ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA HỆ THỐNG THÔNG TIN

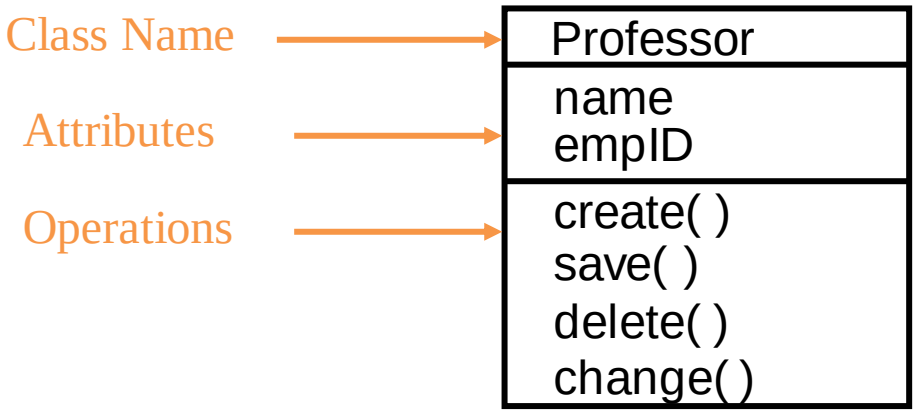
CHƯƠNG 4
MÔ HÌNH HÓA TƯƠNG TÁC ĐỐI TƯỢNG

Nội dung

1. Xác định class
2. Biểu đồ trình tự
3. Biểu đồ cộng tác

Nhắc lại: Class

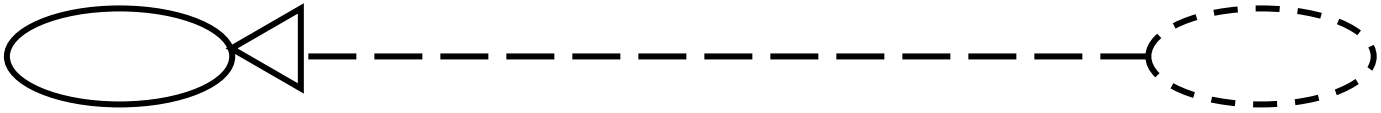
- Là một sự trừu tượng hóa
- Mô tả một nhóm các đối tượng có chung:
  - Properties (attributes)
  - Behavior (operations)
  - Relationships
  - Ngữ nghĩa (Semantics)



