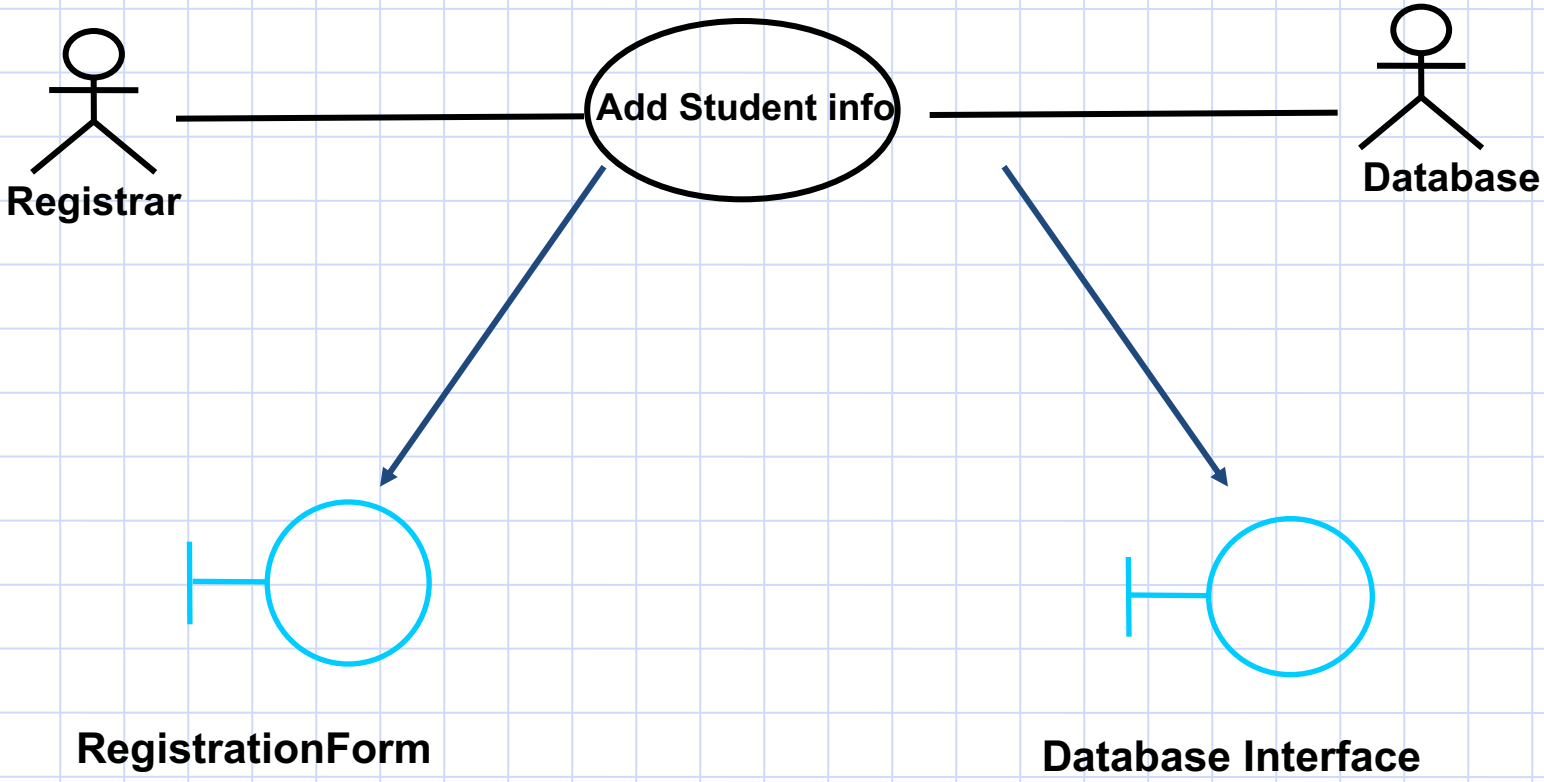
<<entity>>

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

