

### 3. ARCHITECTURAL DESIGN

Nguyen Thi Thu Trang  
trangntt@soict.hust.edu.vn



*Some slides extracted from IBM coursewares*

1

## Content

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

2

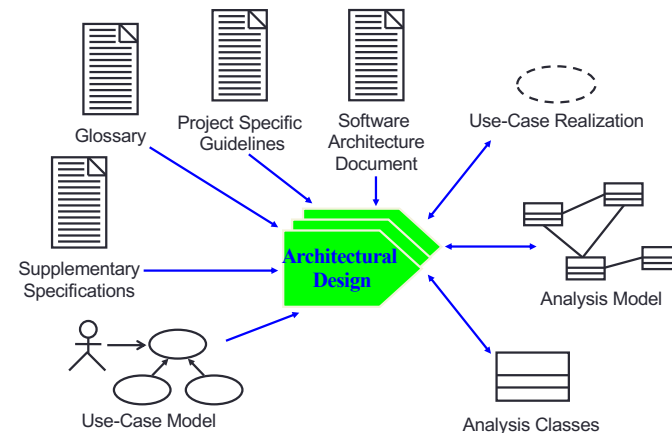
2

### Review: Software Architectural Design process

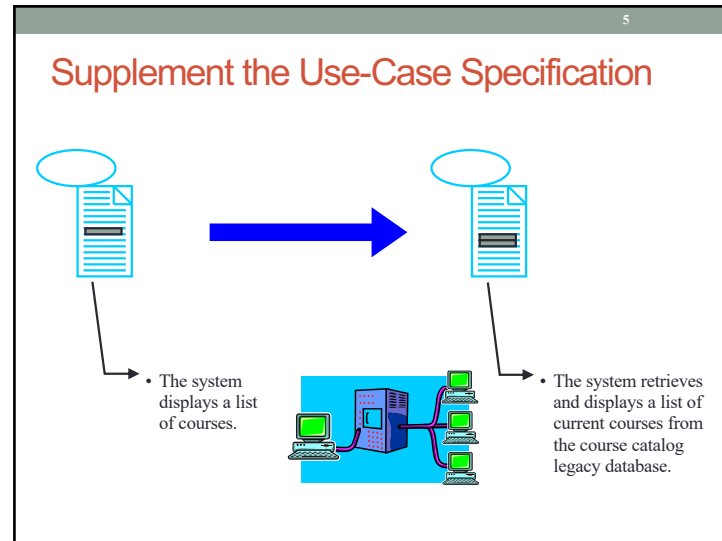
- Purpose: “to provide a design for the software that implements and can be verified against the requirements”
- Software architecture is designed from the software requirements
- Main items
  - a top-level structure of the software and the software components which constructs the software
  - a top-level design for the interfaces external to the software and between the software components
  - a top-level design for the database

