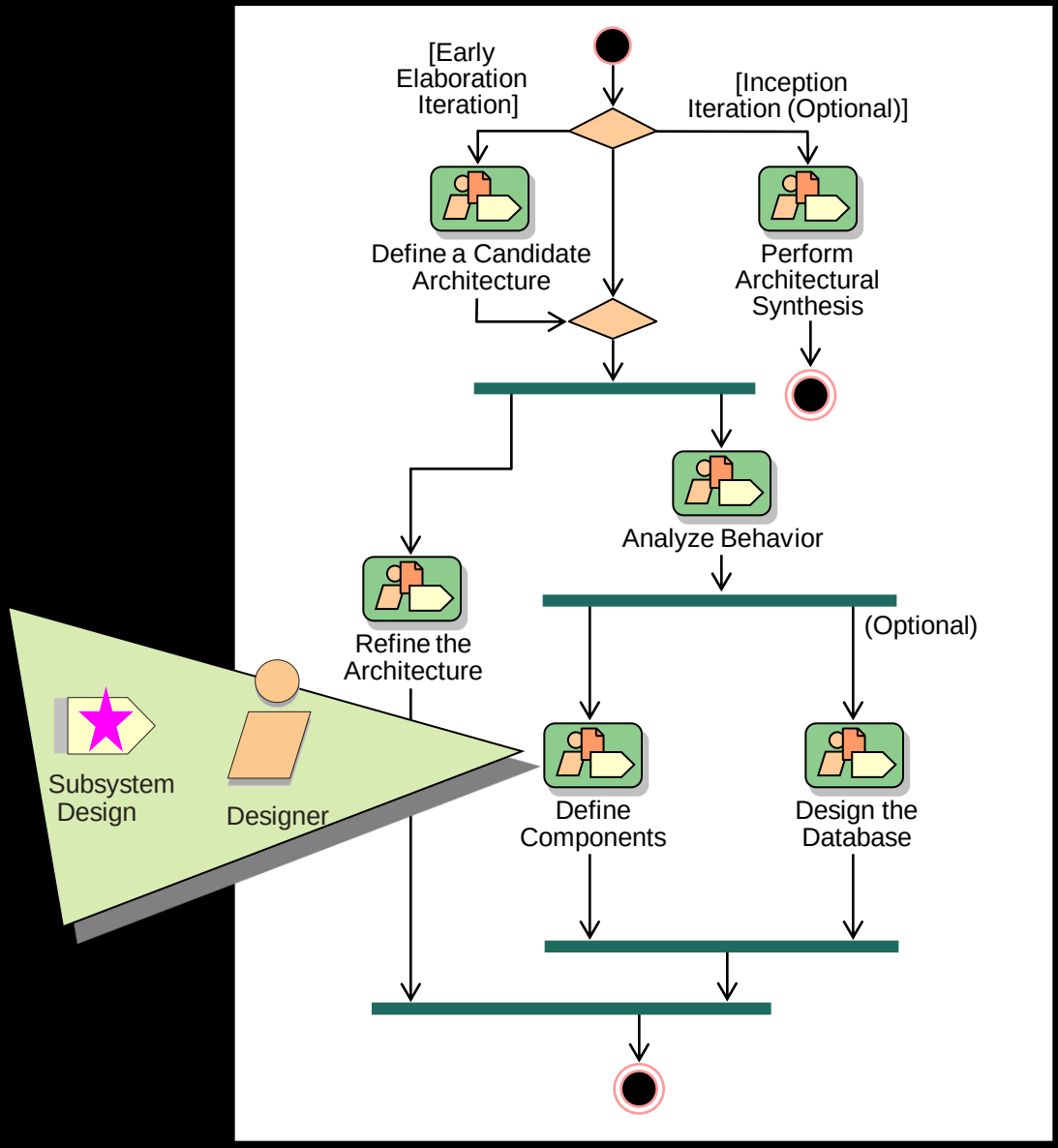
Object-Oriented Analysis and Design

Lecture 12: Subsystem Design

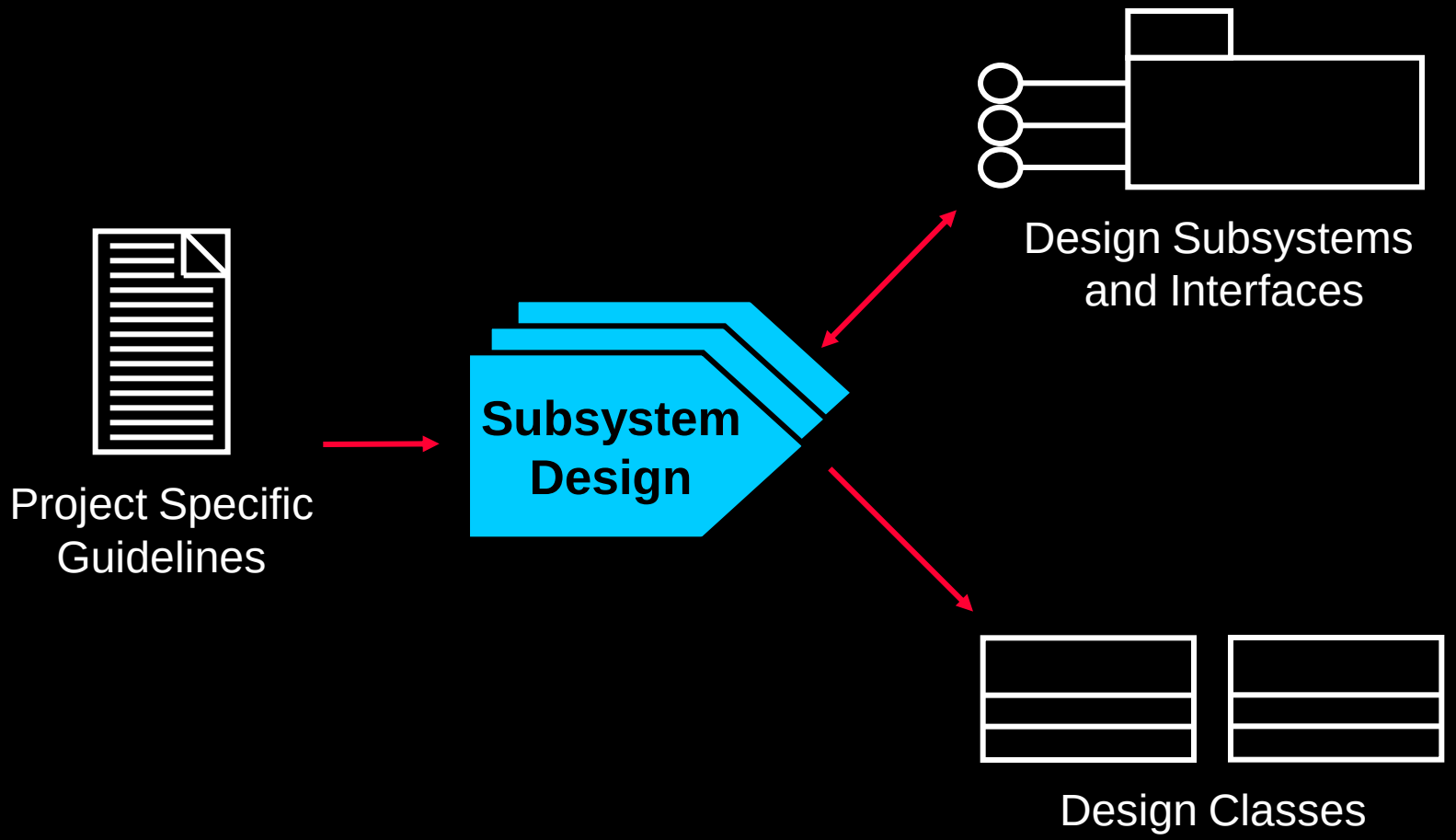
Objectives: Subsystem Design

- Describe the purpose of Subsystem Design and where in the lifecycle it is performed
- Define the behaviors specified in the subsystem's interfaces in terms of collaborations of contained classes
- Document the internal structure of the subsystem
- Determine the dependencies upon elements external to the subsystem