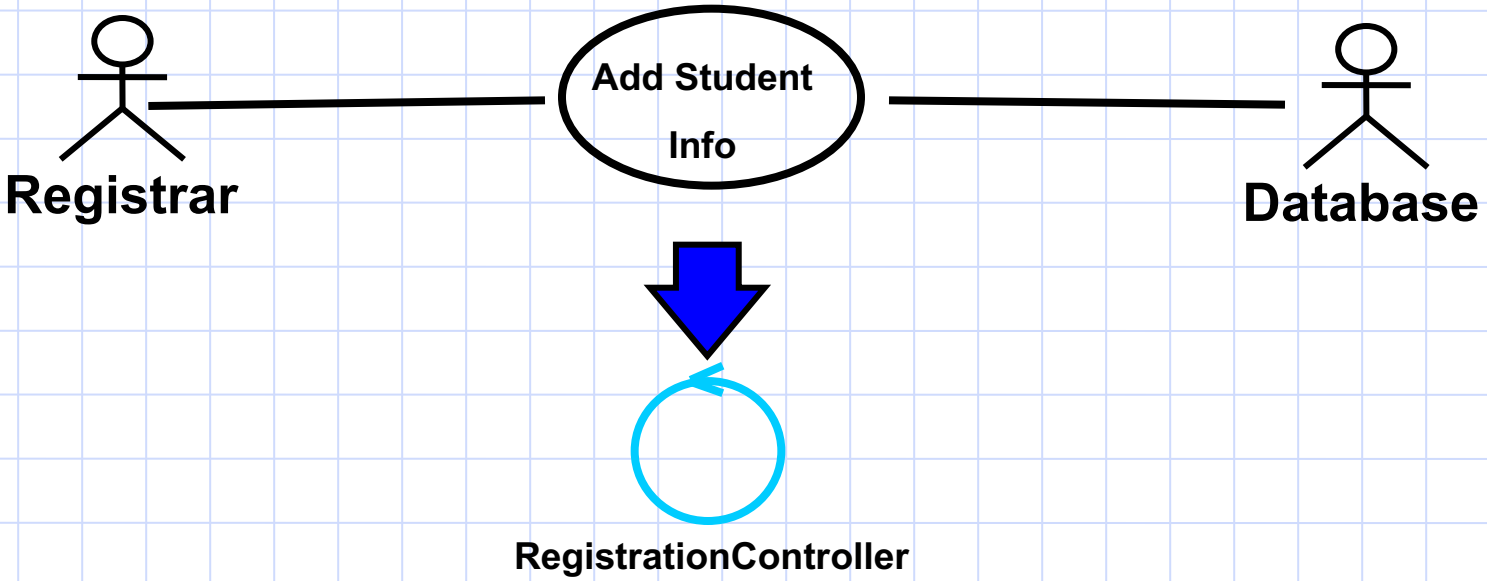
How to Find Boundary, Control, and Entity Classes