3

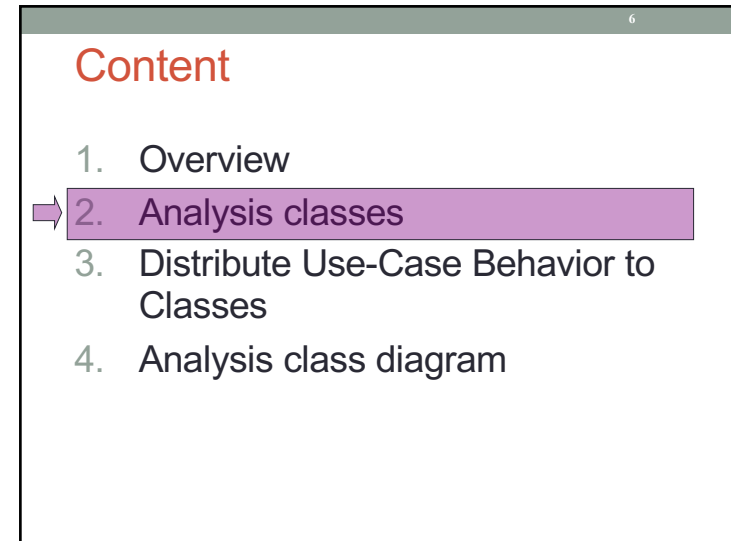
## Architectural Design Overview



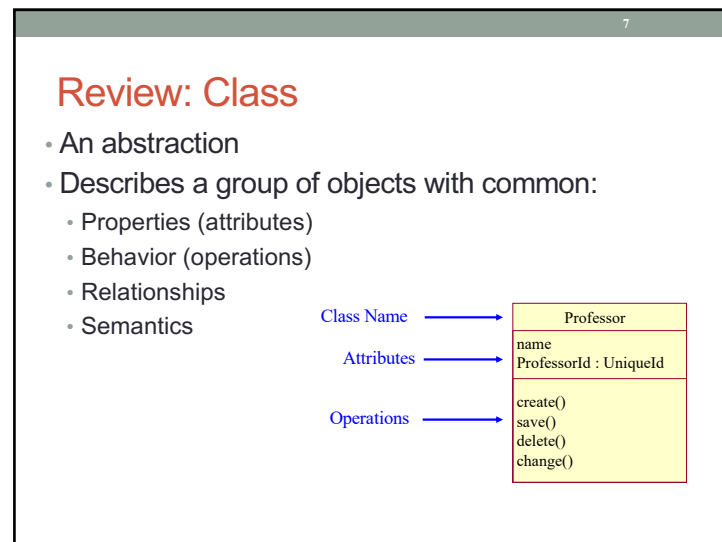
4



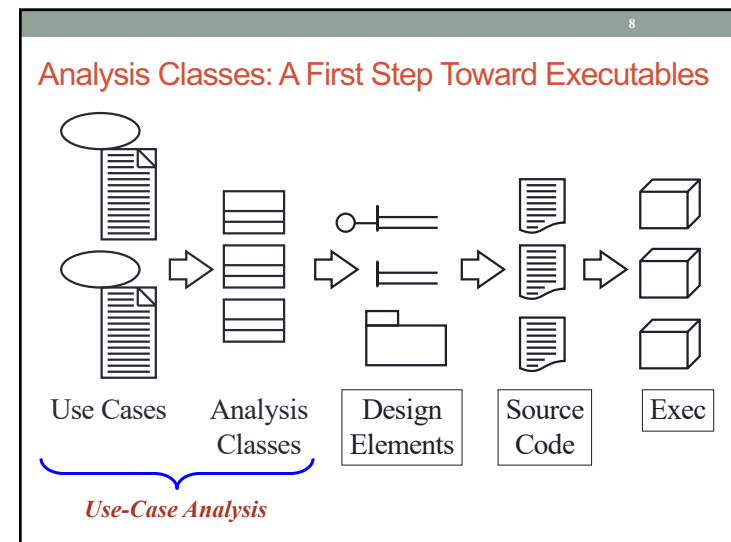
5



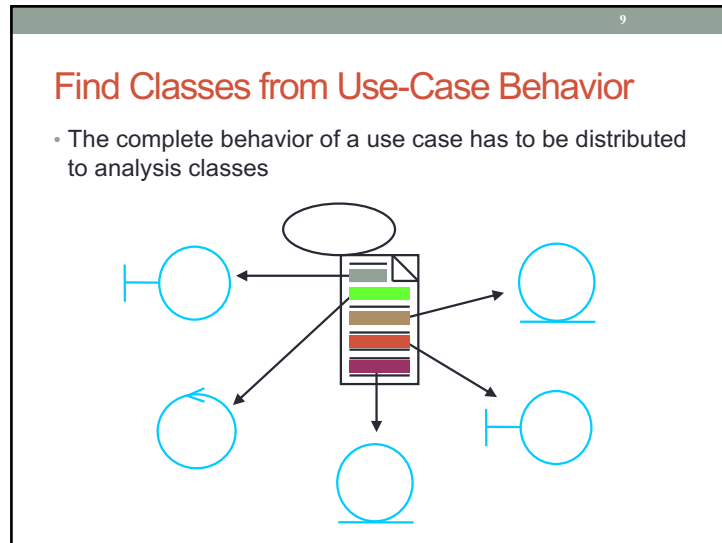
6



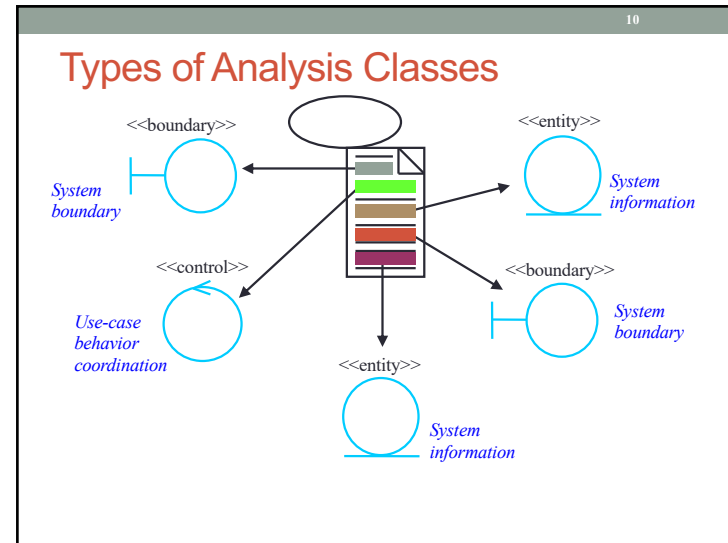
7



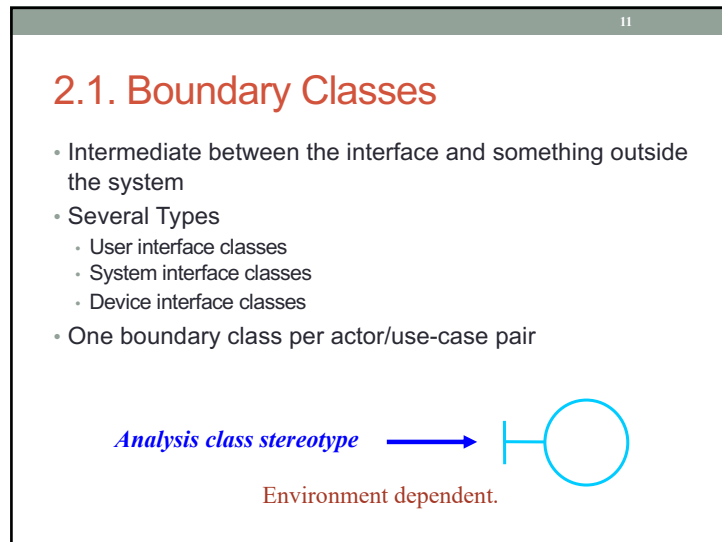
8



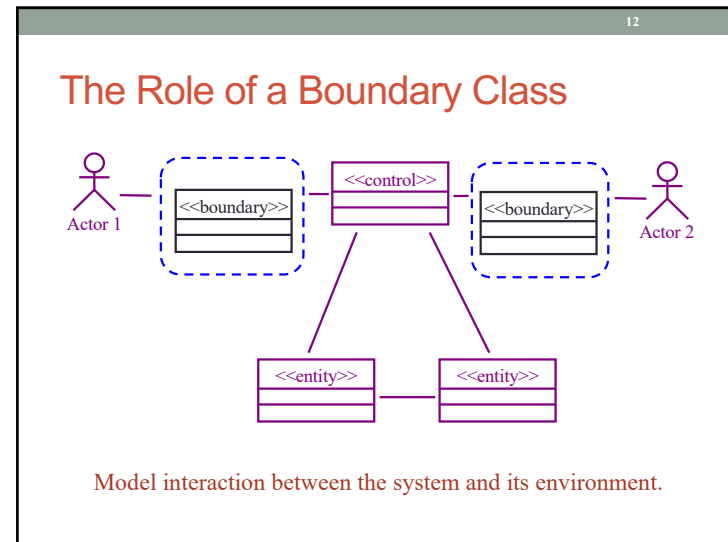
9



10



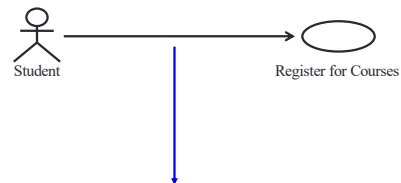
11



12

## Example in Course Registration CS: Finding Boundary Classes

- One boundary class per actor/use case pair



13

## Guidelines: Boundary Classes

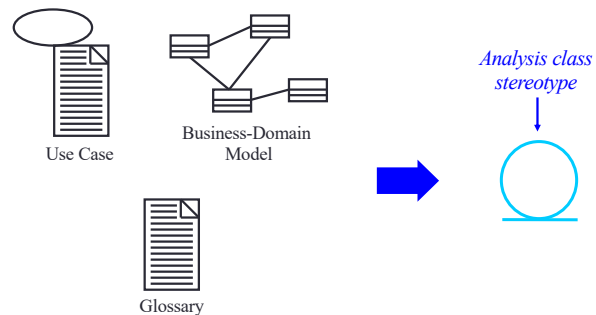
