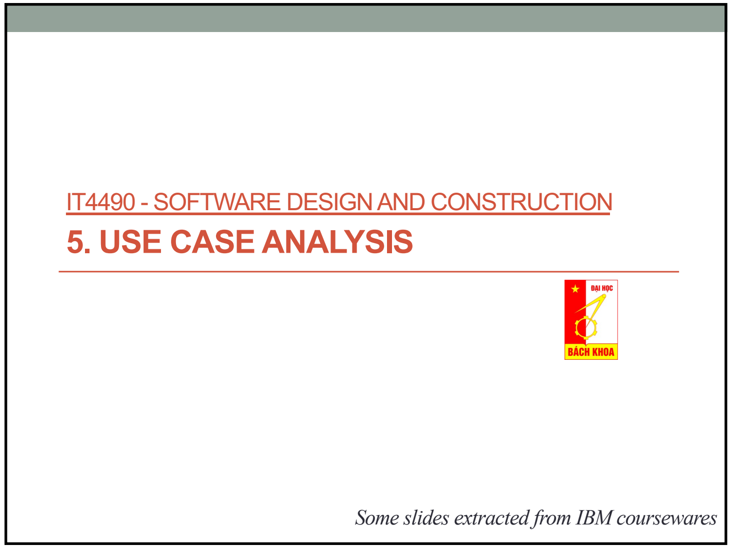


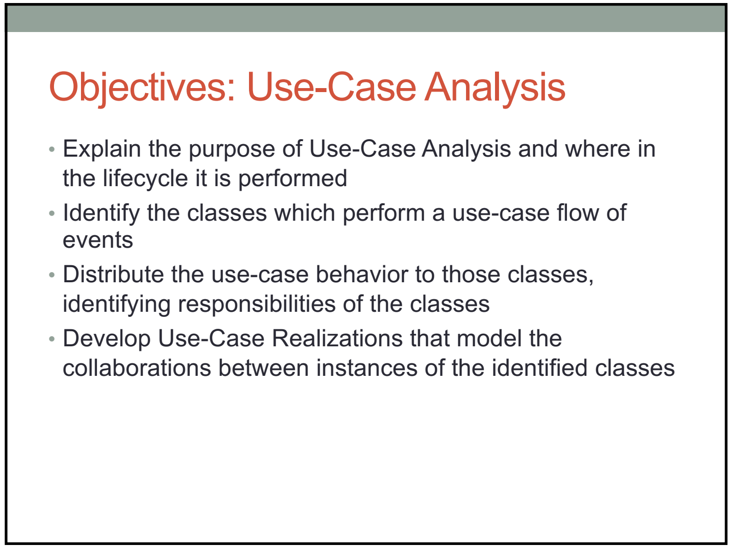
IT4490 - SOFTWARE DESIGN AND CONSTRUCTION

5. USE CASE ANALYSIS

ĐẠI HỌC BÁCH KHOA

Some slides extracted from IBM coursewares

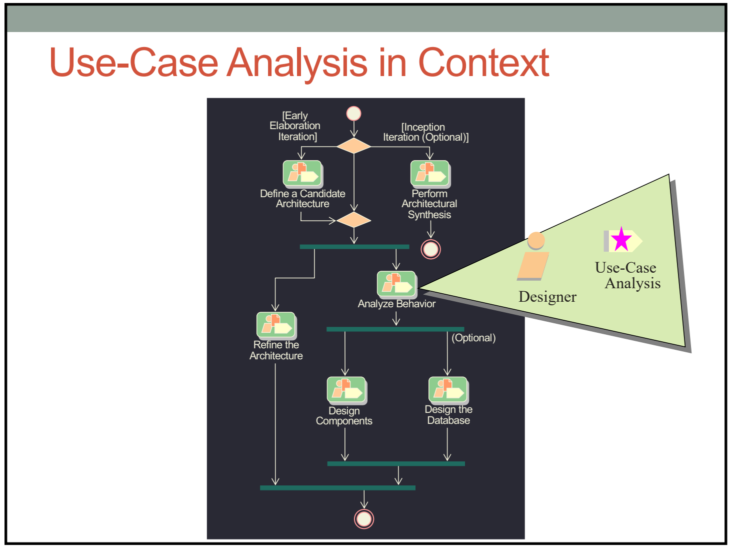
1



Objectives: Use-Case Analysis

- Explain the purpose of Use-Case Analysis and where in the lifecycle it is performed
- Identify the classes which perform a use-case flow of events
- Distribute the use-case behavior to those classes, identifying responsibilities of the classes
- Develop Use-Case Realizations that model the collaborations between instances of the identified classes

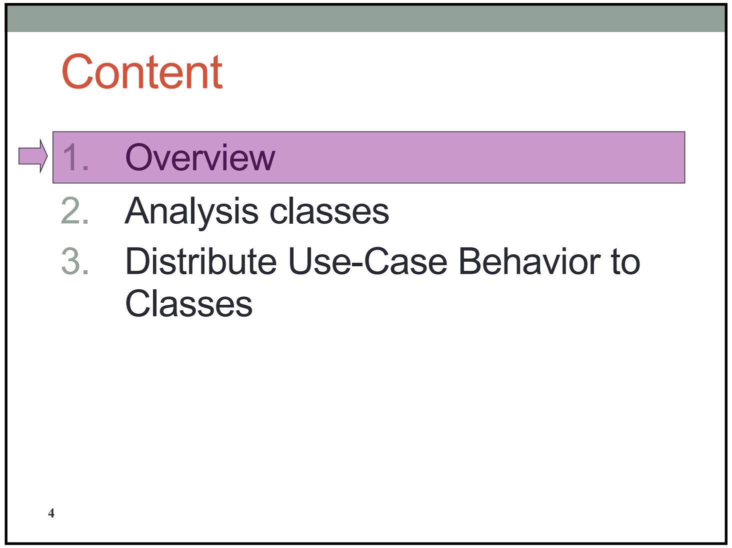
2



Use-Case Analysis in Context

[Early Elaboration Iteration]
[Inception Iteration (Optional)]
Define a Candidate Architecture
Perform Architectural Synthesis
Analyze Behavior
Refine the Architecture
(Optional)
Design Components
Design the Database
Designer
Use-Case Analysis