Nhắc lại: Use case Realization

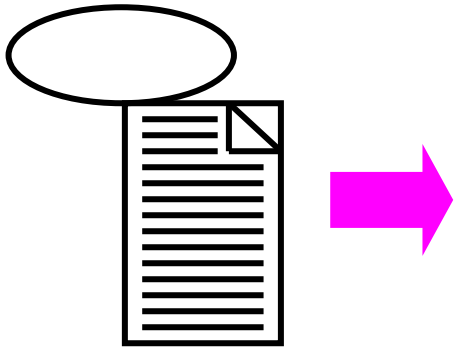
Use-Case Model

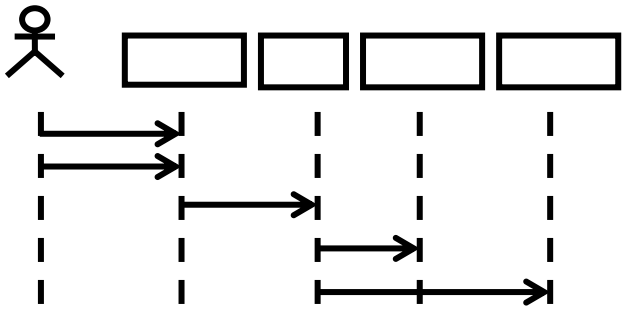
Design Model



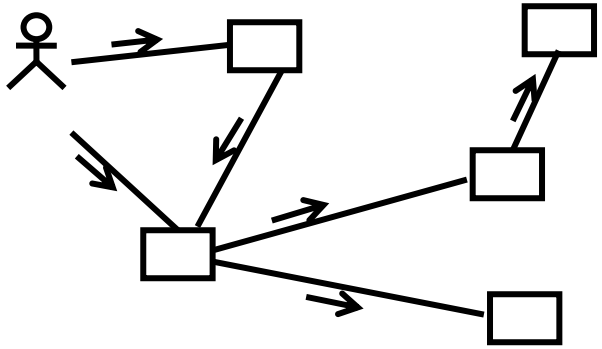
Use Case

Use-Case Realization

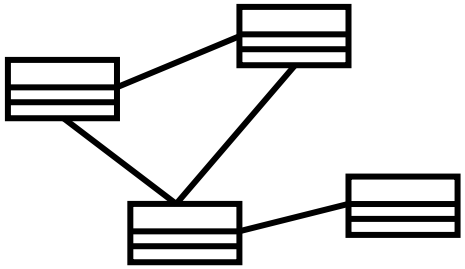




Sequence Diagrams



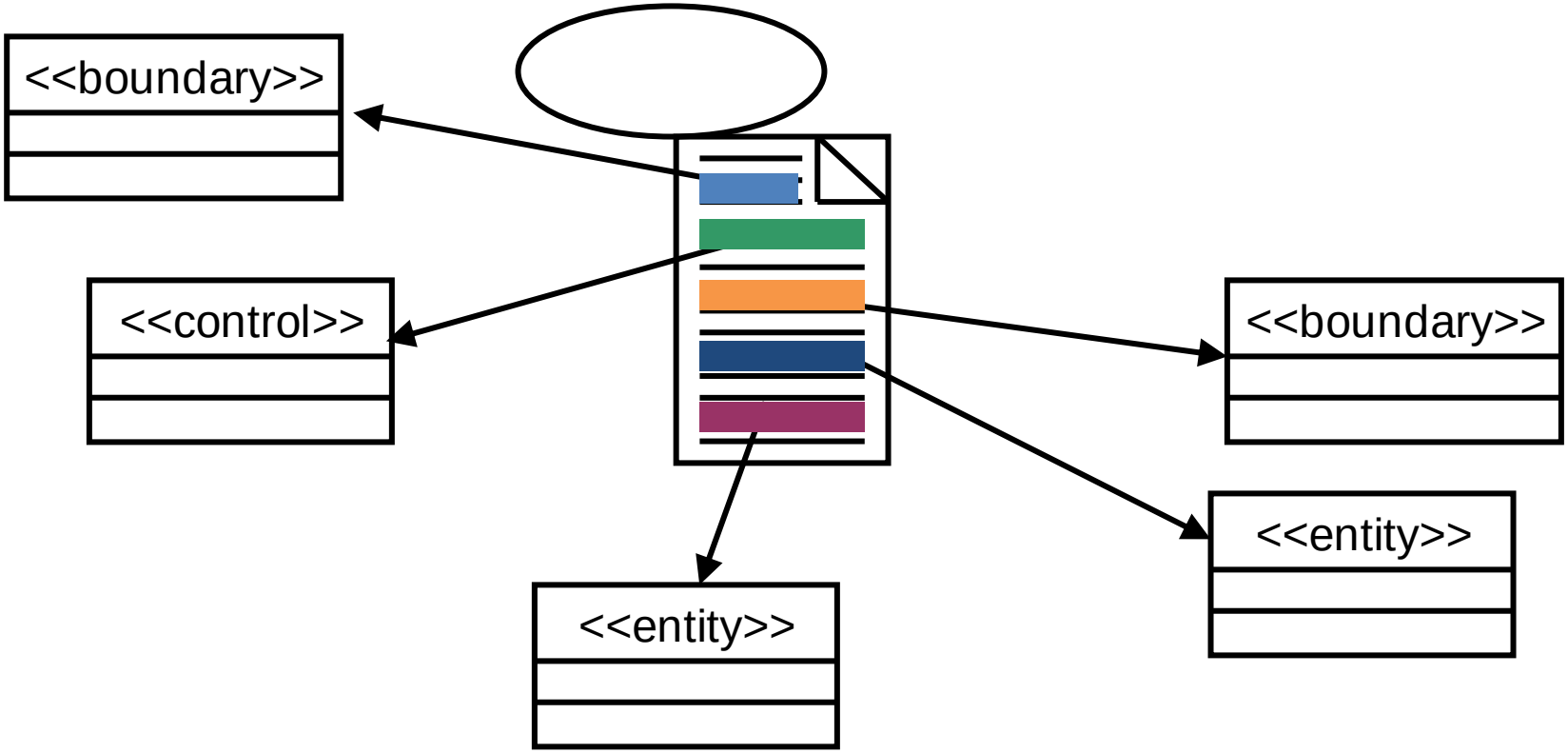
Collaboration Diagrams



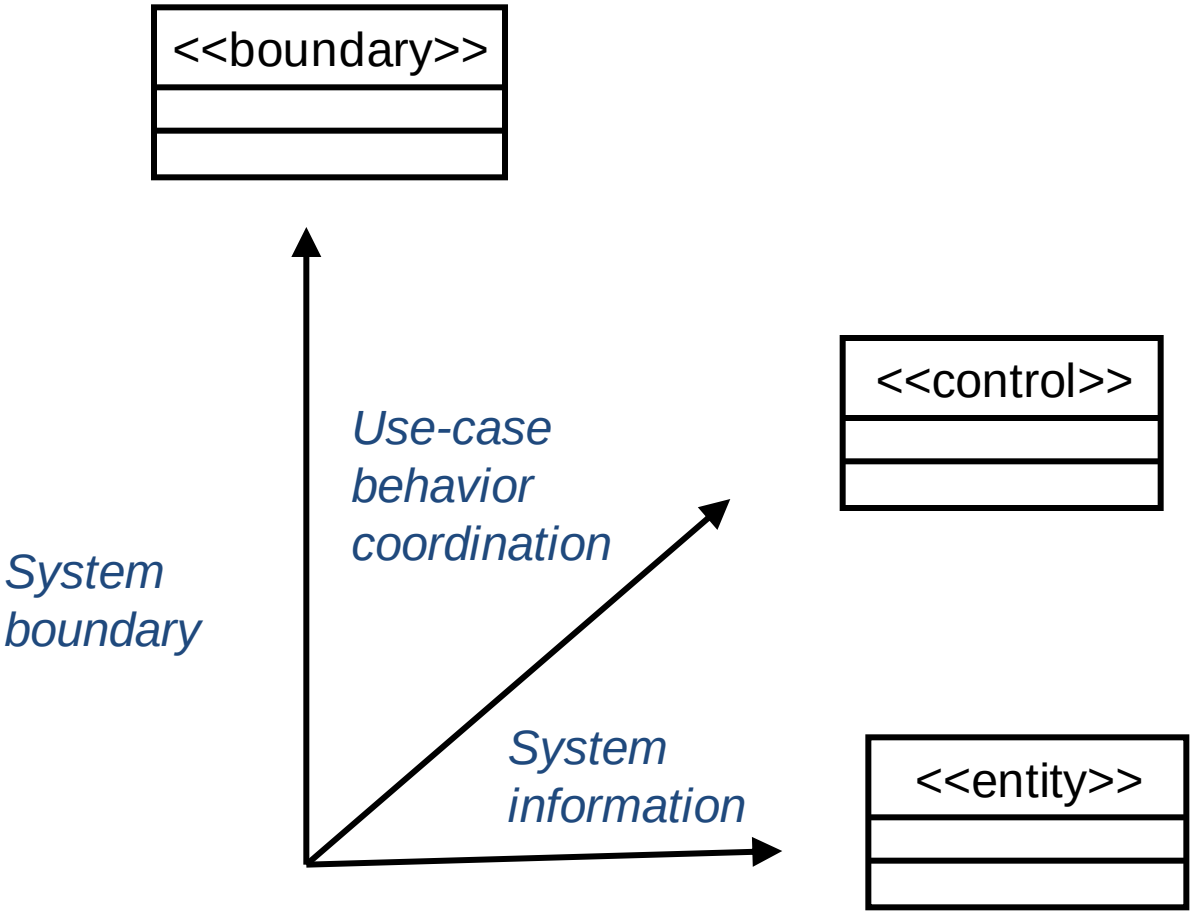
Class Diagrams

Tìm các Class từ Use case Behavior

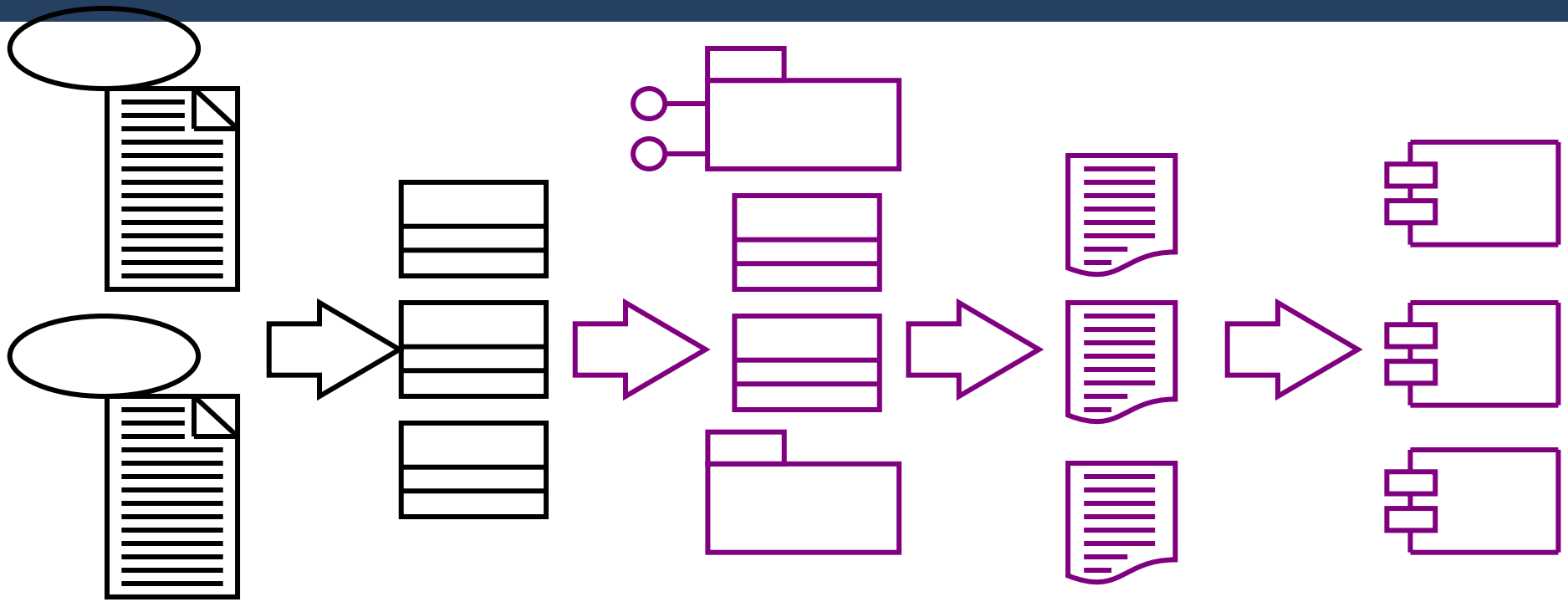
- Toàn bộ hành vi của một use case phải được phân bổ về cho các analysis class