- User Interface Classes
  - Concentrate on what information is presented to the user
  - Do NOT concentrate on the UI details
- System and Device Interface Classes
  - Concentrate on what protocols must be defined
  - Do NOT concentrate on how the protocols will be implemented

Concentrate on the responsibilities, not the details!

14

## 2.2. Entity Classes

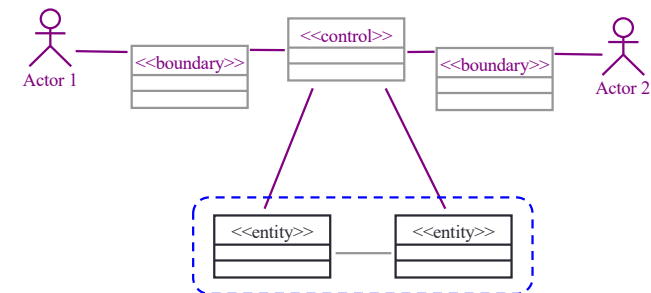
- Key abstractions of the system



Environment independent.

15

## The Role of Entity Classes



Store and manage information in the system.

16

## 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

17

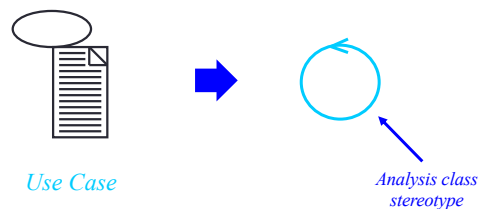
## Example in Course Registration CS: Finding Entity Classes

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

18

## 3.3. Control Classes

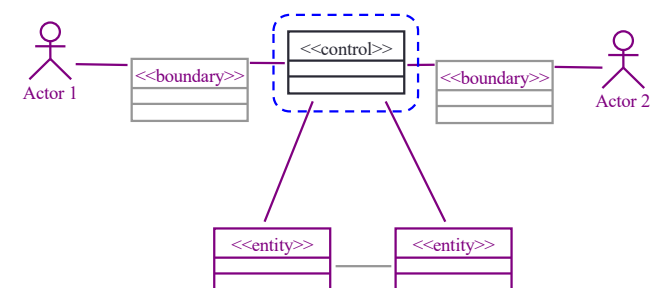
- ◆ Provide coordinating behavior in the system
- ◆ model control behavior specific to one or more use cases



Use-case dependent. Environment independent.

19

## The Role of Control Classes



20

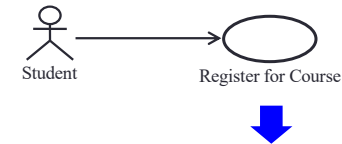
## Guidelines: Control Classes

- ◆ 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.