3

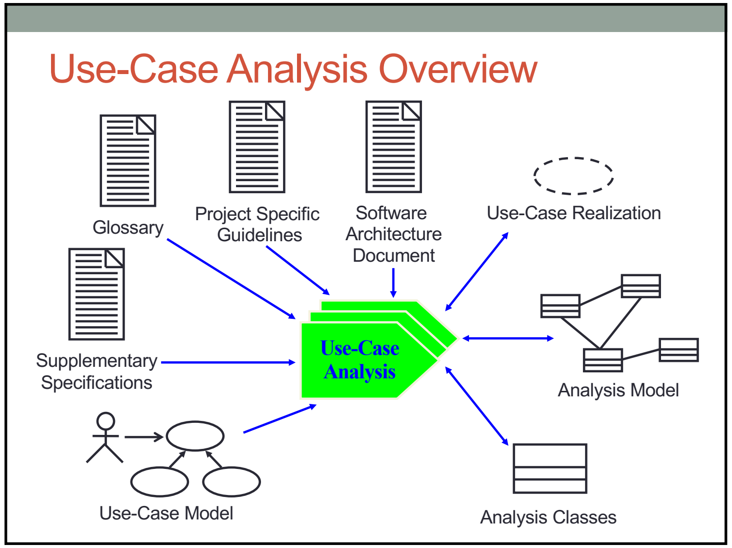


Content

1. Overview
2. Analysis classes
3. Distribute Use-Case Behavior to Classes

4

4



Use-Case Analysis Overview

Glossary

Project Specific Guidelines

Software Architecture Document

Use-Case Realization

Supplementary Specifications

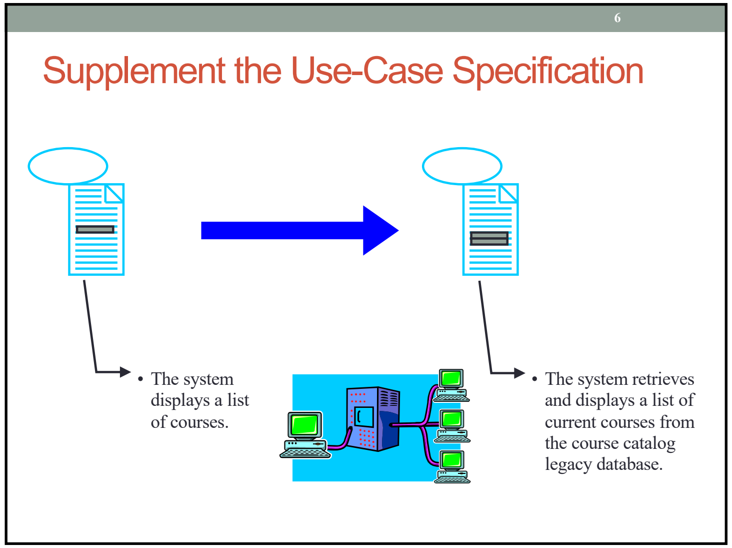
Use-Case Analysis

Analysis Model

Use-Case Model

Analysis Classes

5

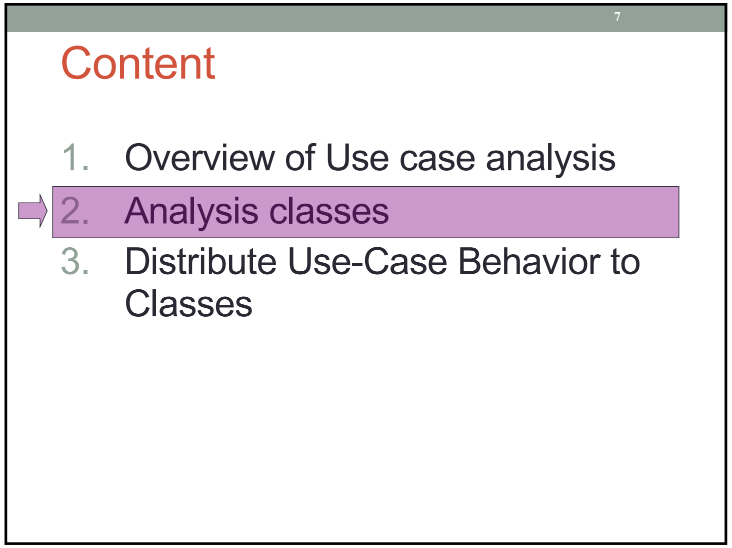


Supplement the Use-Case Specification

- The system displays a list of courses.

- The system retrieves and displays a list of current courses from the course catalog legacy database.