Thế nào là một Analysis Class?



Analysis Classes

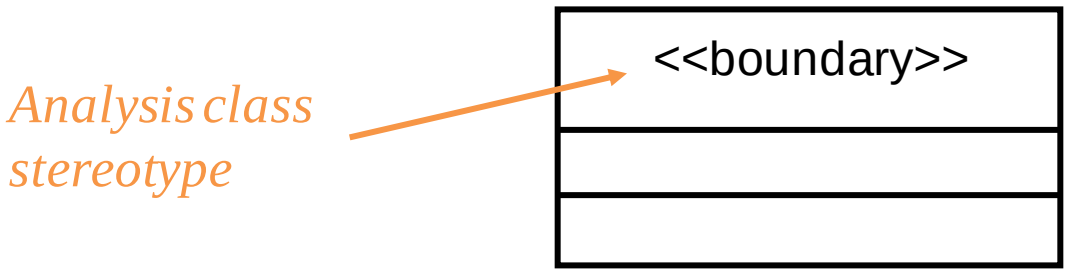


Use Cases Analysis Classes Design Elements Source Code Exec

Use-Case Analysis

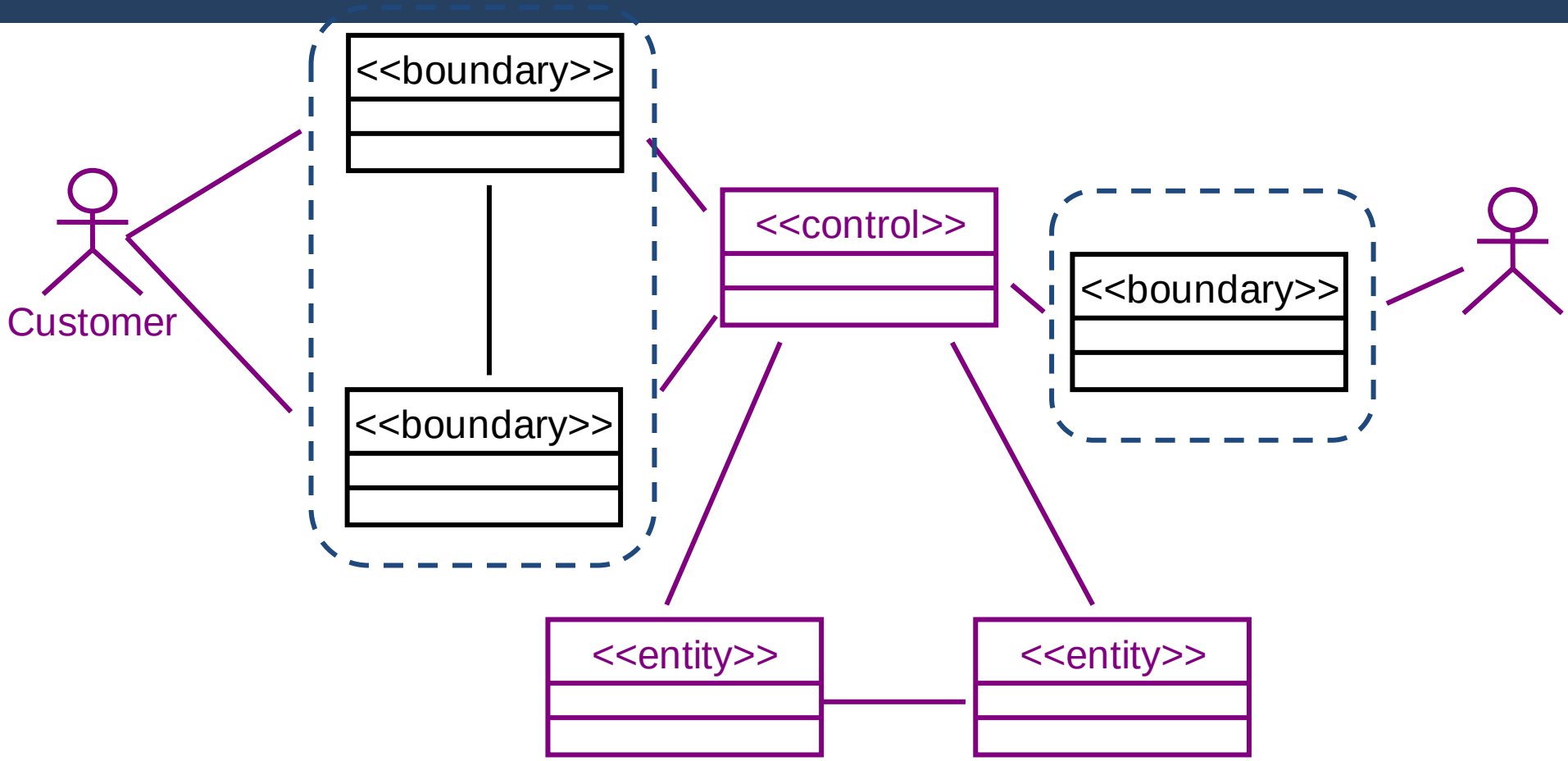
Thế nào là một Boundary Class?