21

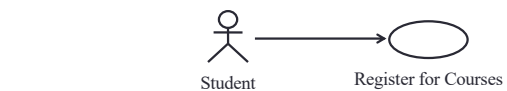
## Example in Course Registration CS: Finding Control Classes

- For “Register for Course” use case:



22

## Course Registration CS Summary: Analysis Classes



Use-Case Model

Analysis Model

23

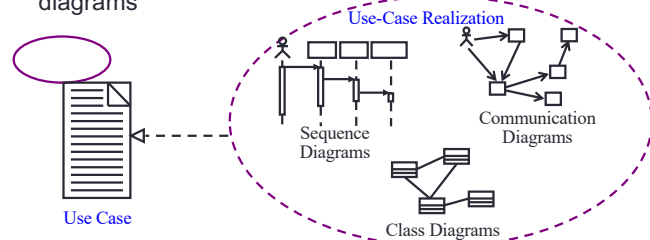
## Content

1. Overview
2. Analysis classes
- ➡ 3. Distribute Use-Case Behavior to Classes
4. Analysis class diagram

24

### 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



25

#### 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

26

#### 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

27

#### 3.2. Interaction Diagrams

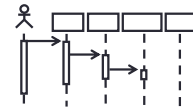
- Generic term that applies to several diagrams that emphasize object interactions
  - Sequence Diagram
  - Communication Diagram
- Specialized Variants
  - Timing Diagram
  - Interaction Overview Diagram

28

## 3.2. Interaction Diagrams (2)

### ◆ Sequence Diagram

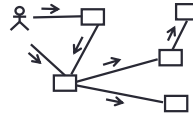
- Time oriented view of object interaction



Sequence Diagrams

### ◆ Communication Diagram

- Structural view of messaging objects



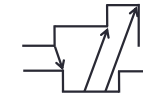
Communication Diagrams

29

## 3.2. Interaction Diagrams (3)

### • Timing Diagram

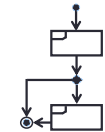
- Time constraint view of messages involved in an interaction



Timing Diagrams

### • Interaction Overview Diagram

- High level view of interaction sets combined into logic sequence

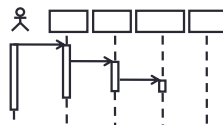


Interaction Overview Diagrams

30

## 3.2.1. Sequence Diagram

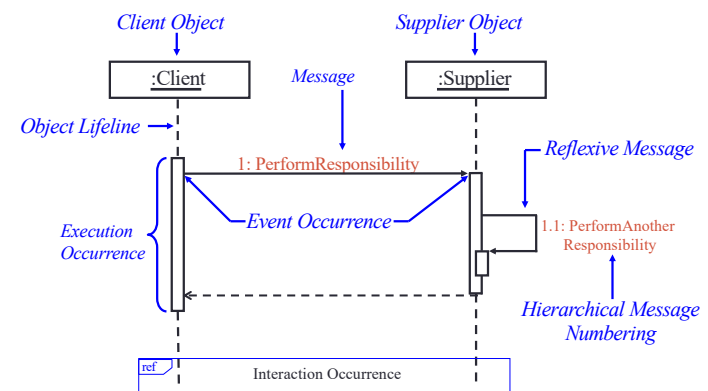
- A sequence diagram is an interaction diagram that emphasizes the time ordering of messages.
- The diagram shows:
  - The objects participating in the interaction.
  - The sequence of messages exchanged.



Sequence Diagram

31

## The Anatomy of Sequence Diagrams



32



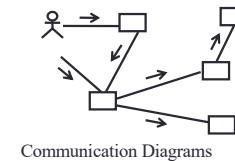
## Exercise: Course Registration CS

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

33

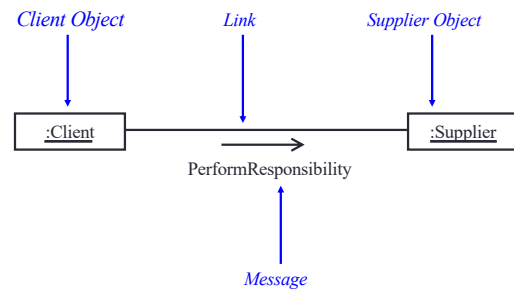
## 3.2.2. Communication Diagram

- 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.