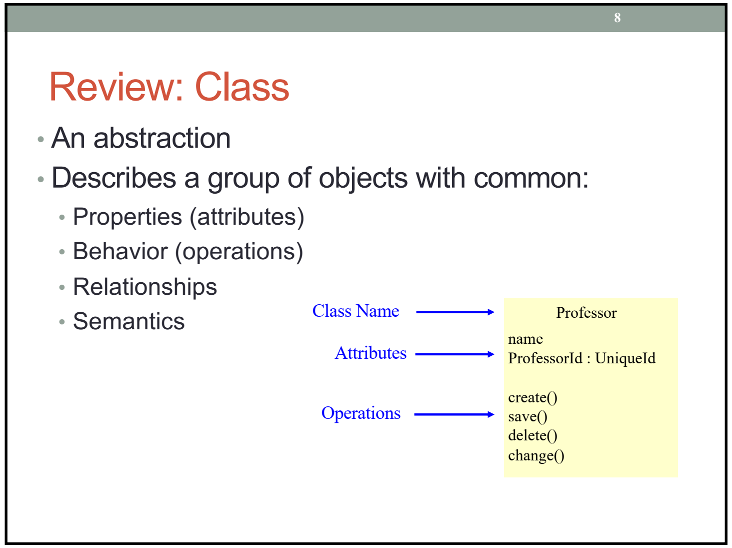
6



Content

1. Overview of Use case analysis
2. Analysis classes
3. Distribute Use-Case Behavior to Classes

7



Review: Class

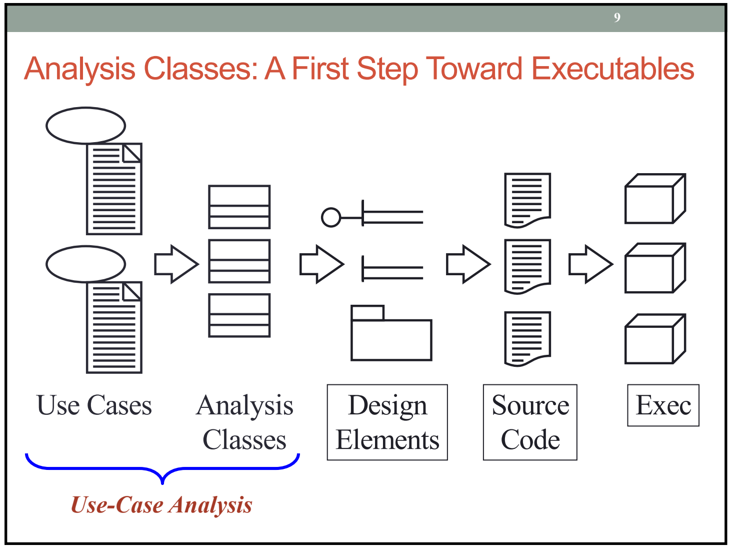
- An abstraction
- Describes a group of objects with common:
  - Properties (attributes)
  - Behavior (operations)
  - Relationships
  - Semantics

Class Name → Professor

Attributes → name
ProfessorId : UniqueId

Operations → create()
save()
delete()
change()