Subsystem Design in Context



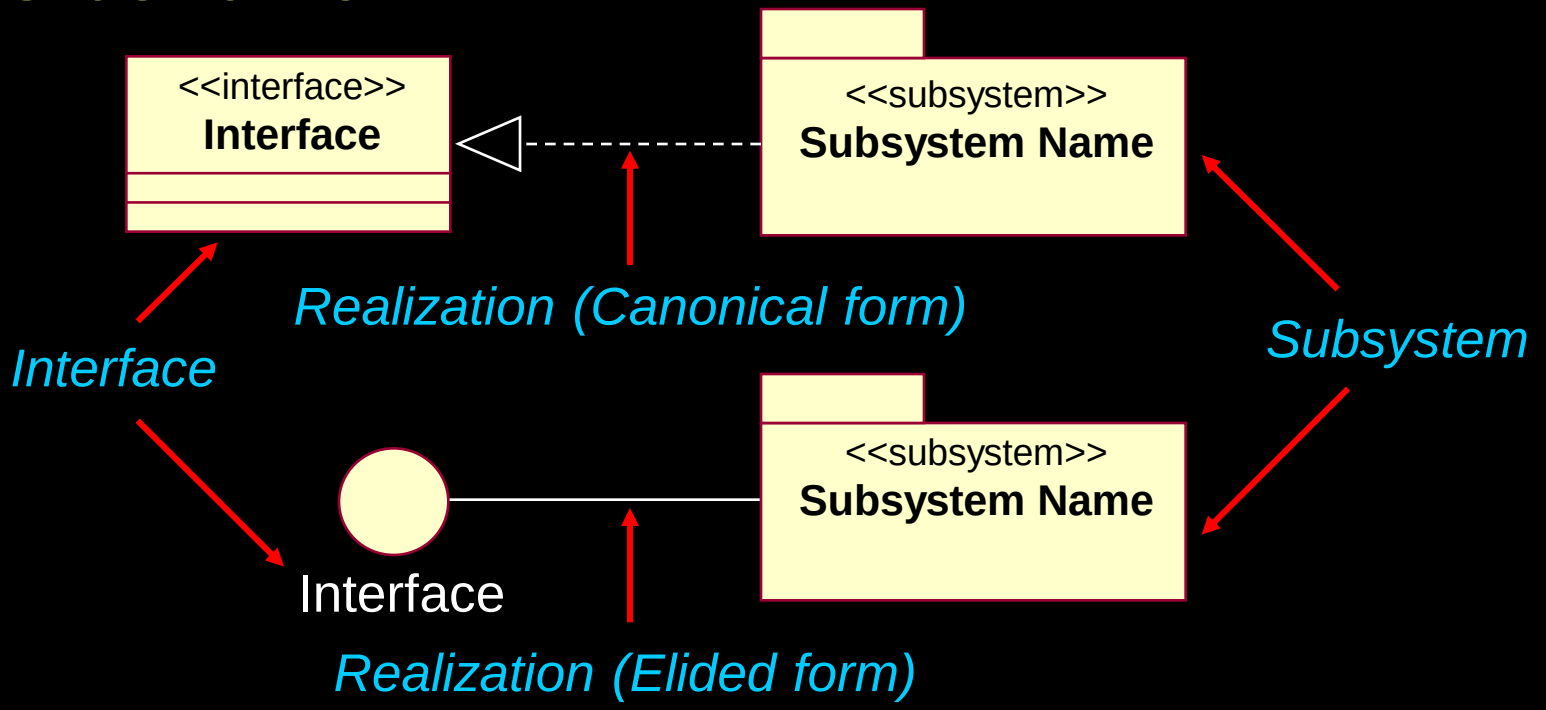
Subsystem Design Overview



Review: Subsystems and Interfaces

A Subsystem:

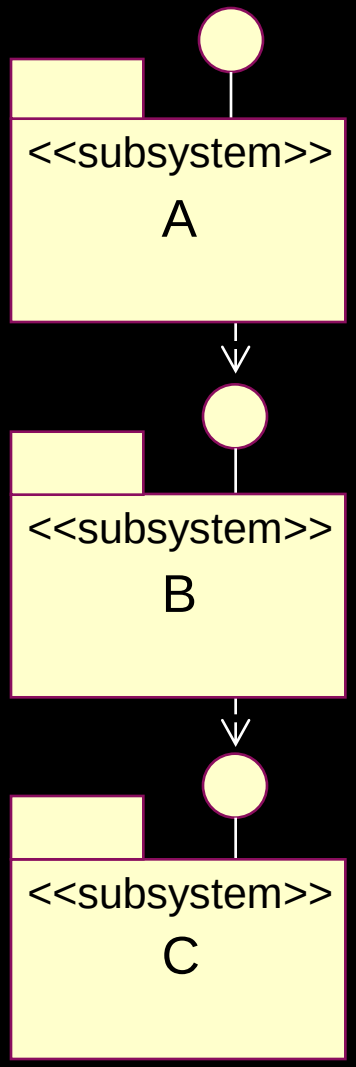
- Is a “cross between” a package and a class
- Realizes one or more interfaces that define its behavior