Finding Boundary Classes

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



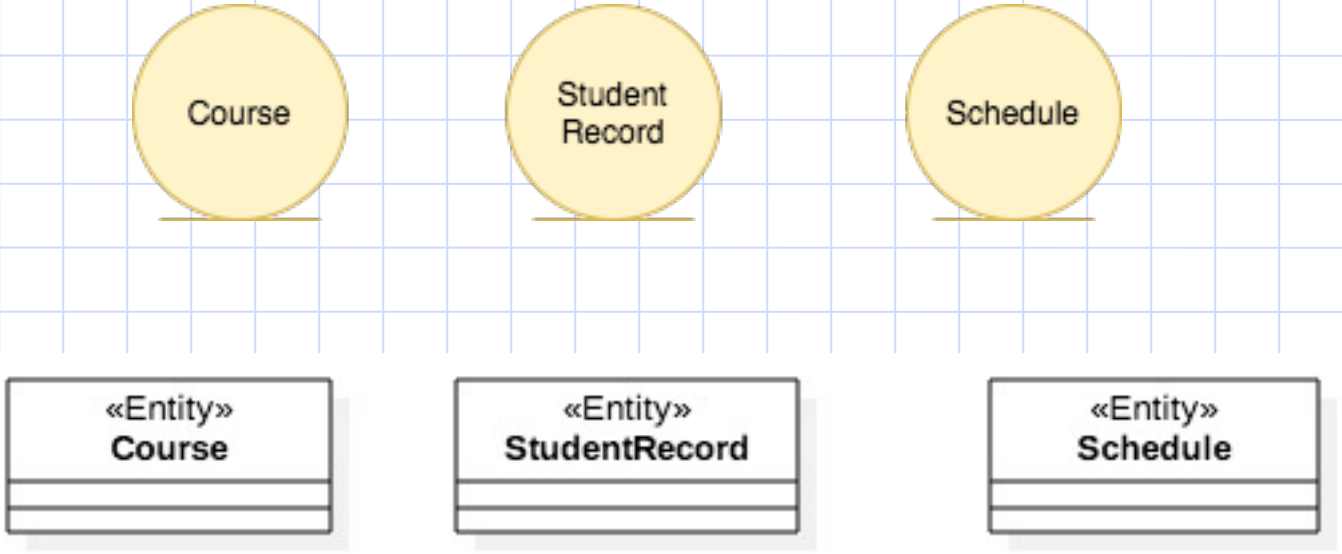
Finding Control Classes



- 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

Example: Entity Classes

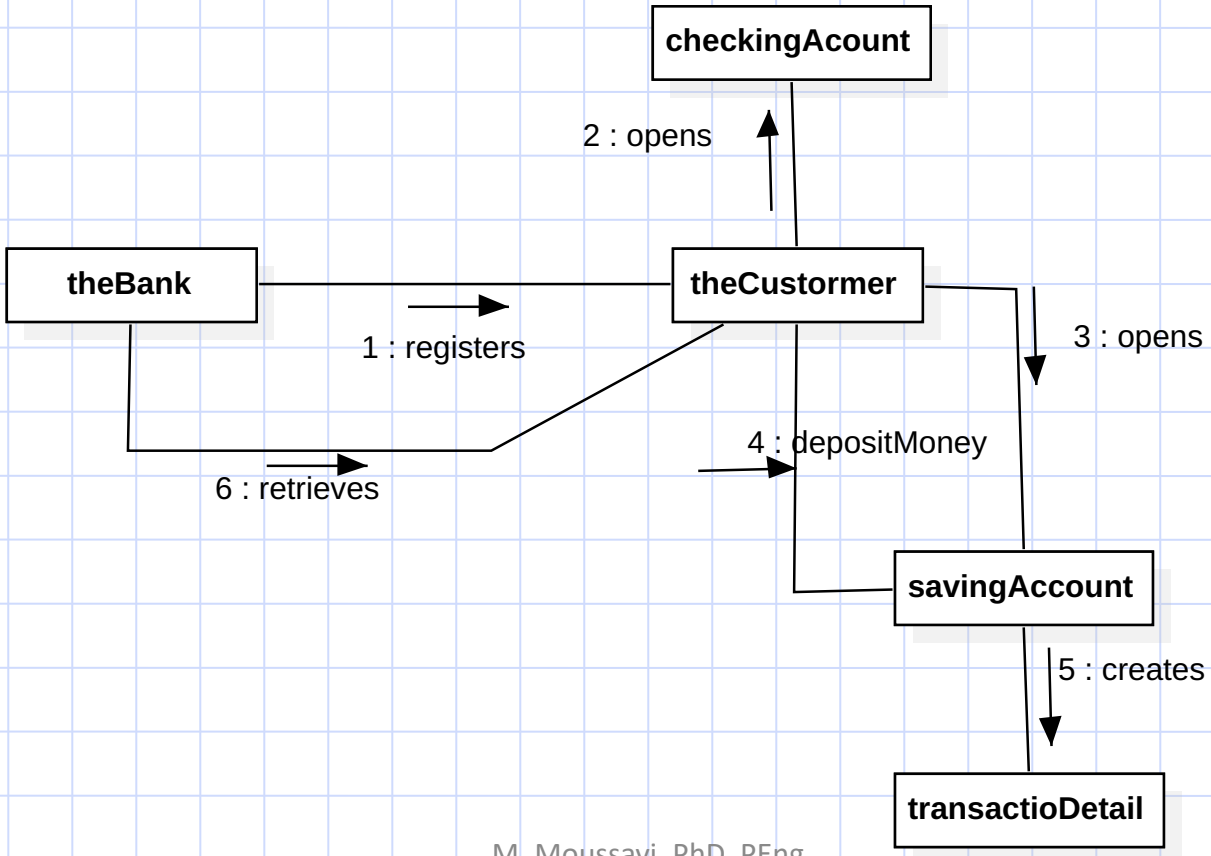
- Schedule: A list of student courses for a semester.
- Student Record: List of previously taken courses by the student
- Course Offerings: An offering for a semester