8



Analysis Classes: A First Step Toward Executables

Use Cases

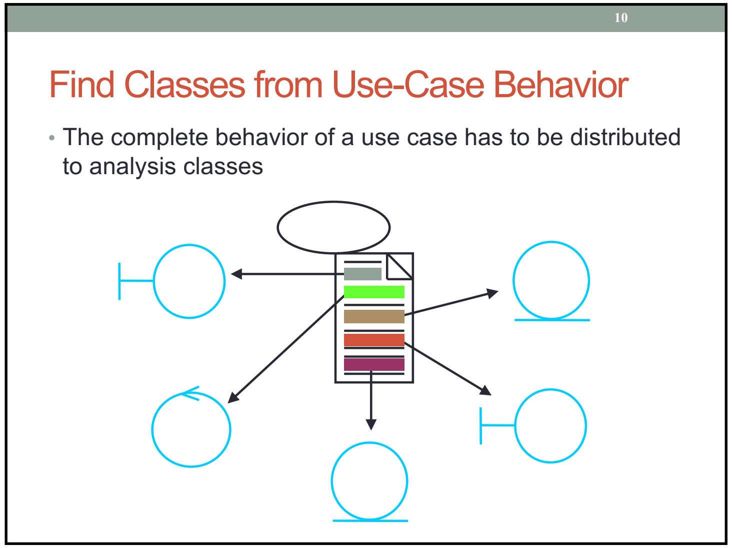
Analysis Classes

Design Elements

Source Code

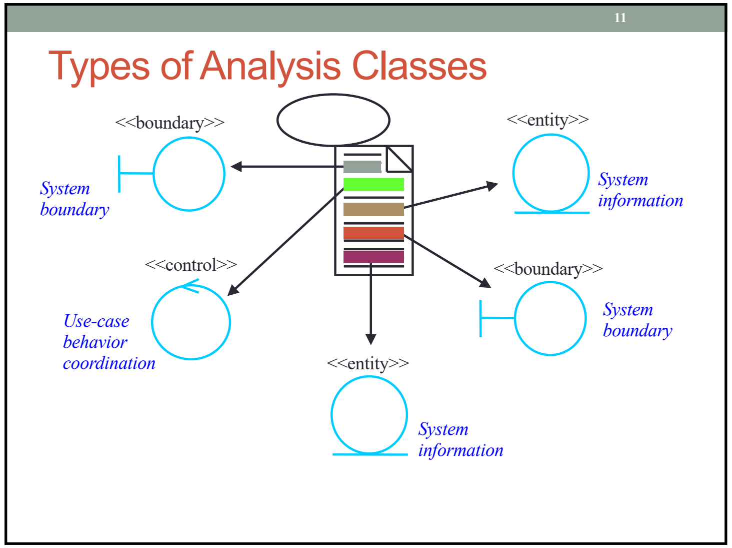
Exec

Use-Case Analysis



Find Classes from Use-Case Behavior

- The complete behavior of a use case has to be distributed to analysis classes



Types of Analysis Classes

<<boundary>>

System boundary

<<entity>>

System information

<<control>>

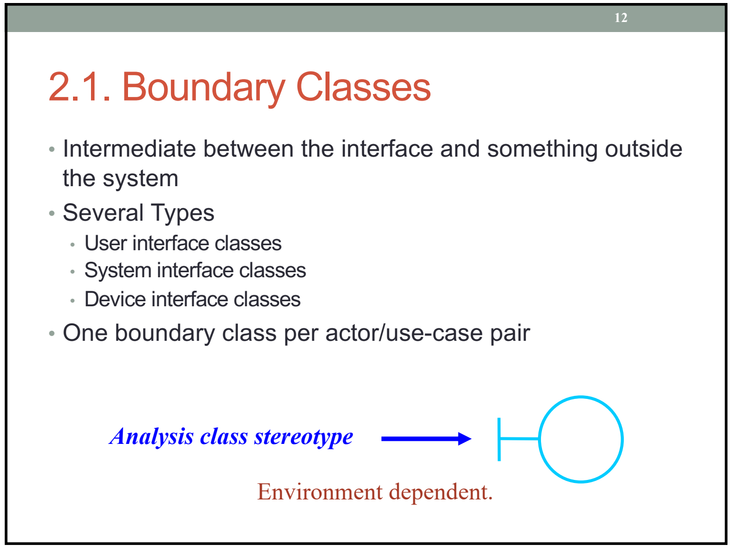
Use-case behavior coordination

<<boundary>>

System boundary

<<entity>>

System information

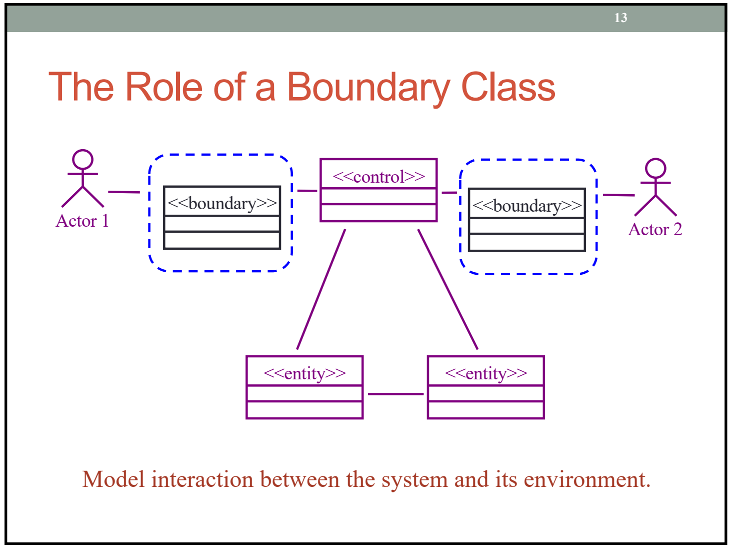


2.1. Boundary Classes

- Intermediate between the interface and something outside the system
- Several Types
  - User interface classes
  - System interface classes
  - Device interface classes
- One boundary class per actor/use-case pair

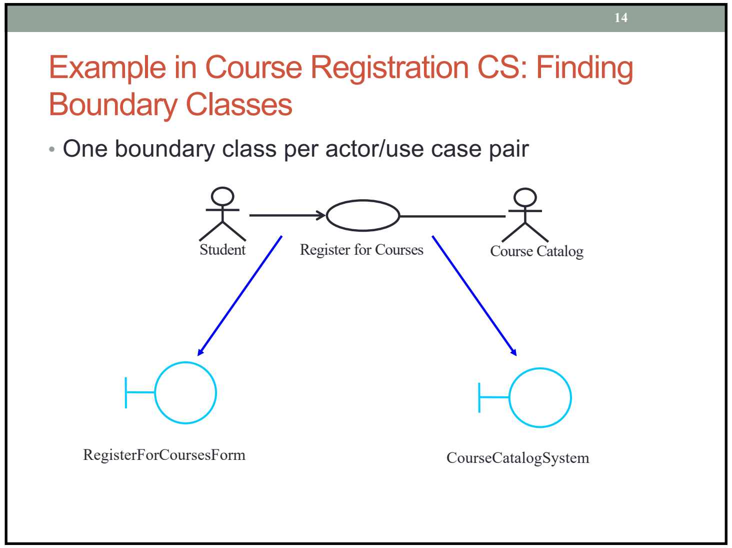
Analysis class stereotype

Environment dependent.



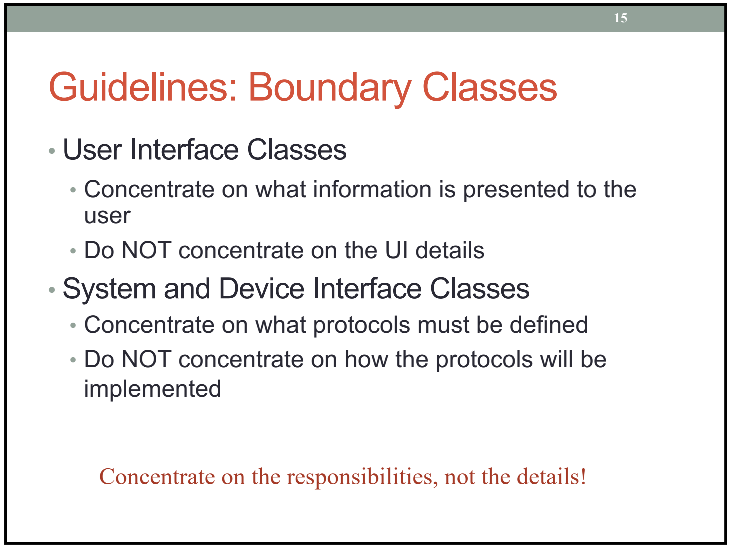
The Role of a Boundary Class

Model interaction between the system and its environment.



Example in Course Registration CS: Finding Boundary Classes

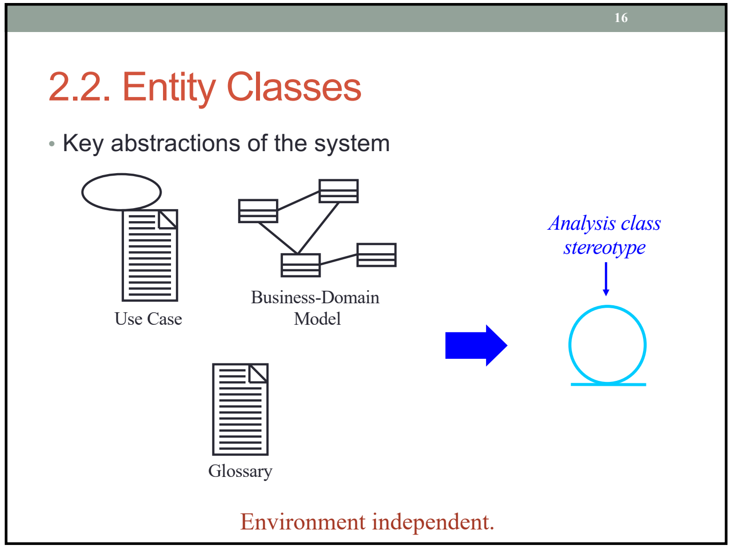
- One boundary class per actor/use case pair



Guidelines: Boundary Classes

- User Interface Classes
  - Concentrate on what information is presented to the user
  - Do NOT concentrate on the UI details
- System and Device Interface Classes
  - Concentrate on what protocols must be defined
  - Do NOT concentrate on how the protocols will be implemented

Concentrate on the responsibilities, not the details!