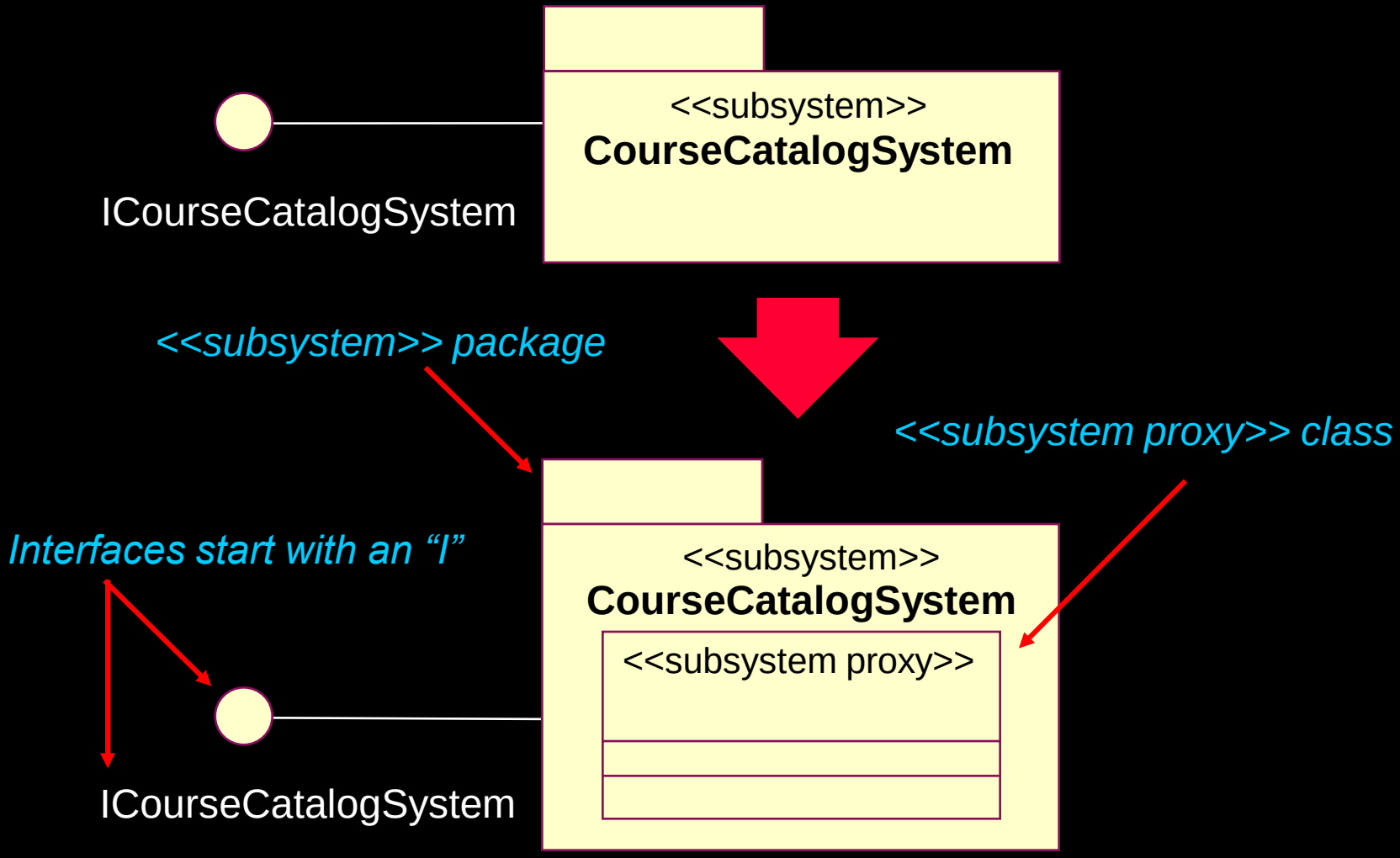
Subsystem Guidelines

- Goals
  - Loose coupling
  - Portability, plug-and-play compatibility
  - Insulation from change
  - Independent evolution
- Strong Suggestions
  - Do not expose details, only interfaces
  - Depend only on other interfaces

Key is abstraction and encapsulation



Review: Modeling Convention for Subsystems and Interfaces



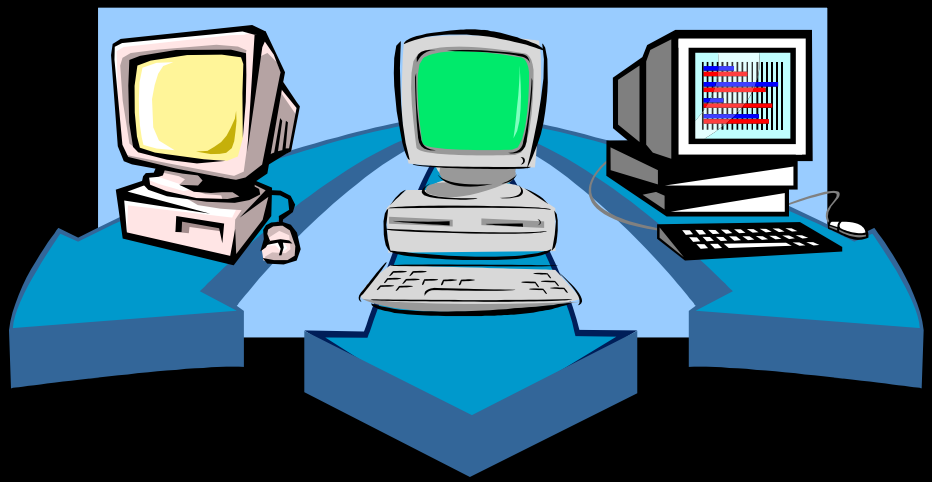
Interfaces are EXTERNAL to the subsystem.

Subsystem Design Steps

- Distribute subsystem behavior to subsystem elements
- Document subsystem elements
- Describe subsystem dependencies
- Checkpoints

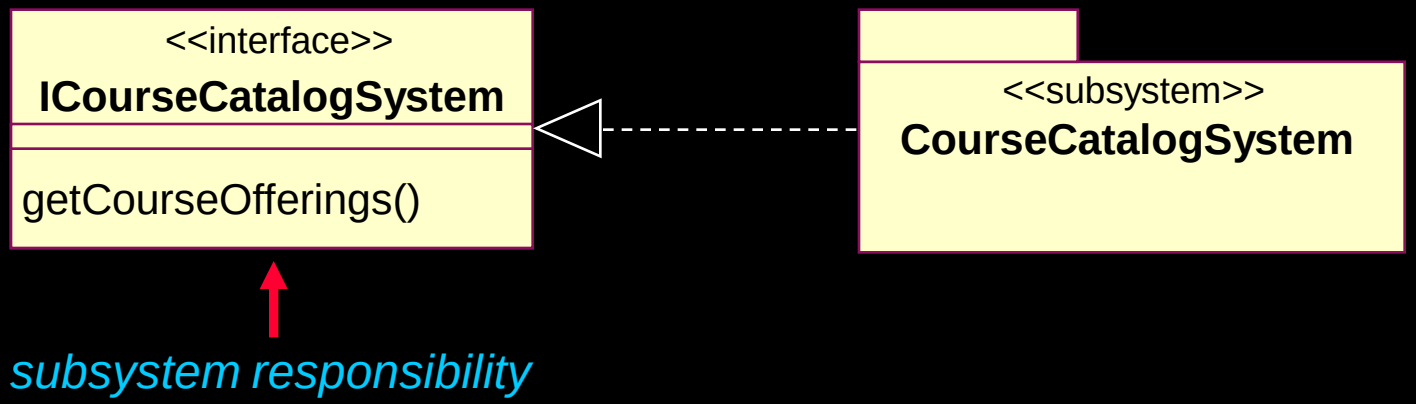
Subsystem Design Steps

- ★ Distribute subsystem behavior to subsystem elements
  - Document subsystem elements
  - Describe subsystem dependencies
  - Checkpoints