34

## The Anatomy of Communication Diagrams

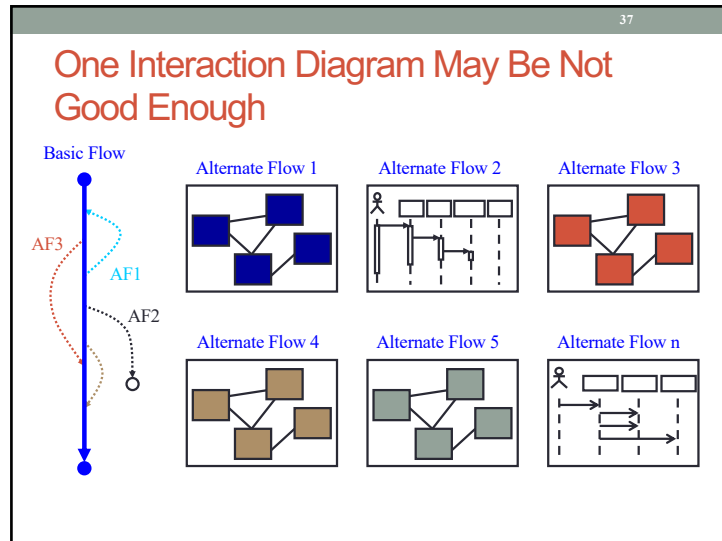


35

## Exercise: Course Registration CS

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

36



37

38

### 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

38

39

### 3.2.3. Sequence and Communication Diagram Comparison (2)

Sequence diagrams	Communication diagrams
<ul style="list-style-type: none"> <li>▪ Show the explicit sequence of messages</li> <li>▪ Show execution occurrence</li> <li>▪ Better for visualizing overall flow</li> <li>▪ Better for real-time specifications and for complex scenarios</li> </ul>	<ul style="list-style-type: none"> <li>▪ Show relationships in addition to interactions</li> <li>▪ Better for visualizing patterns of communication</li> <li>▪ Better for visualizing all of the effects on a given object</li> <li>▪ Easier to use for brainstorming sessions</li> </ul>