How to Find Relationships

How to Choose Associations

- Associations can be discovered from Interaction diagram (sequence and communication).

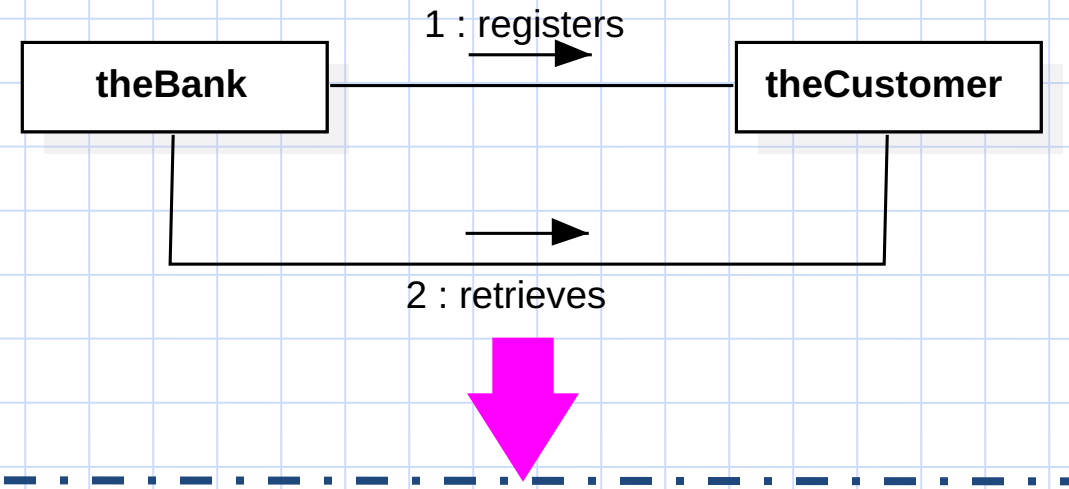


Associations and Analysis-Class Diagram

- During the analysis it would be fine to have multiple associations among classes. It will help a better communication between stakeholders.

Associations and Design-Class Diagram

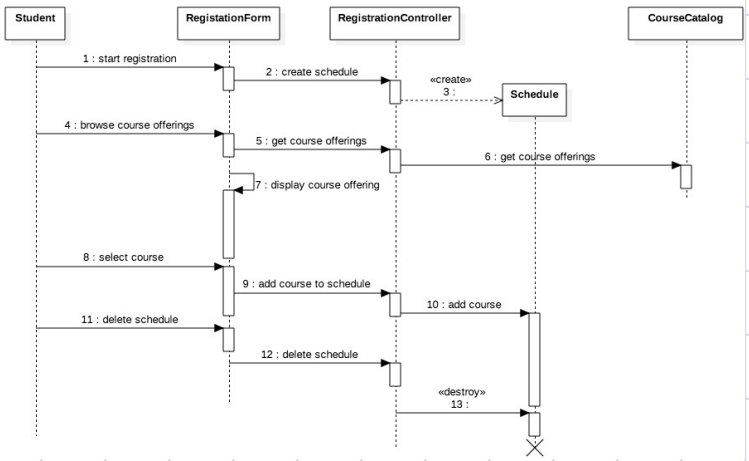
- Several instances of communication message can be translated to a general form of association during the design phase:

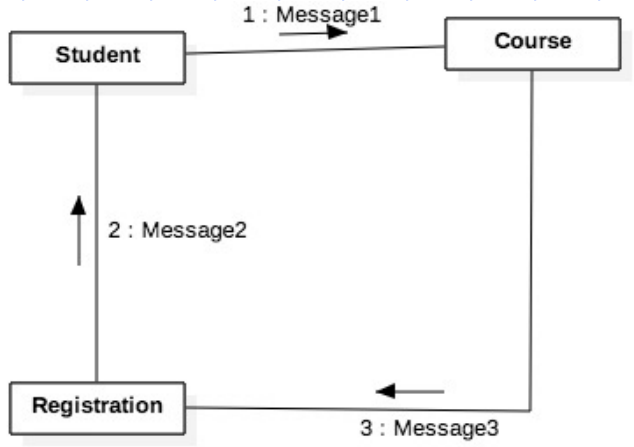


Object Collaborations and Responsibilities

What Is an Interaction Diagram?

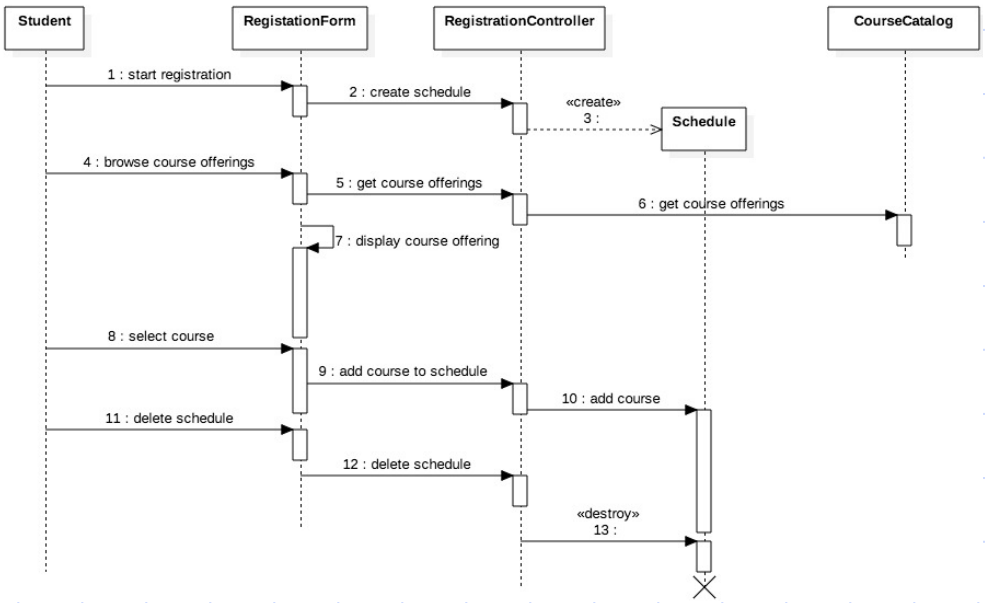
- An interaction diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them.
- It models the dynamic aspects of a system. There are two types of interaction diagrams:



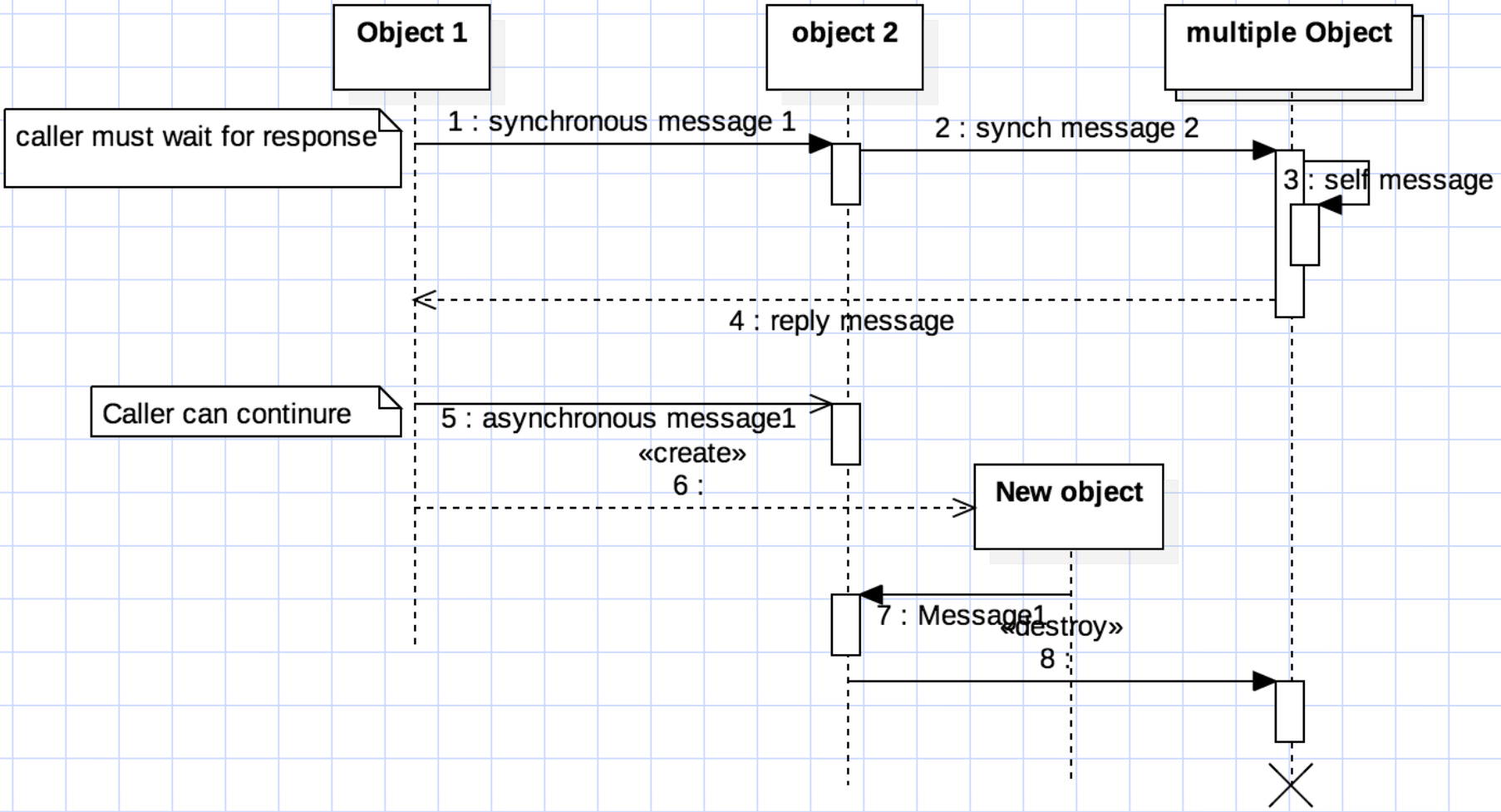


What Is a Sequence Diagram?