Subsystem Responsibilities

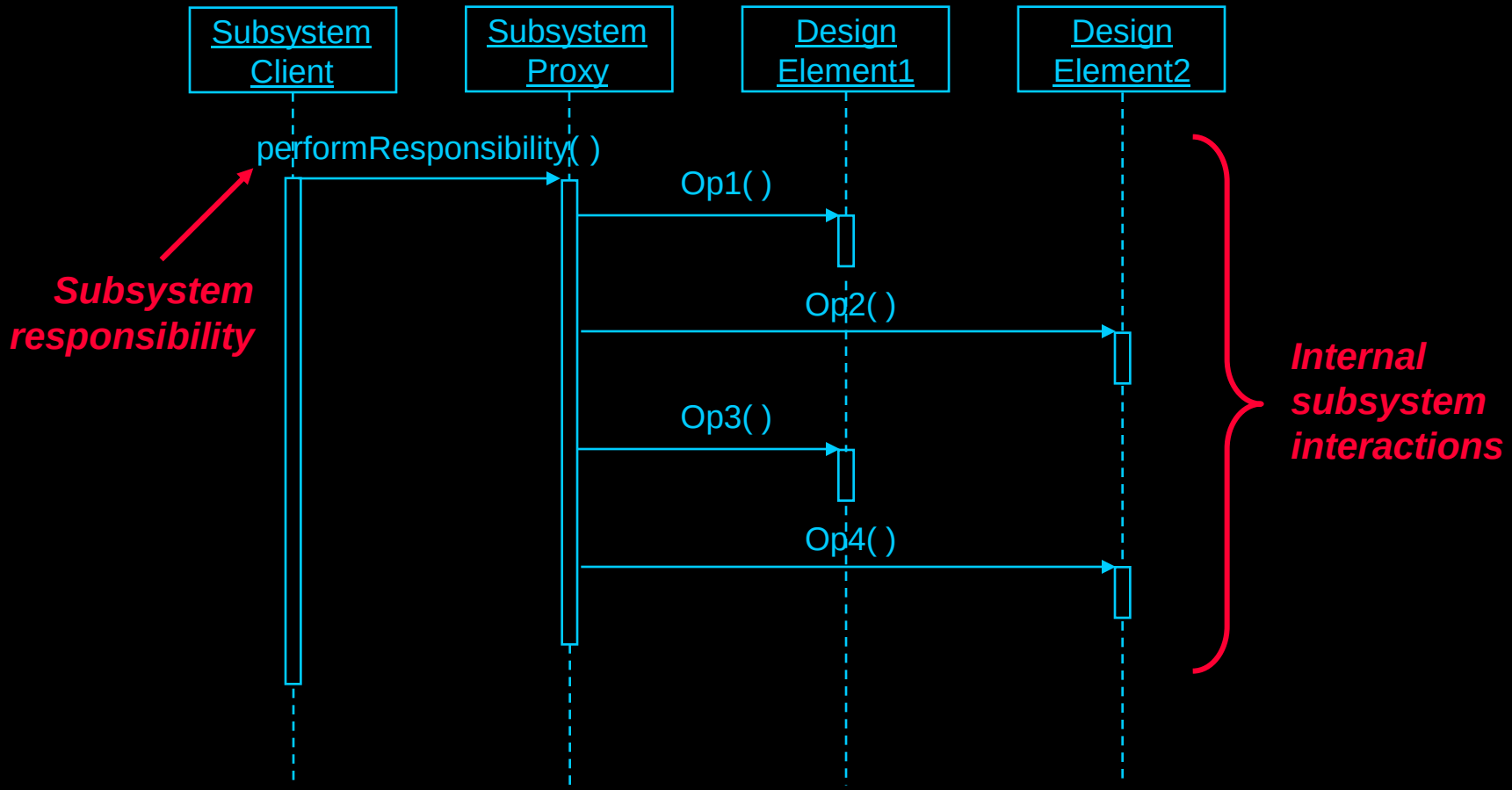
- Subsystem responsibilities defined by interface operations
  - Model interface realizations
- Interface operations may be realized by
  - Internal class operations
  - Internal subsystem operations



Distributing Subsystem Responsibilities

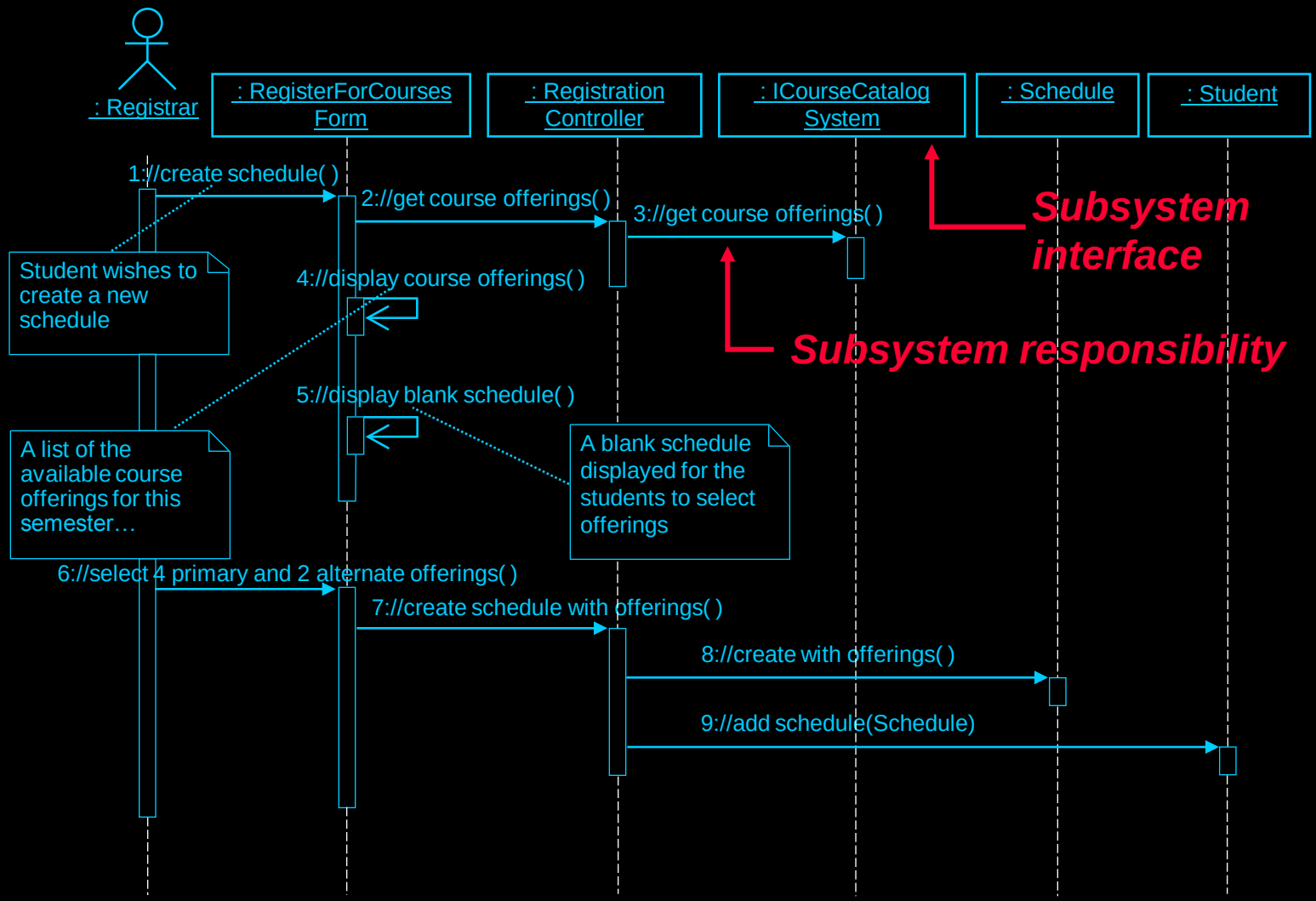
- Identify new, or reuse existing, design elements (for example, classes and/or subsystems)
- Allocate subsystem responsibilities to design elements
- Incorporate applicable mechanisms (for example, persistence, distribution)
- Document design element collaborations in “interface realizations”
  - One or more interaction diagrams per interface operation
  - Class diagram(s) containing the required design element relationships
- Revisit “*Identify Design Elements*”
  - Adjust subsystem boundaries and dependencies, as needed

Modeling Convention: Subsystem Interaction Diagrams



Subsystem interface not shown

Example: CourseCatalogSystem Subsystem in Context



Legacy RDBMS Database Access

Incorporating the Architectural Mechanisms: Persistency

- Analysis-Class-to-Architectural-Mechanism Map from Use-Case Analysis

Analysis Class	Analysis Mechanism(s)
Student	Persistency, Security
Schedule	Persistency, Security
CourseOffering	*Persistency, Legacy Interface*
Course	*Persistency, Legacy Interface*
RegistrationController	Distribution