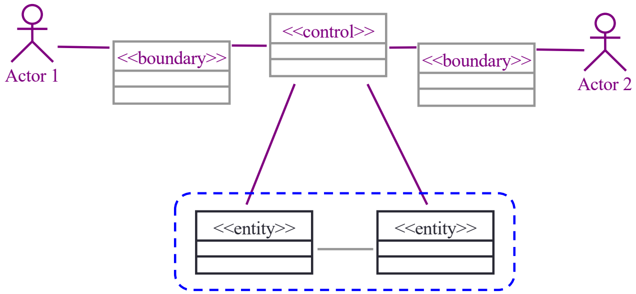
2.2. Entity Classes

- Key abstractions of the system

Environment independent.

17

The Role of Entity Classes



Store and manage information in the system.

17

18

Guidelines: Entity Classes

- Use use-case flow of events as input
- Key abstractions of the use case
- Traditional, filtering nouns approach
  - Underline noun clauses in the use-case flow of events
  - Remove redundant candidates
  - Remove vague candidates
  - Remove actors (out of scope)
  - Remove implementation constructs
  - Remove attributes (save for later)
  - Remove operations

18

19

Example in Course Registration CS: Finding Entity Classes

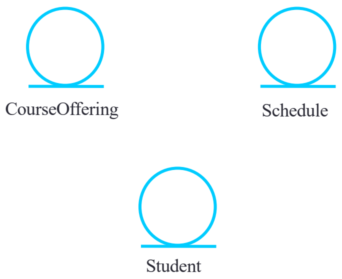
- For “Register For Course” use case, there are some candidate entity classes:

19

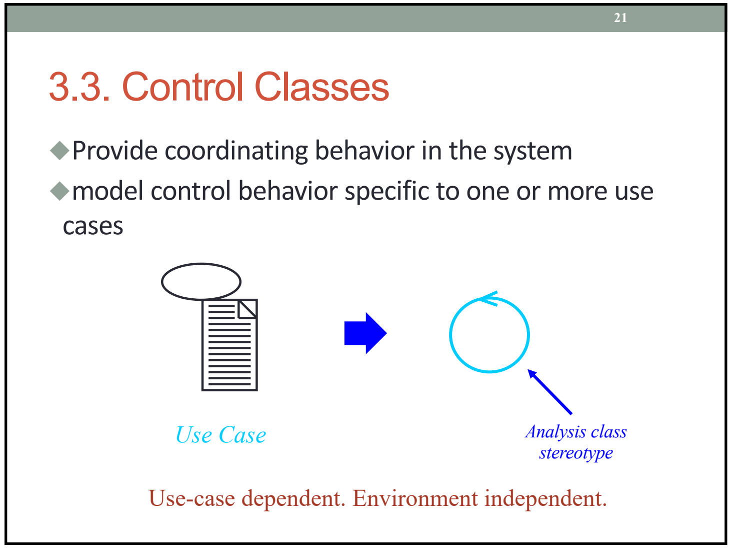
20

Example in Course Registration CS: Finding Entity Classes

- For “Register For Course” use case, there are some candidate entity classes:



20

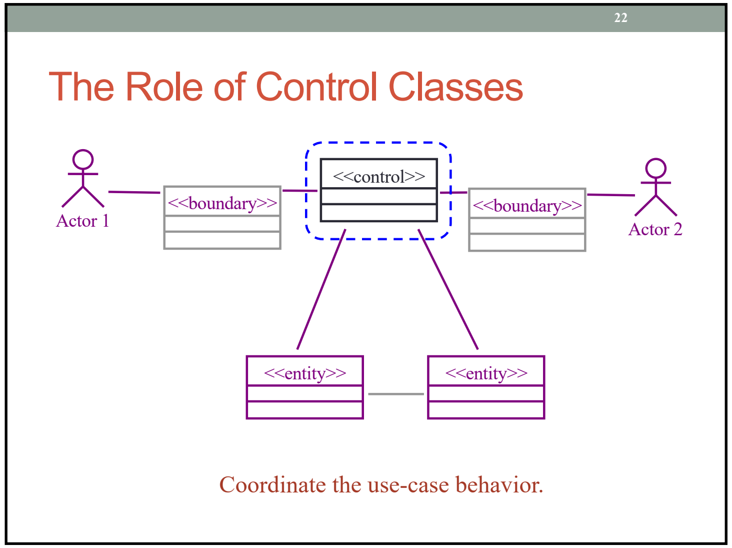


3.3. Control Classes

- Provide coordinating behavior in the system
- model control behavior specific to one or more use cases

Use-case dependent. Environment independent.

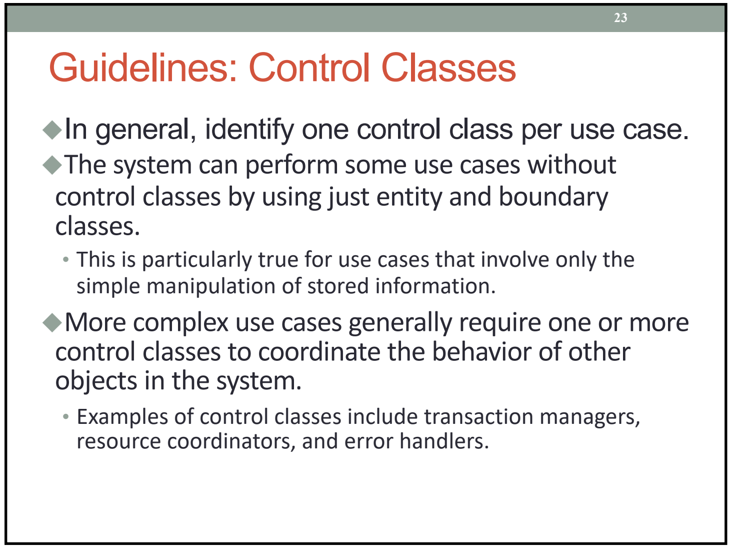
21



The Role of Control Classes

Coordinate the use-case behavior.