- Làm trung gian giao tiếp với những gì nằm ngoài hệ thống
- Một số kiểu
  - Các User interface class
  - Các System interface class
  - Các Device interface class
- *Một boundary class cho 1 cặp actor/use case*



Phụ thuộc môi trường

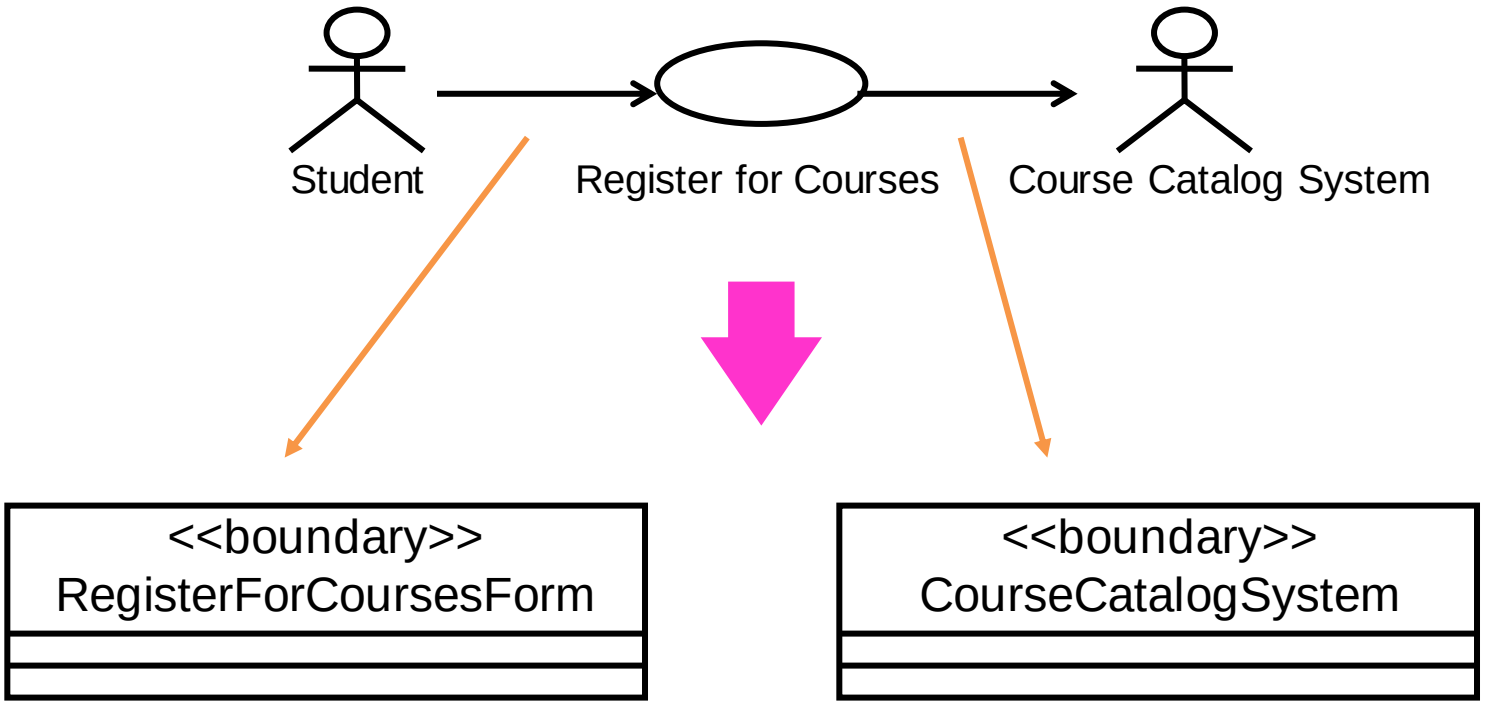
Vai trò của Boundary Class



Mô hình hóa sự tương tác giữa system và môi trường của nó

Ví dụ: Tìm các Boundary Class

- Một boundary class cho 1 cặp actor/use case



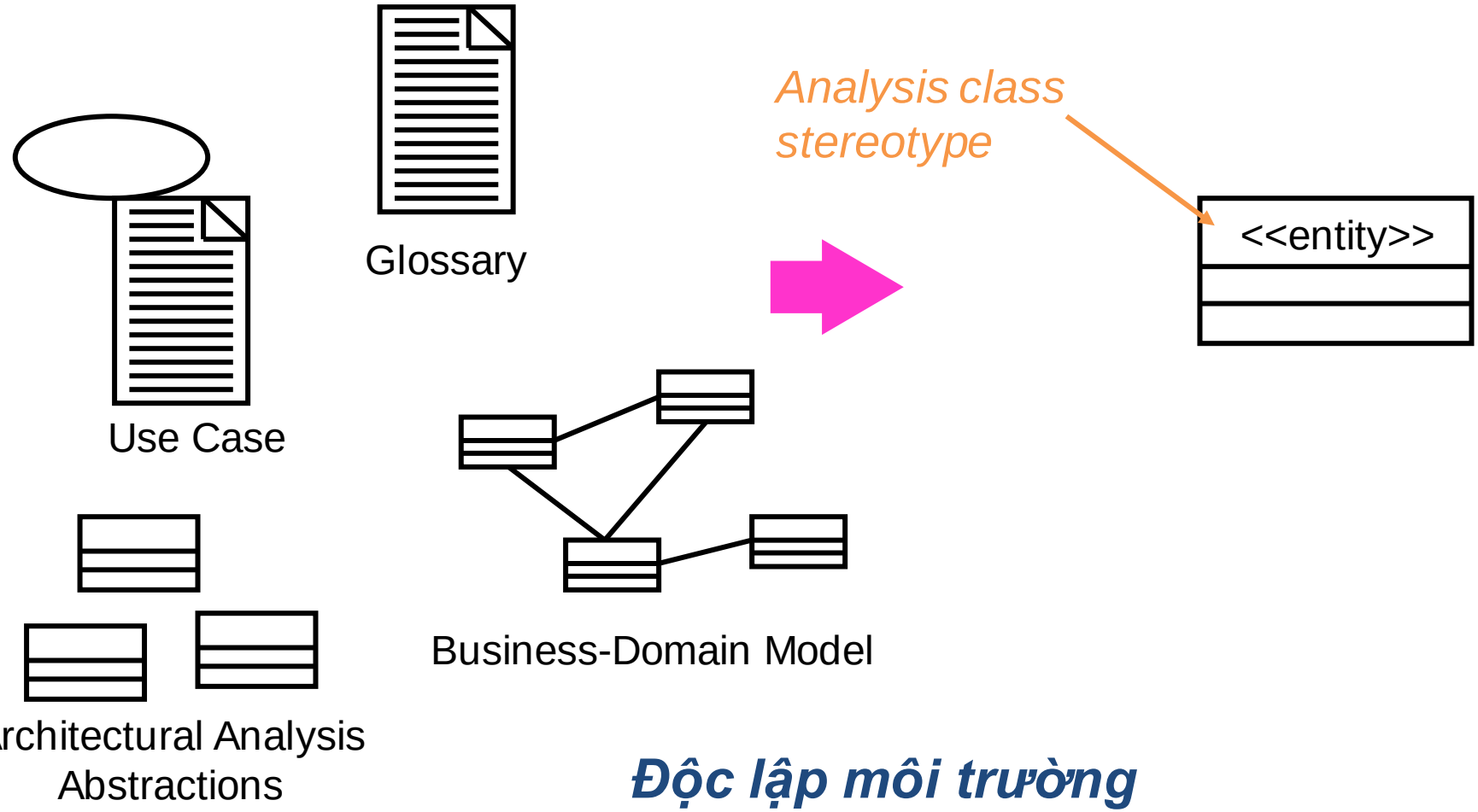
Boundary Class

- Các User Interface Class
  - Tập trung vào những thông tin gì được thể hiện cho người dùng
  - **KHÔNG** tập trung vào các chi tiết UI
- Các System và Device Interface Class
  - Tập trung vào những protocols nào phải định nghĩa
  - **KHÔNG** tập trung vào cách mà các protocol sẽ được cài đặt

Tập trung vào các nhiệm vụ, chứ không phải chi tiết!