OODBMS Persistency

RDBMS Persistency

OODBMS Persistency was discussed in Use-Case Design

Review: Incorporating JDBC: Steps

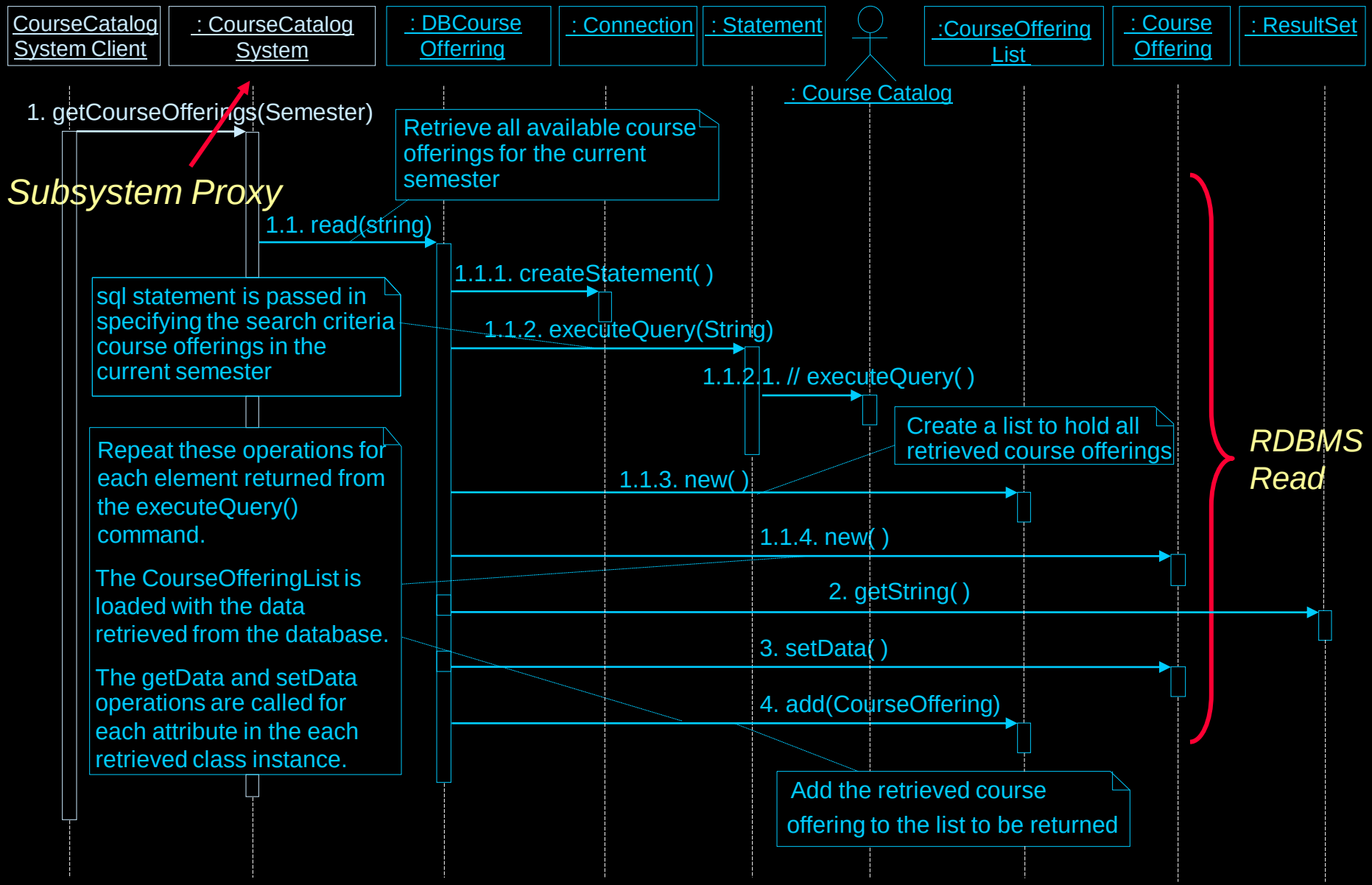
- Provide access to the class libraries needed to implement JDBC
  - √ *Provide java.sql package*
- Create the necessary DBClasses
  - One DBClass per persistent class
  - Course Offering persistent class => DBCourseOffering

√ - Done

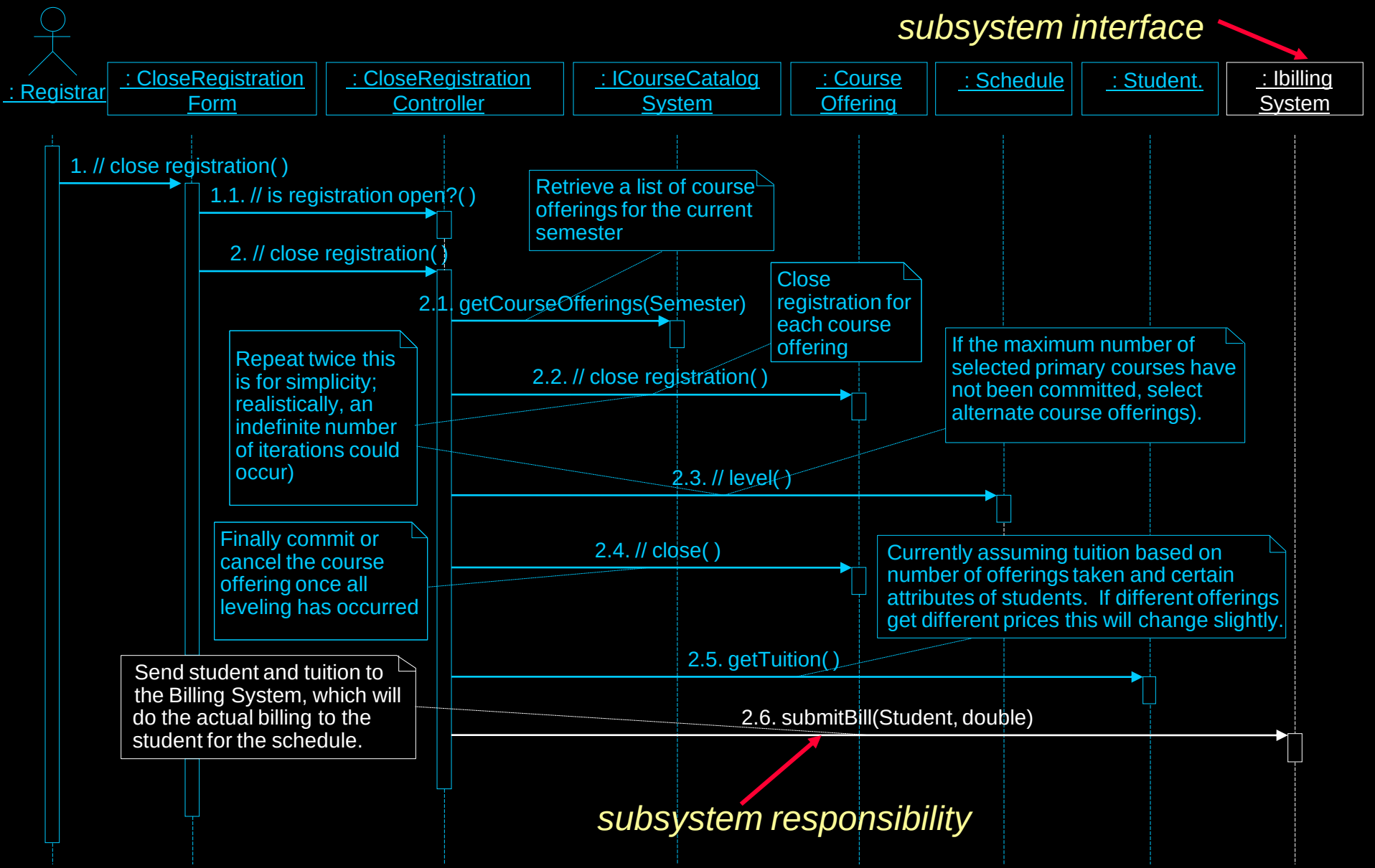
Review: Incorporating JDBC: Steps (cont.)