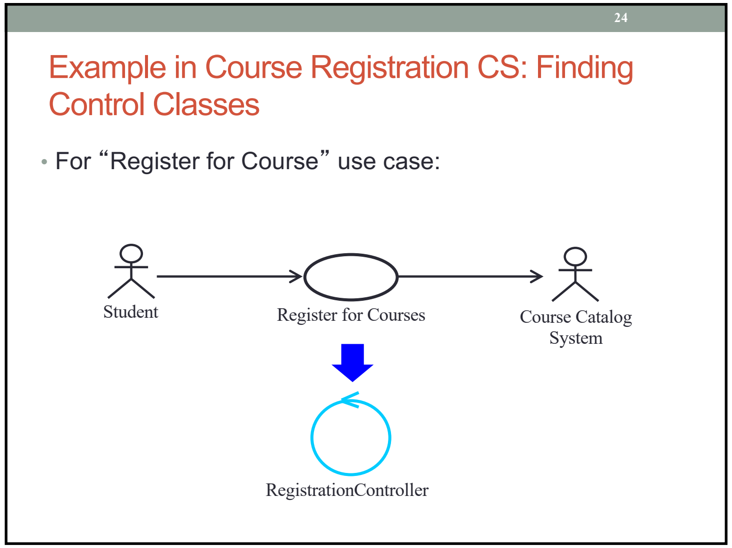
22



Guidelines: Control Classes

- In general, identify one control class per use case.
- The system can perform some use cases without control classes by using just entity and boundary classes.
  - This is particularly true for use cases that involve only the simple manipulation of stored information.
- More complex use cases generally require one or more control classes to coordinate the behavior of other objects in the system.
  - Examples of control classes include transaction managers, resource coordinators, and error handlers.

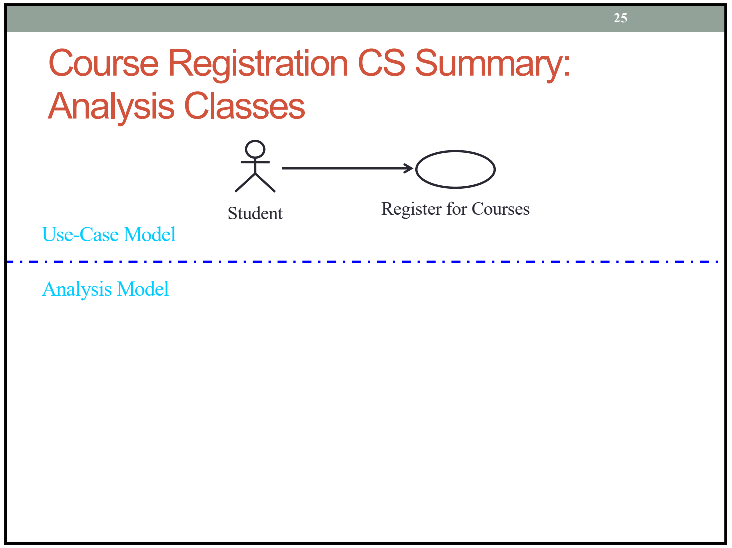
23



Example in Course Registration CS: Finding Control Classes

- For "Register for Course" use case:

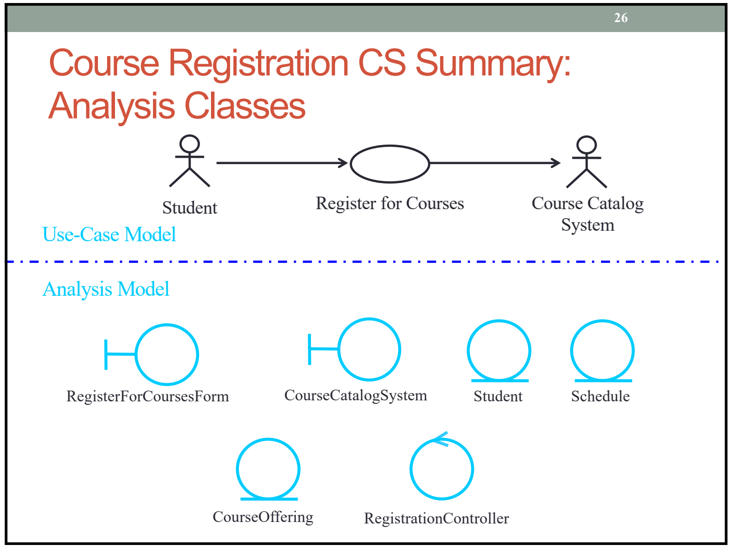
24



25

Course Registration CS Summary: Analysis Classes

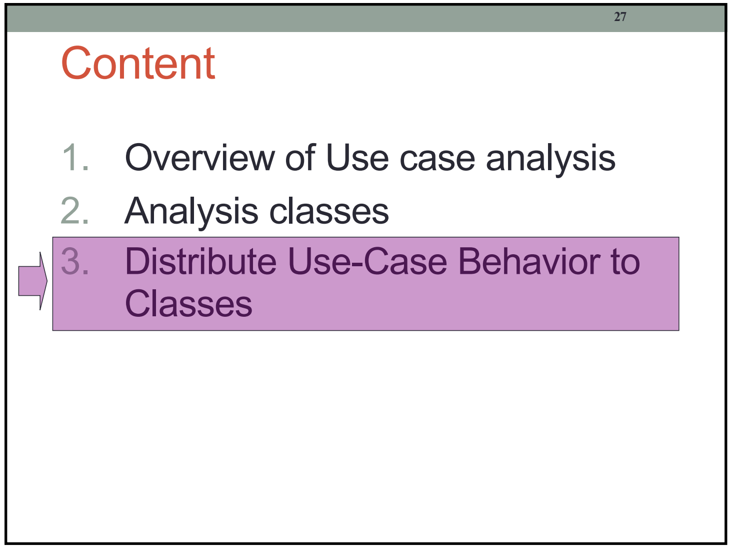
25



26

Course Registration CS Summary: Analysis Classes

26

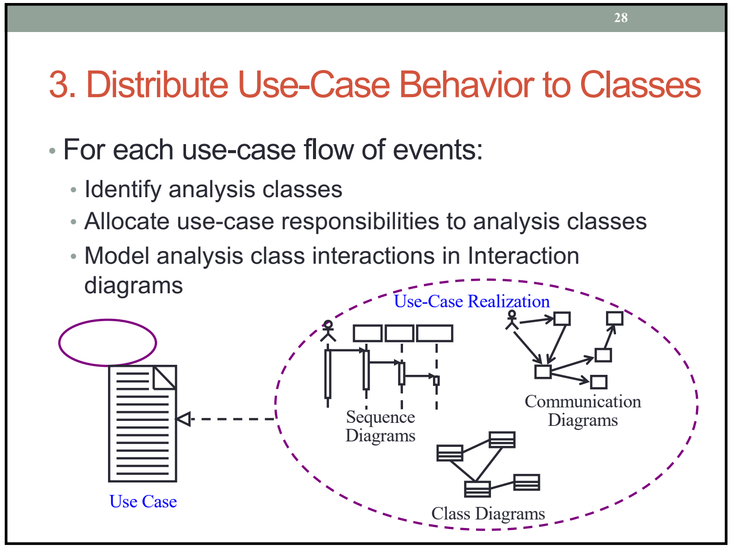


27

Content

1. Overview of Use case analysis
2. Analysis classes
3. Distribute Use-Case Behavior to Classes

27



28

3. Distribute Use-Case Behavior to Classes

- For each use-case flow of events:
  - Identify analysis classes
  - Allocate use-case responsibilities to analysis classes
  - Model analysis class interactions in Interaction diagrams

28

3.1. Allocating Responsibilities to Classes

- Use analysis class stereotypes as a guide
  - Boundary Classes
    - Behavior that involves communication with an actor
  - Entity Classes
    - Behavior that involves the data encapsulated within the abstraction
  - Control Classes
    - Behavior specific to a use case or part of a very important flow of events

3.1. Allocating Responsibilities to Classes (2)

- Who has the data needed to perform the responsibility?
  - If one class has the data, put the responsibility with the data
  - If multiple classes have the data:
    - Put the responsibility with one class and add a relationship to the other
    - Create a new class, put the responsibility in the new class, and add relationships to classes needed to perform the responsibility
    - Put the responsibility in the control class, and add relationships to classes needed to perform the responsibility

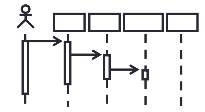
3.2. Interaction Diagrams

- Generic term that applies to several diagrams that emphasize object interactions
  - Sequence Diagram
  - Communication Diagram
- Specialized Variants
  - Timing Diagram
  - Interaction Overview Diagram

3.2. Interaction Diagrams (2)

◆Sequence Diagram

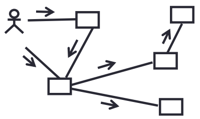
- Time oriented view of object interaction



Sequence Diagrams

◆Communication Diagram

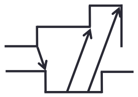
- Structural view of messaging objects



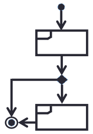
Communication Diagrams

3.2. Interaction Diagrams (3)

- Timing Diagram
  - Time constraint view of messages involved in an interaction
- Interaction Overview Diagram
  - High level view of interaction sets combined into logic sequence



Timing Diagrams

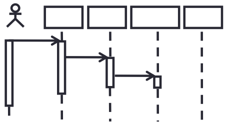


Interaction Overview Diagrams

33

3.2.1. Sequence Diagram

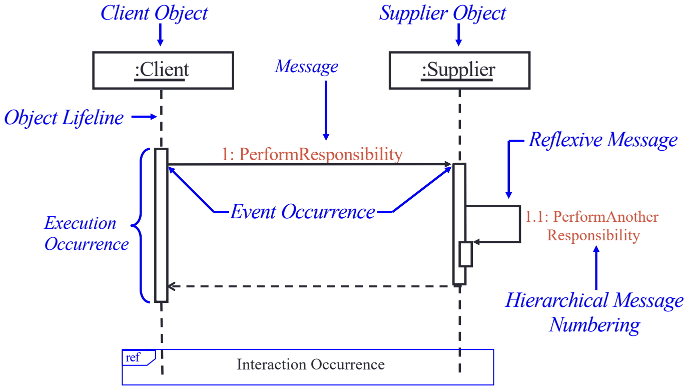
- A sequence diagram is an interaction diagram that emphasizes the time ordering of messages.
- The diagram shows:
  - The objects participating in the interaction.
  - The sequence of messages exchanged.