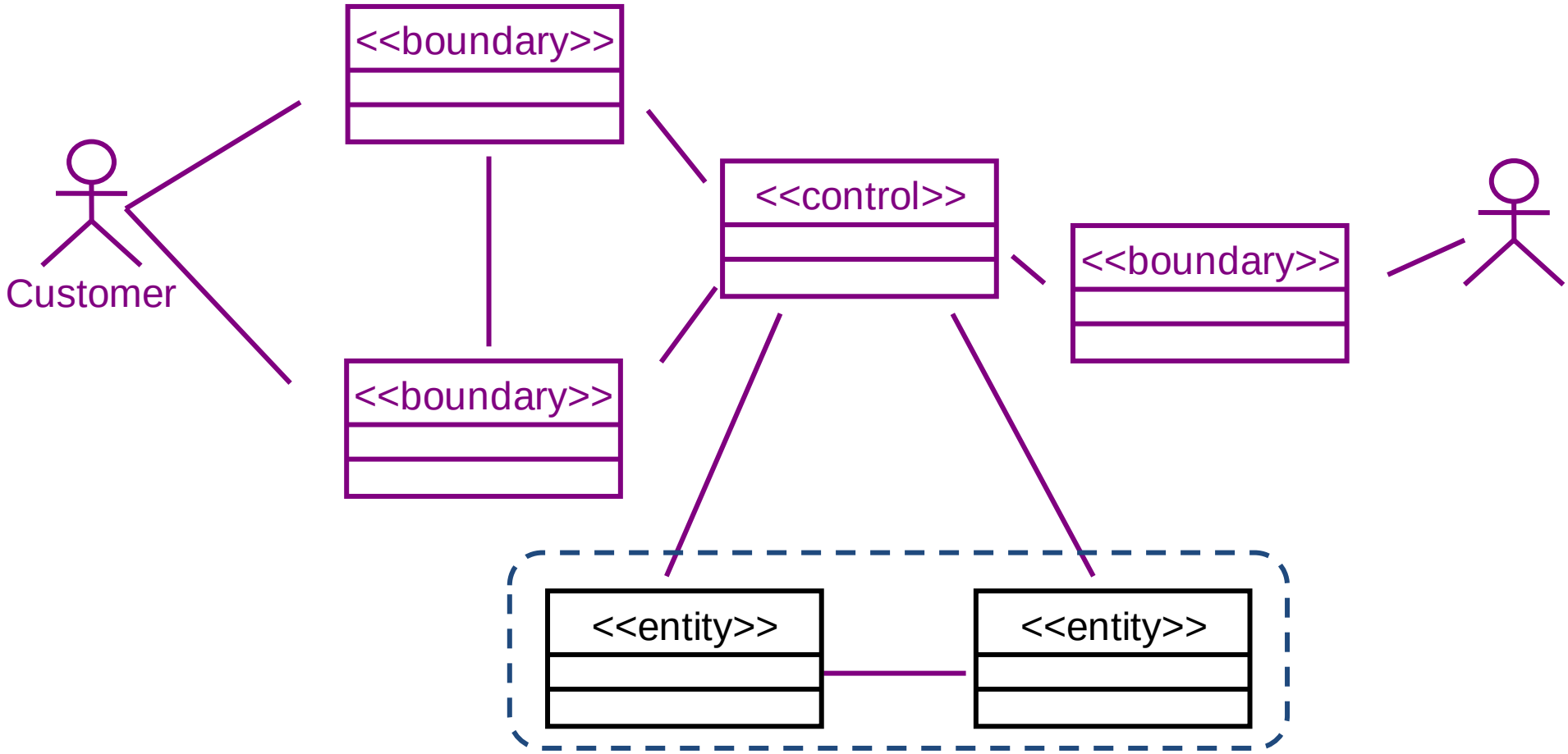
Thế nào là một Entity Class?

- Các trừu tượng hóa then chốt của system



Độc lập môi trường

Vai trò của Entity Class



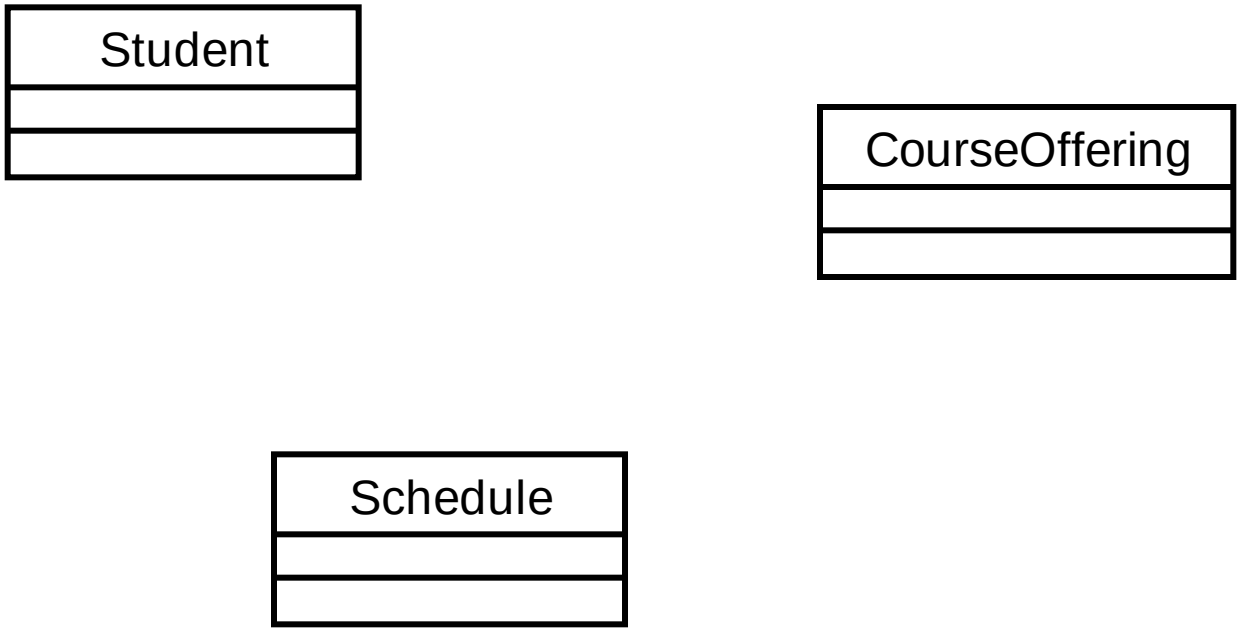
Lưu trữ và quản trị các thông tin trong system