- Incorporate DBClasses into the design
  - Allocate to package/layer
    - *DBCourseOffering placed in CourseCatalogSystem subsystem*
  - Add relationships from persistency clients
    - *Persistency clients are the CourseCatalogSystem subsystem clients*
- Create/Update interaction diagrams that describe:
  - Database initialization
  - Persistent class access: Create, Read, Update, Delete

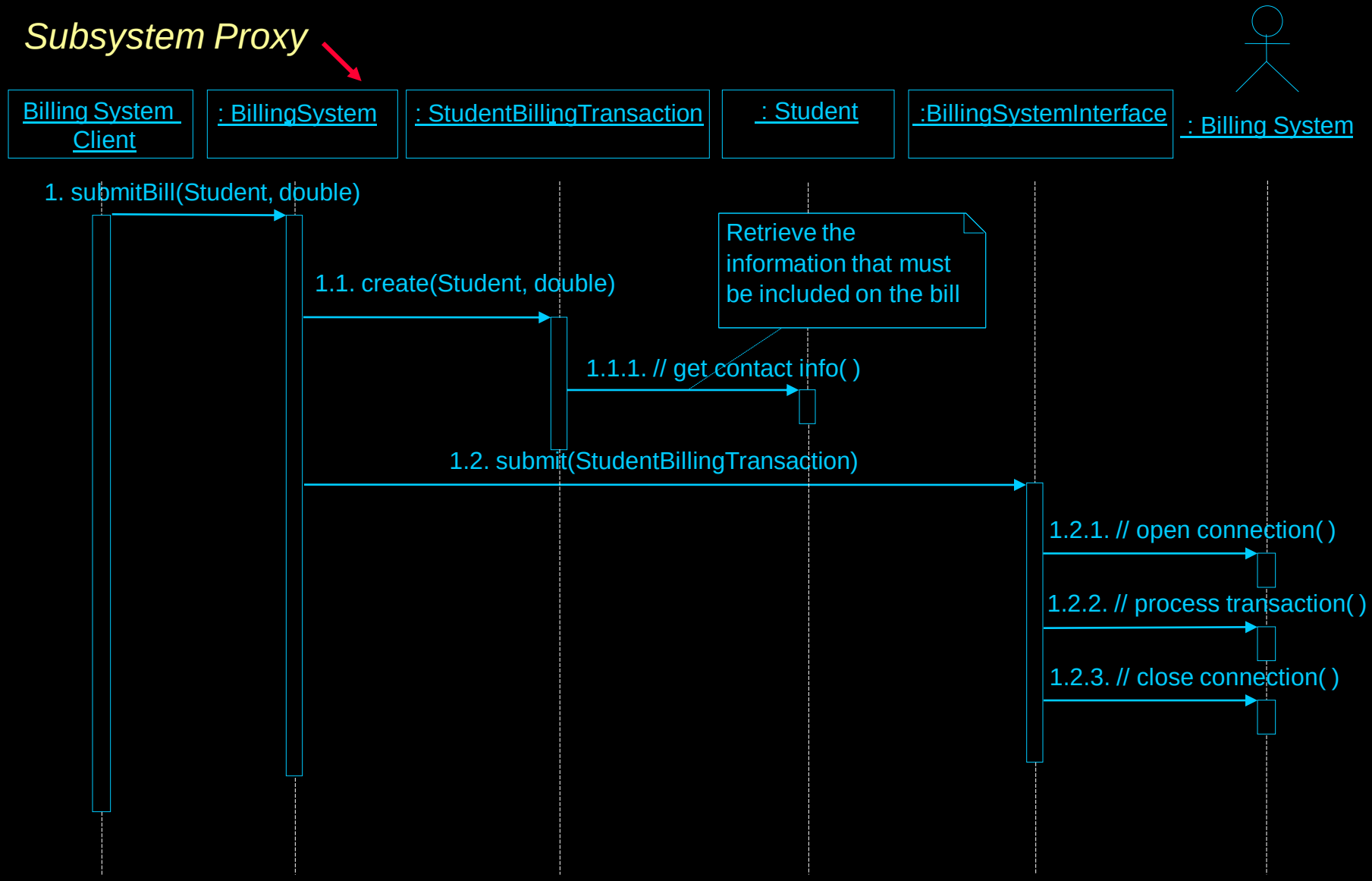
Example: Local CourseCatalogSystem Subsystem Interaction



Example: Billing System Subsystem In Context



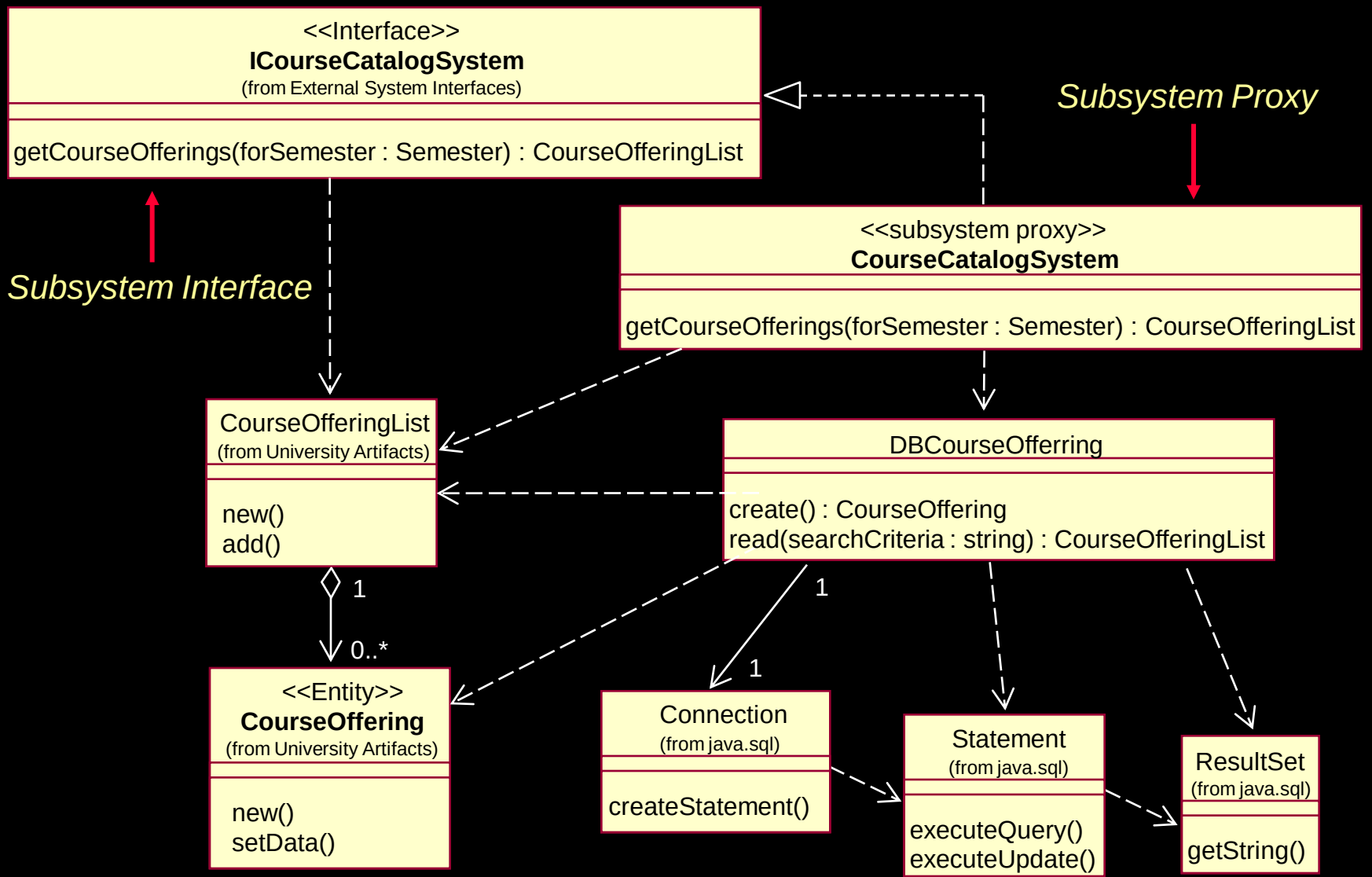
Example: Local BillingSystem Subsystem Interaction