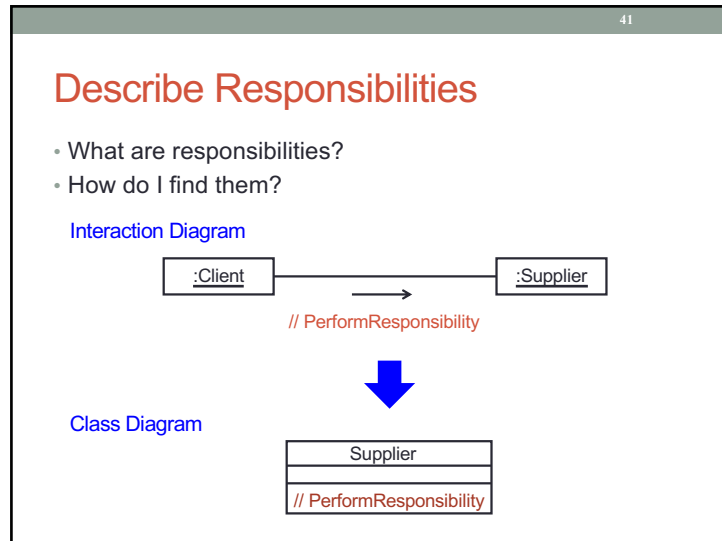
39

40

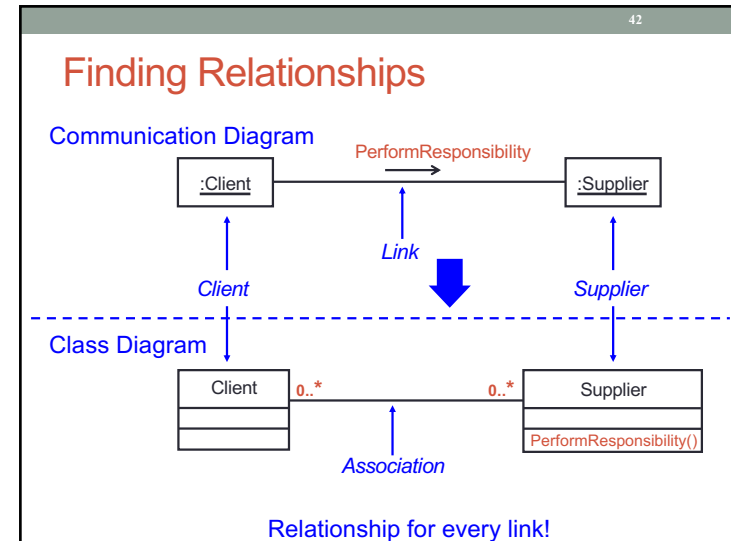
### Content

1. Overview
2. Analysis classes
3. Distribute Use-Case Behavior to Classes
- ➔ 4. Analysis class diagram

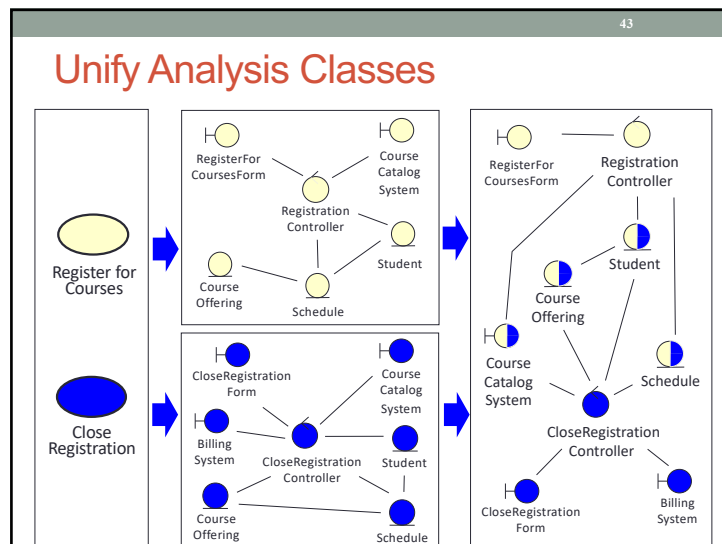
40



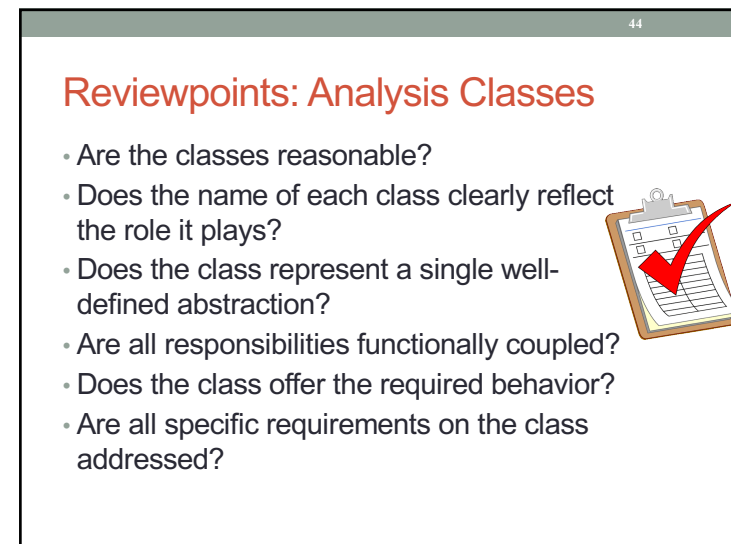
41



42



43



44

## 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?

45

## Question?



46