Ví dụ: Tìm các Entity Class

- Dùng use-case flow of events như input
- Các trừu tượng hóa then chốt của use case
- Hướng tiếp cận truyền thống (nouns filtering)
  - Gạch dưới các cụm danh từ trong flow of events
  - Loại bỏ các ứng viên dư thừa
  - Loại bỏ các ứng viên mơ hồ, không rõ ràng
  - Loại bỏ các actor (ngoài phạm vi)
  - Loại bỏ các kiến trúc cài đặt
  - Loại bỏ các attribute (để lại dùng sau)
  - Loại bỏ các operation

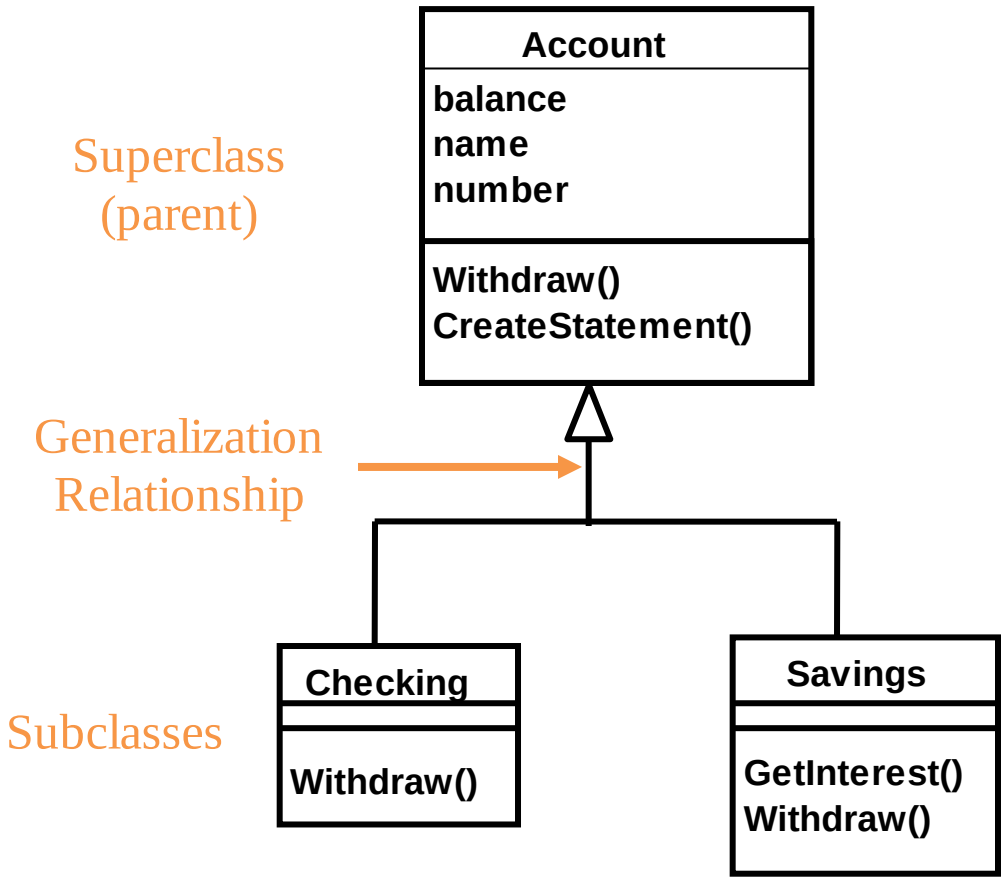
Ví dụ: Entity Class

- Register for Courses (Create Schedule)