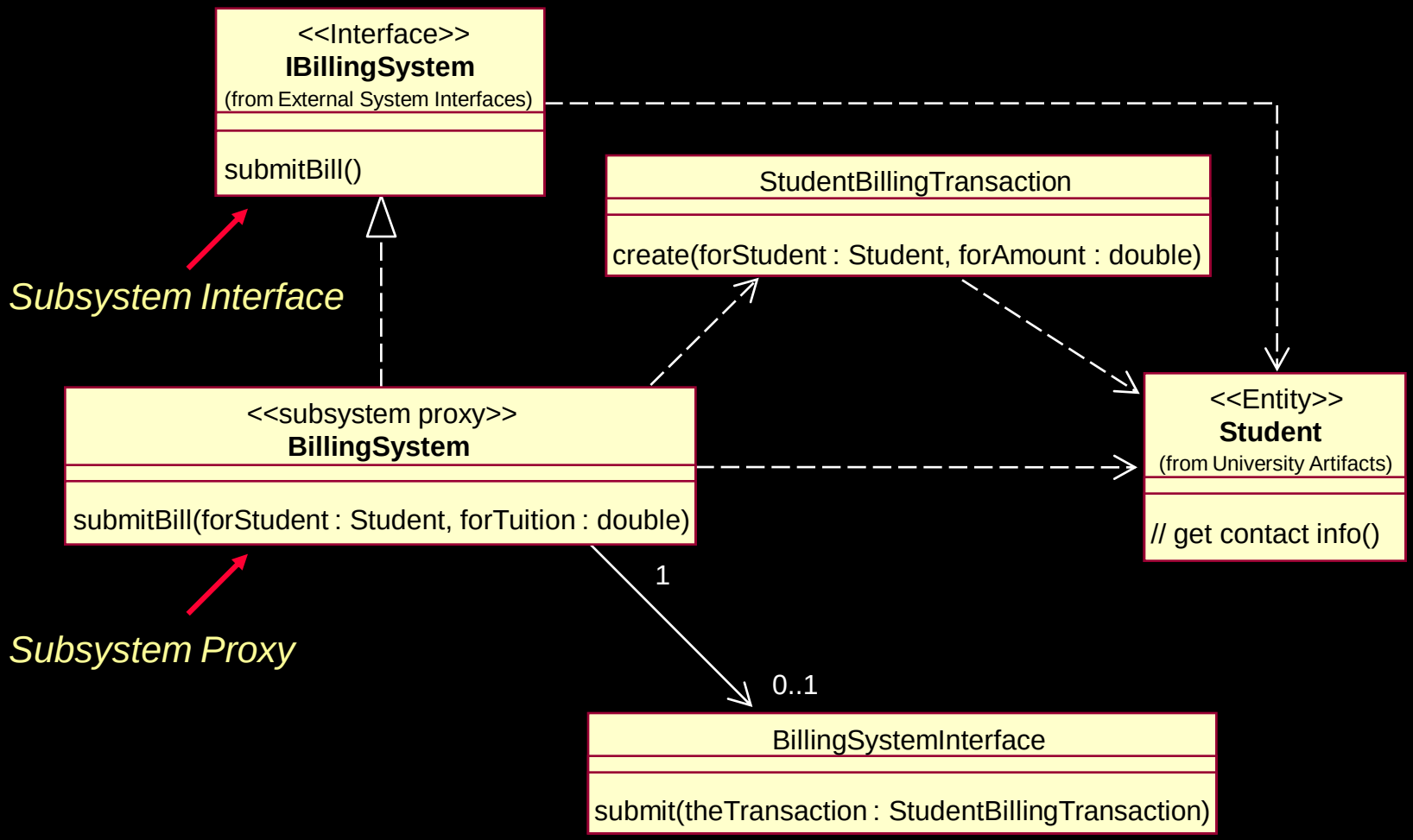
Subsystem Design Steps

- Distribute subsystem behavior to subsystem elements
- ★ Document subsystem elements
- Describe subsystem dependencies
- Checkpoints

Example: CourseCatalogSystem Subsystem Elements



Example: Billing System Subsystem Elements

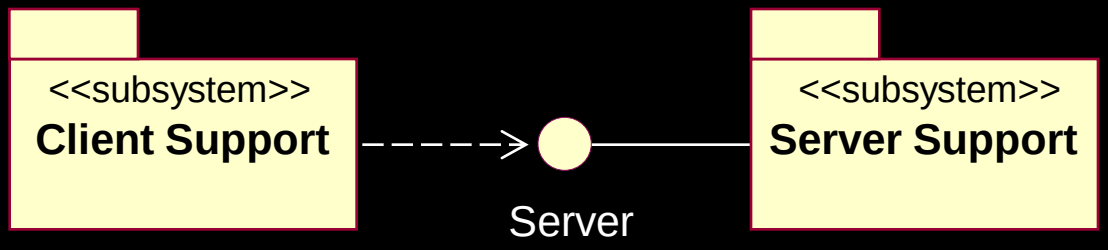


Subsystem Design Steps

- Distribute subsystem behavior to subsystem elements
- Document subsystem elements
- ★ Describe subsystem dependencies
- Checkpoints