Sequence Diagram

34

The Anatomy of Sequence Diagrams

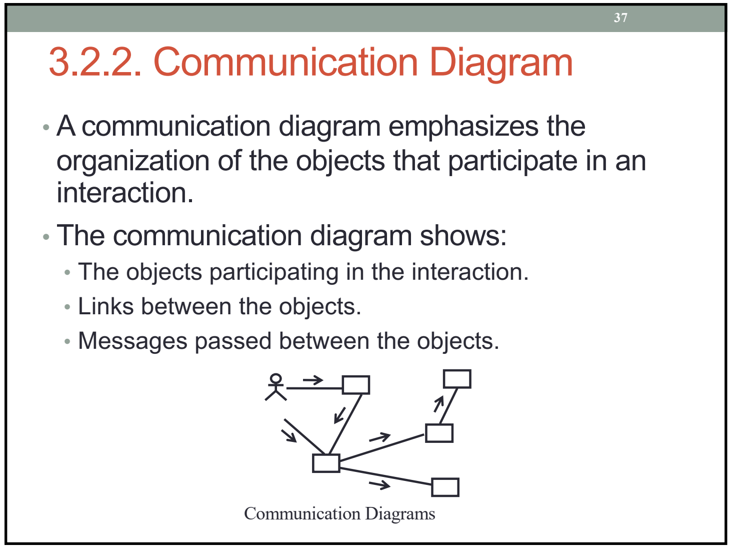


35

Exercise: Course Registration CS

- Draw a sequence diagram for “Register for course” use case

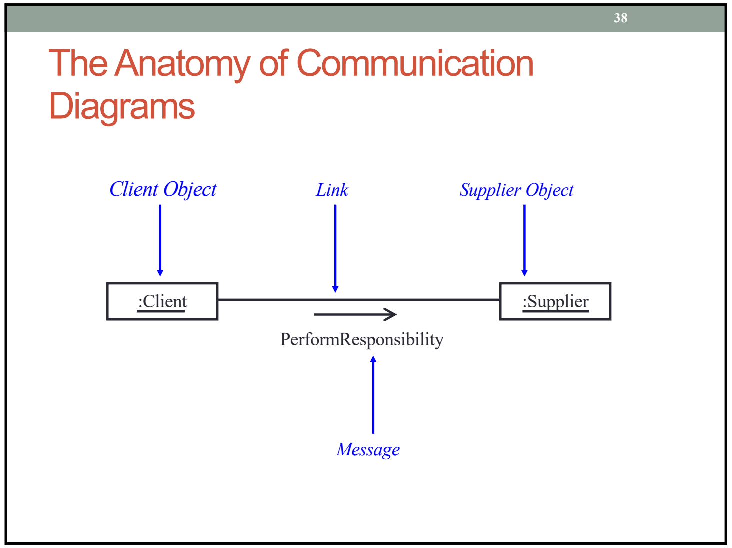
36



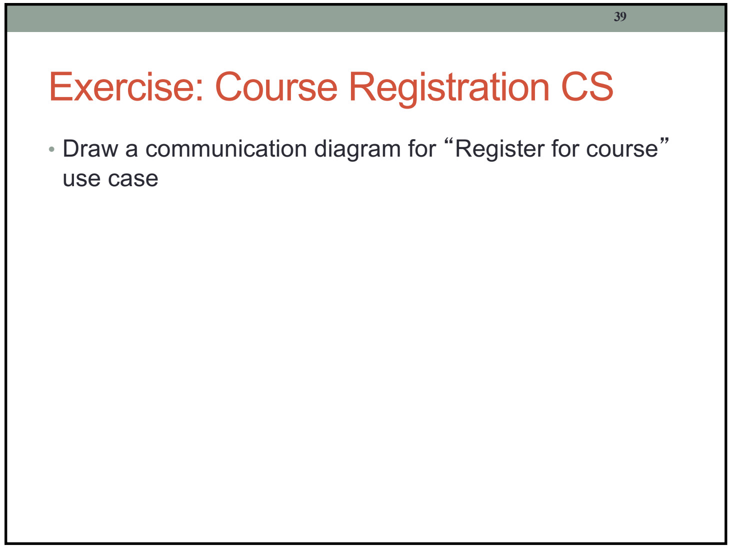
3.2.2. Communication Diagram

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

Communication Diagrams

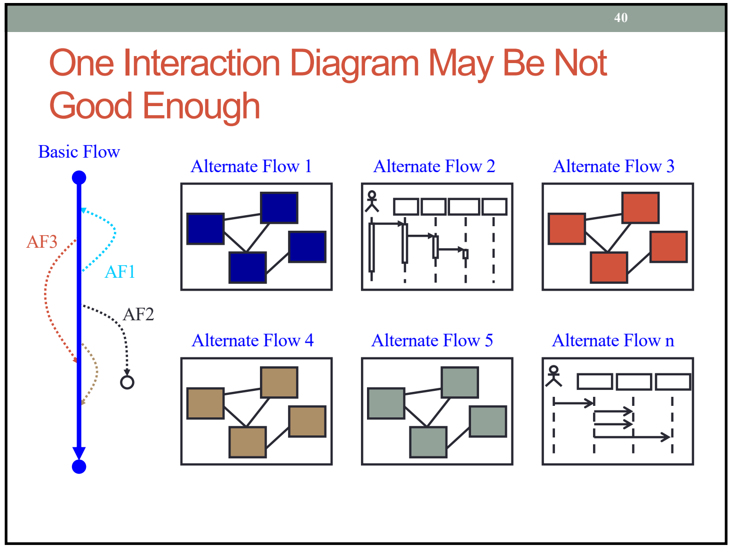


The Anatomy of Communication Diagrams



Exercise: Course Registration CS

- Draw a communication diagram for “Register for course” use case



One Interaction Diagram May Be Not Good Enough

3.2.3. Sequence and Communication Diagram Comparison

- Similarities
  - Semantically equivalent
    - Can convert one diagram to the other without losing any information
  - Model the dynamic aspects of a system
  - Model a use-case scenario

3.2.3. Sequence and Communication Diagram Comparison (2)

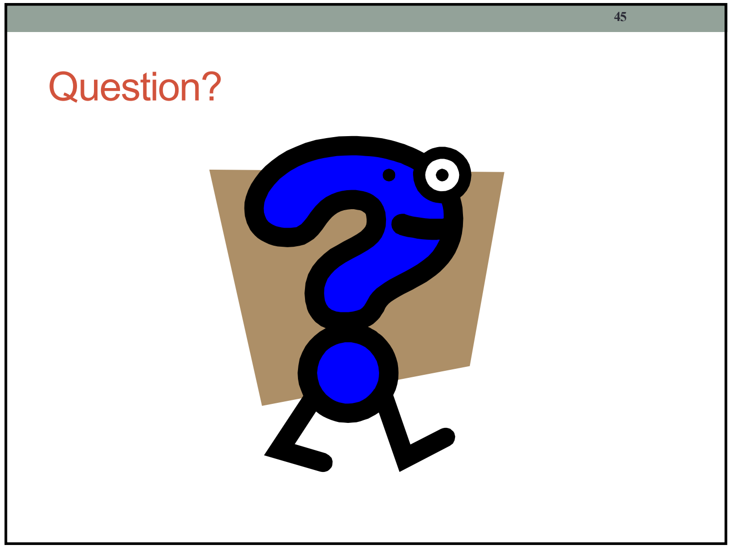
Sequence diagrams	Communication diagrams
Show the explicit sequence of messages	Show relationships in addition to interactions
Show execution occurrence	Better for visualizing patterns of communication
Better for visualizing overall flow	Better for visualizing all of the effects on a given object
Better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

Reviewpoints: Analysis Classes

- Are the classes reasonable?
- Does the name of each class clearly reflect the role it plays?
- Does the class represent a single well-defined abstraction?
- Are all responsibilities functionally coupled?
- Does the class offer the required behavior?
- Are all specific requirements on the class addressed?

Review points: Message Design

- Have all the main and/or sub-flows been handled, including exceptional cases?
- Have all the required objects been found?
- Have all behaviors been unambiguously distributed to the participating objects?
- Have behaviors been distributed to the right objects?
- Where there are several Interaction diagrams, are their relationships clear and consistent?



Question?

45