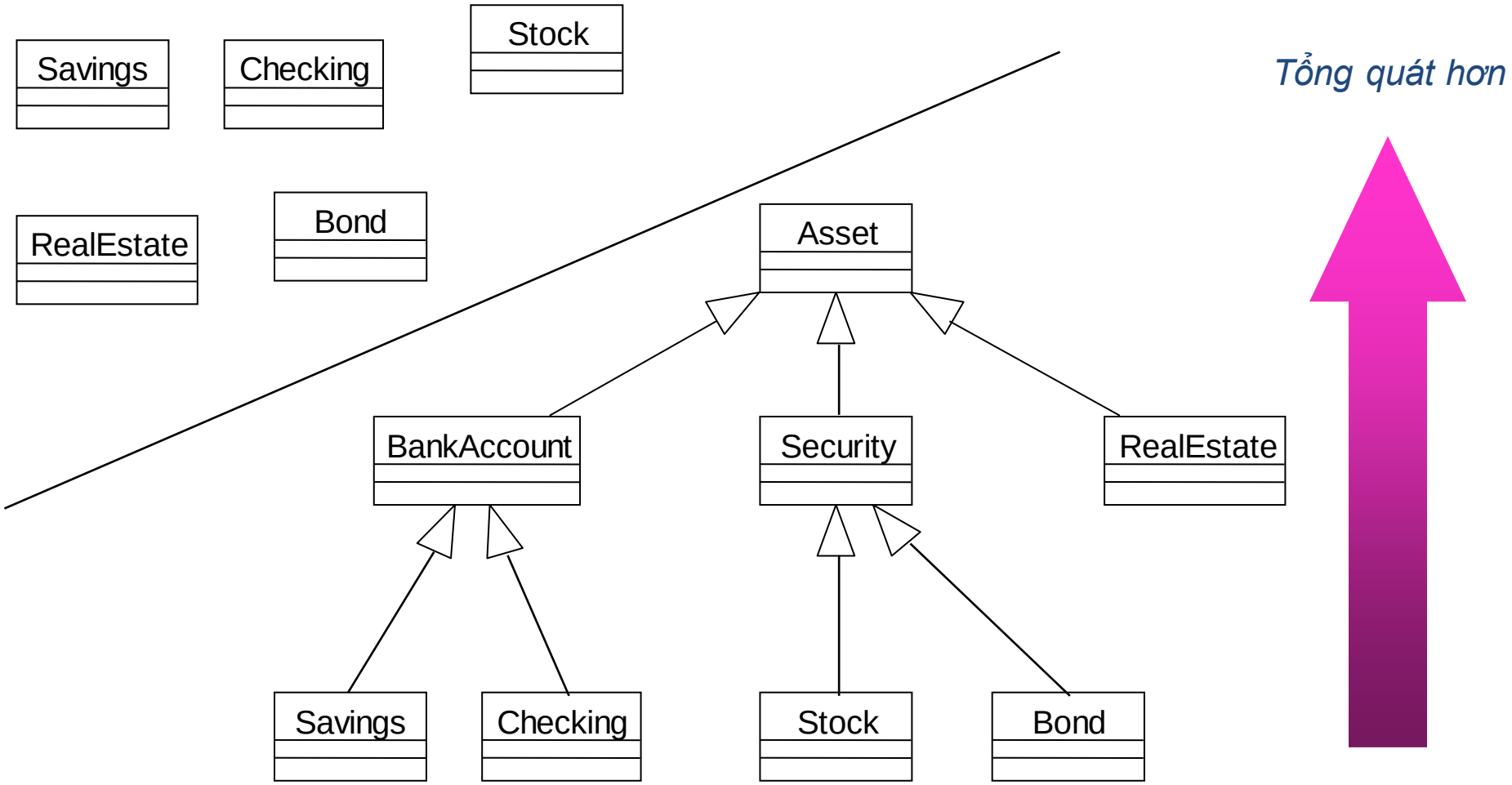


Nhắc lại: Generalization

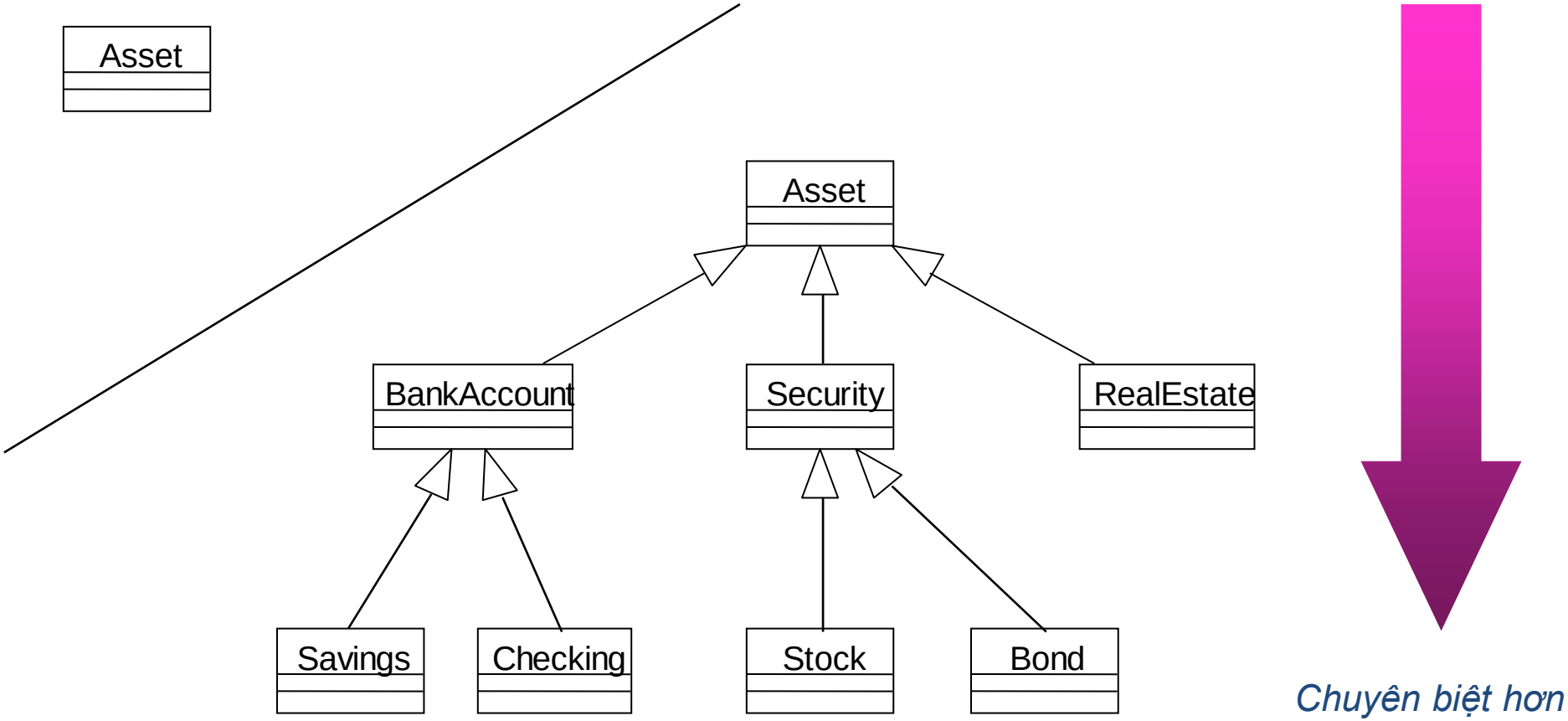
- Một class chia sẻ cấu trúc và/hoặc hành vi của một hay nhiều class
- Mối quan hệ "Là một dạng của"
- Trong phân tích, sử dụng ở mức độ đơn giản, sơ sài



Generalization

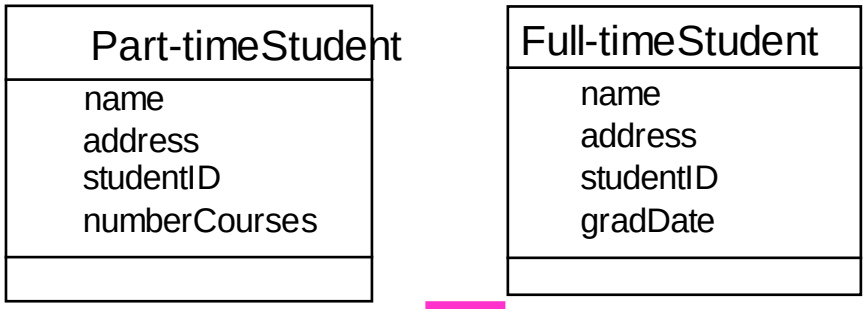


Generalization

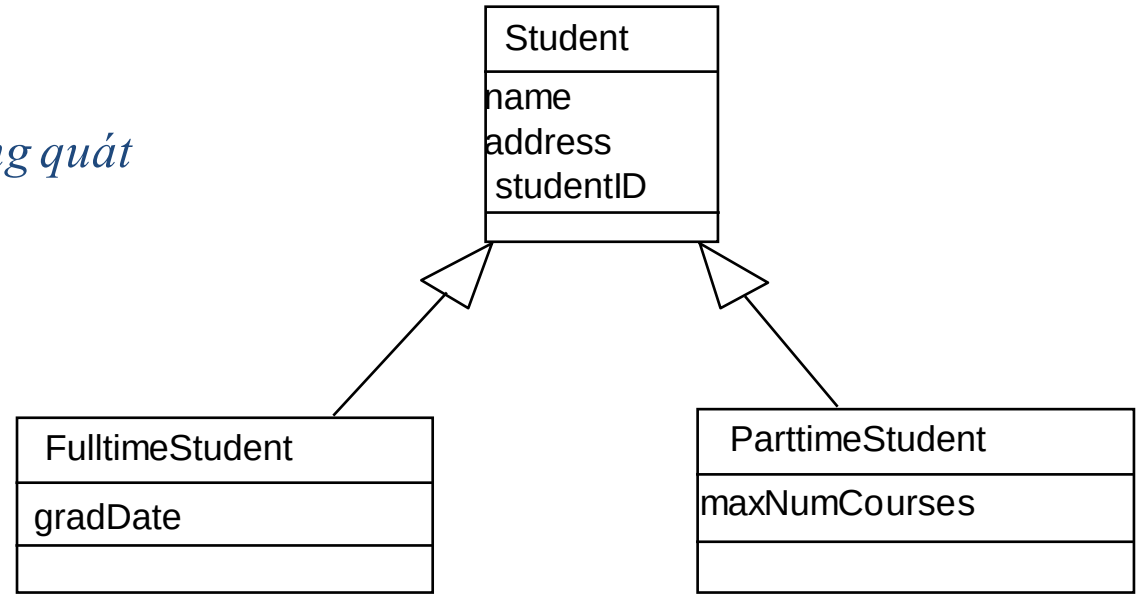


Ví dụ: Generalization (Chia sẻ ngữ nghĩa)