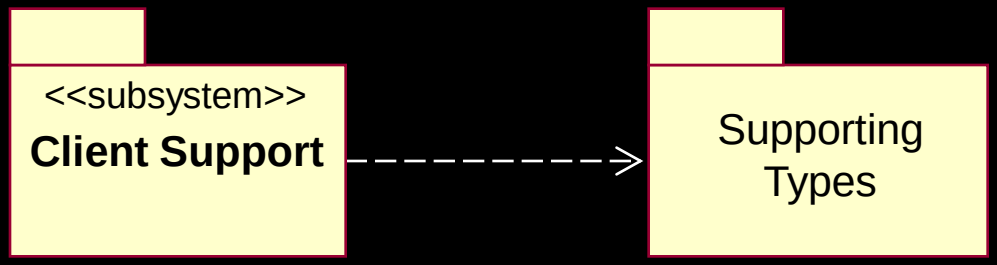
Subsystem Dependencies: Guidelines

- Subsystem dependency on a subsystem



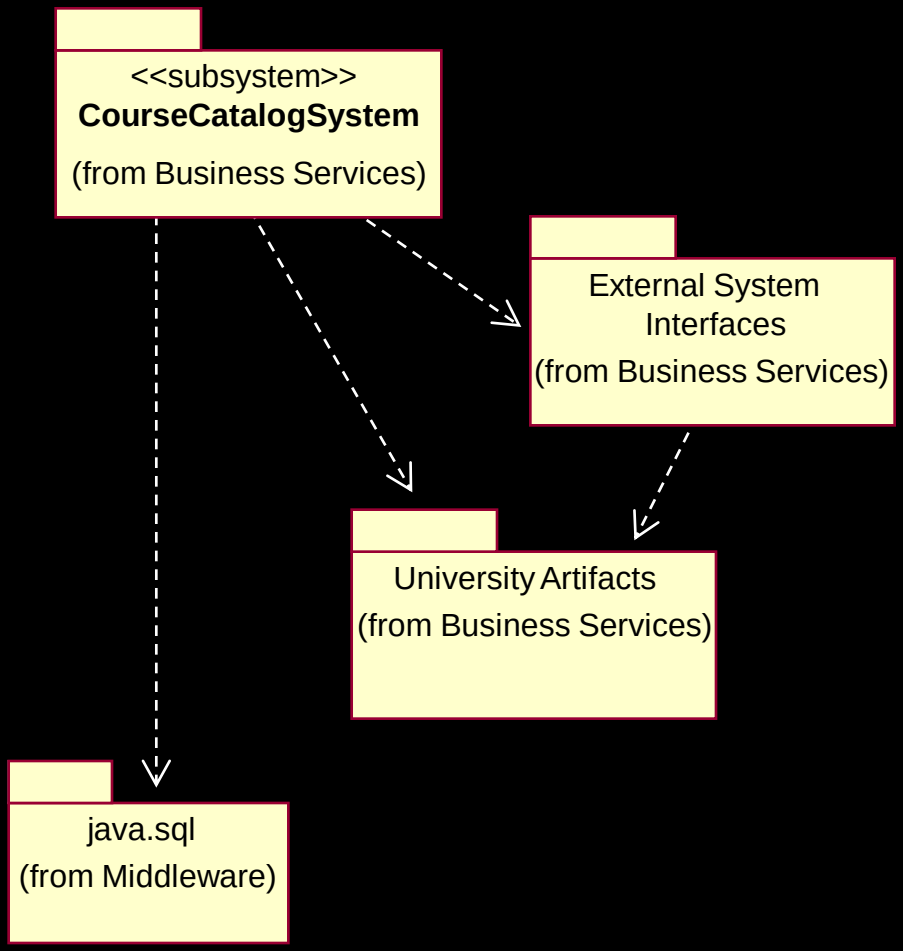
Flexible, Preferred

- Subsystem dependency on a package

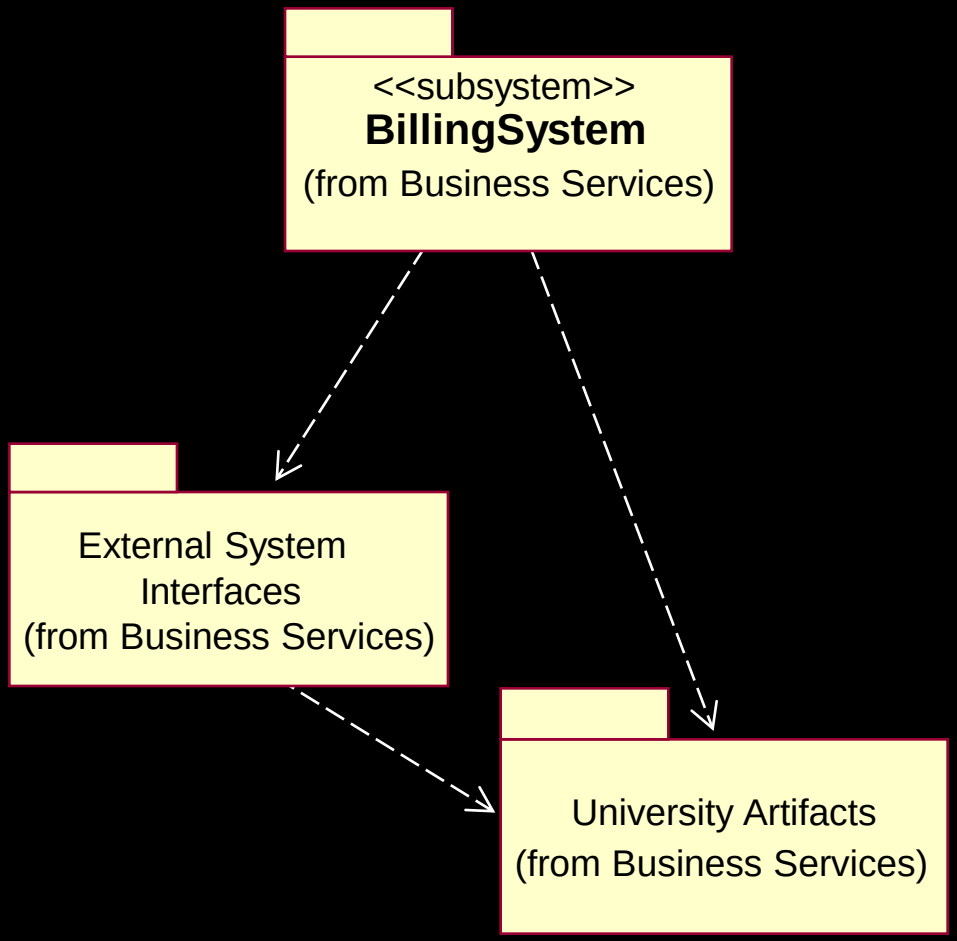


Use with care

Example: CourseCatalogSystem Subsystem Dependencies



Example: BillingSystem Subsystem Dependencies



Subsystem Design Steps

- Distribute subsystem behavior to subsystem elements
- Document subsystem elements
- Describe subsystem dependencies
- ★ Checkpoints

Checkpoints: Design Subsystems

- Is a realization association defined for each interface offered by the subsystem?
- Is a dependency association defined for each interface used by the subsystem?
- Are you sure that none of the elements within the subsystem have public visibility?
- Is each operation on an interface realized by the subsystem documented in a interaction diagram? If not, is the operation realized by a single class, so that it is easy to see that there is a simple 1:1 mapping between the class operation and the interface operation?

Review: Subsystem Design

- What is the purpose of Subsystem Design?
- How many interaction diagrams should be produced during Subsystem Design?
- Why should dependencies on a subsystem be on the subsystem interface?

Exercise: Subsystem Design

- Given the following:
  - The defined subsystems, their interfaces and their relationships with other design elements (the subsystem context diagrams)
  - Patterns of use for the architectural mechanisms

(continued)

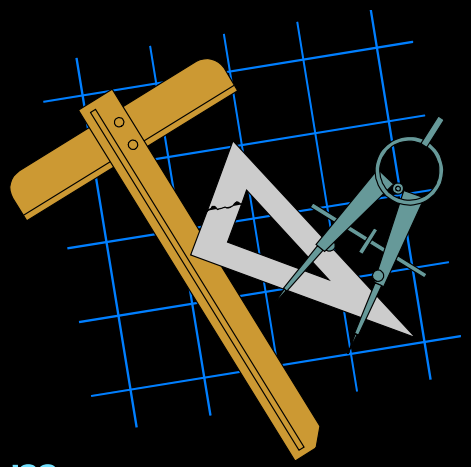
Exercise: Subsystem Design (cont.)

- Identify the following for a particular subsystem(s):
  - The design elements contained within the subsystem and their relationships
  - The applicable architectural mechanisms
  - The interactions needed to implement the subsystem interface operations

(continued)

Exercise: Subsystem Design (cont.)

- Produce the following diagrams for a particular subsystem(s):
  - "Interface realizations"
    - Interaction diagram for each interface operation
    - Class diagram containing the subsystem design elements that realize the interface responsibilities and their relationships
  - Class diagram that shows the subsystem and any dependencies on external package(s) and/or subsystem(s) (subsystem dependencies class diagram)



Exercise: Review

- Compare your Subsystem Interface Realizations
  - Have all the main and/or subflows for the interface operations been handled?
  - Has all behavior been distributed among the participating design elements?
  - Has behavior been distributed to the right design elements?
  - Are there any messages coming from the interfaces?