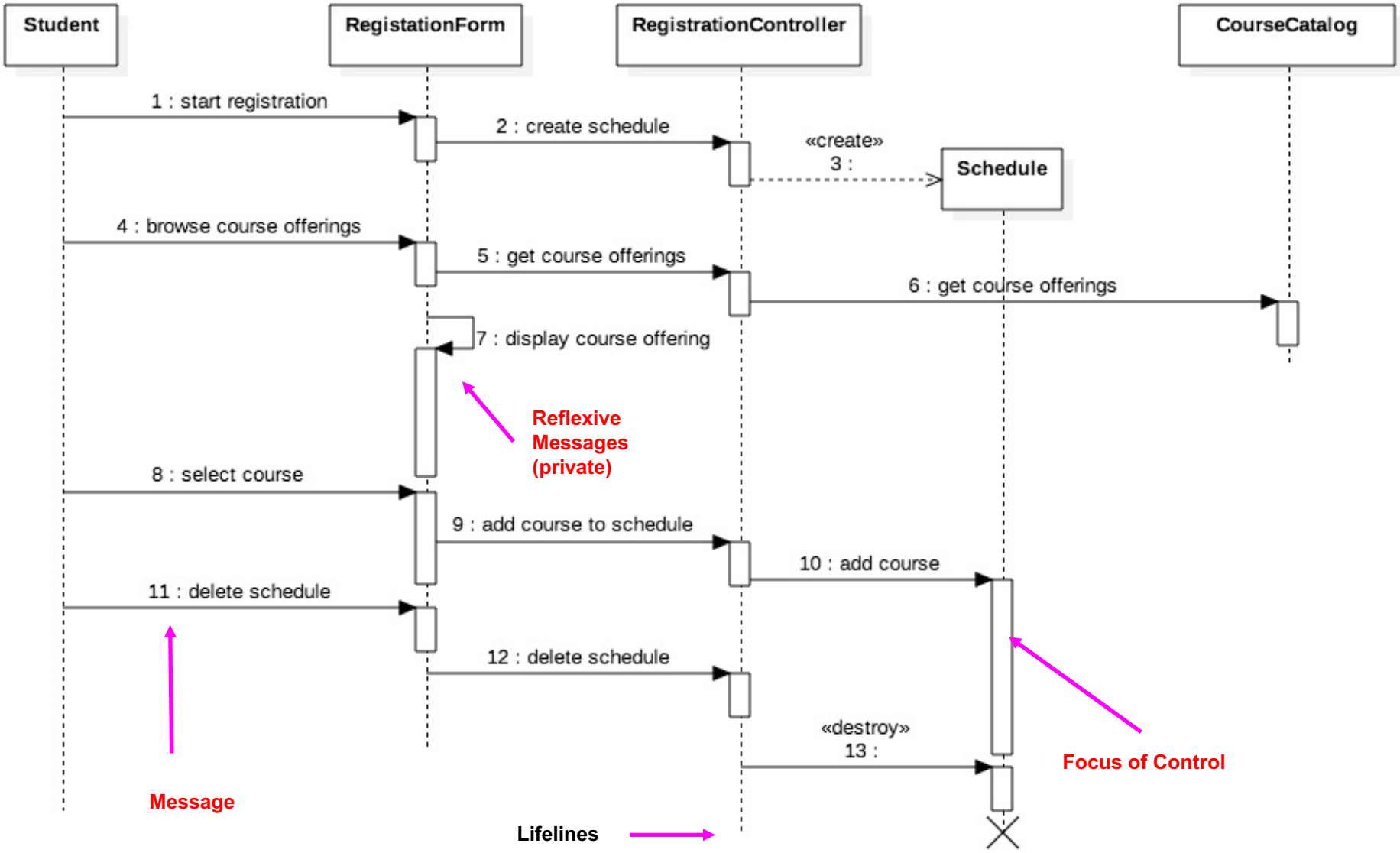
- A sequence diagram is an interaction diagram that emphasizes the time ordering of messages.
- The diagram shows
  - The objects participating in the interaction.
  - The sequence of messages exchanged.



Sequence Diagram Details



Example: Sequence Diagram



The **focus of control** is a tall, thin rectangle that shows the period of time during which an object is performing an action

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