Không có sự tổng quát hóa

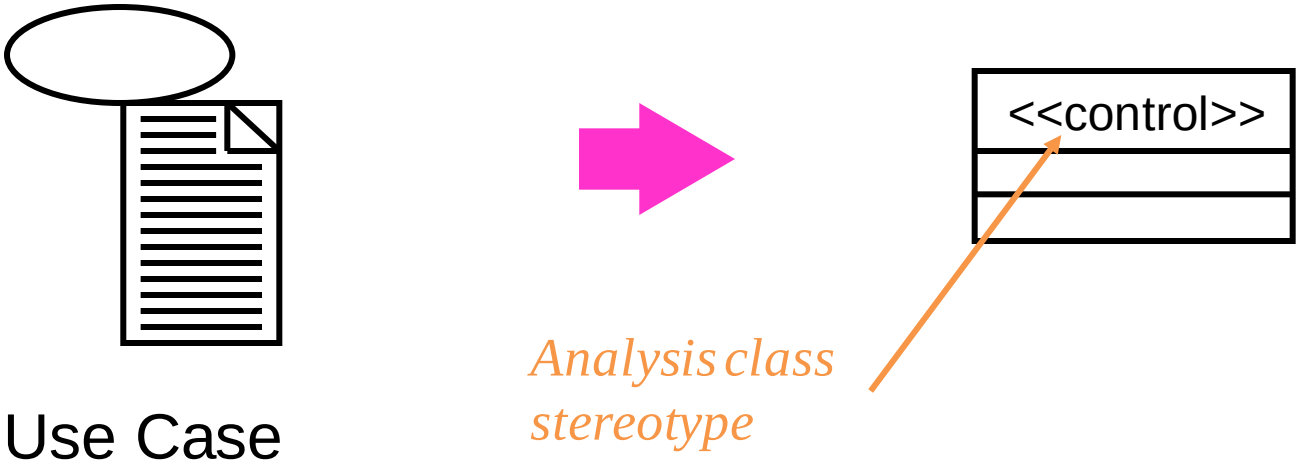


Có sự tổng quát hóa



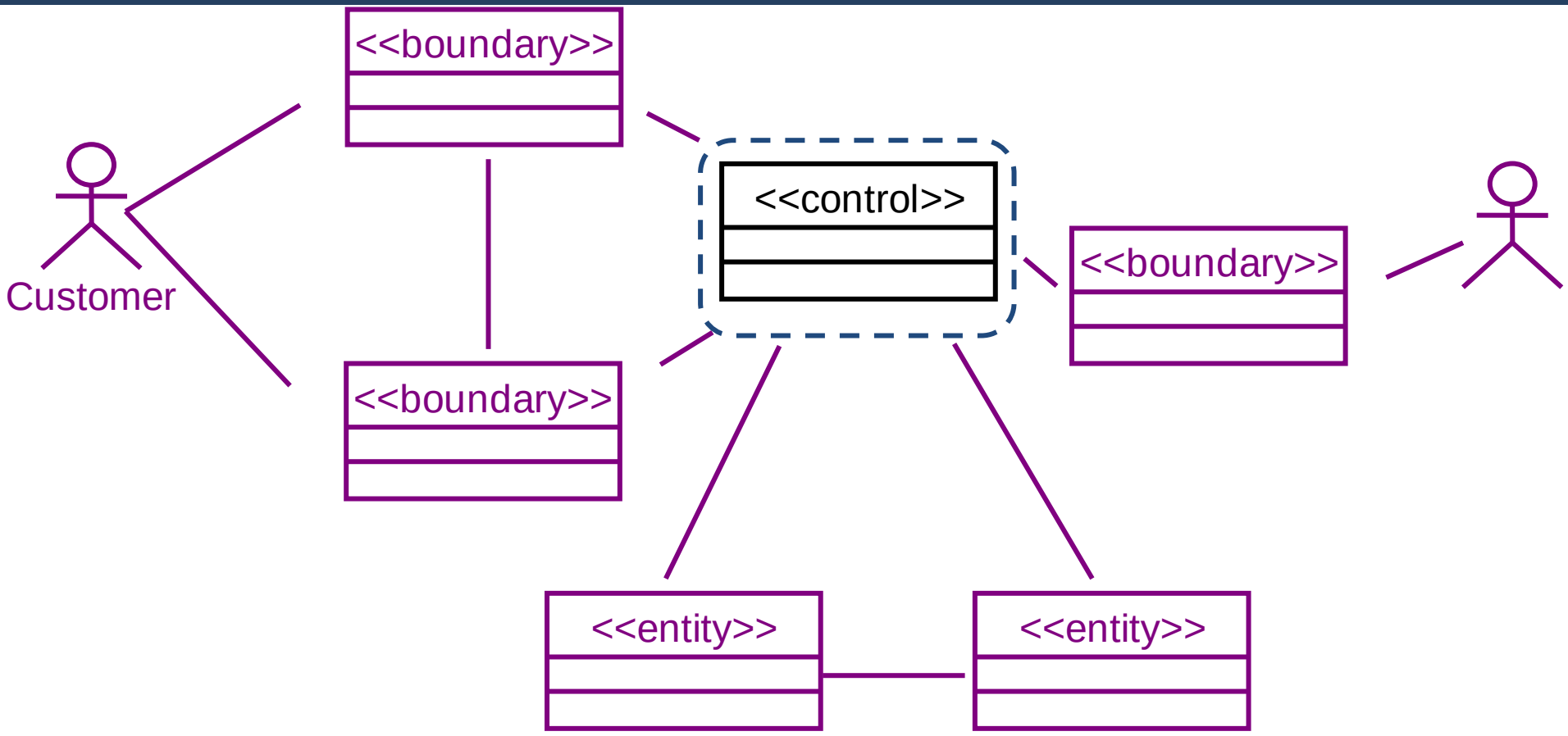
Thế nào là một Control Class?

- Nhà điều phối các hành vi của Use-case
- *Chỉ một control class cho một use case*



Phụ thuộc use-case, độc lập môi trường

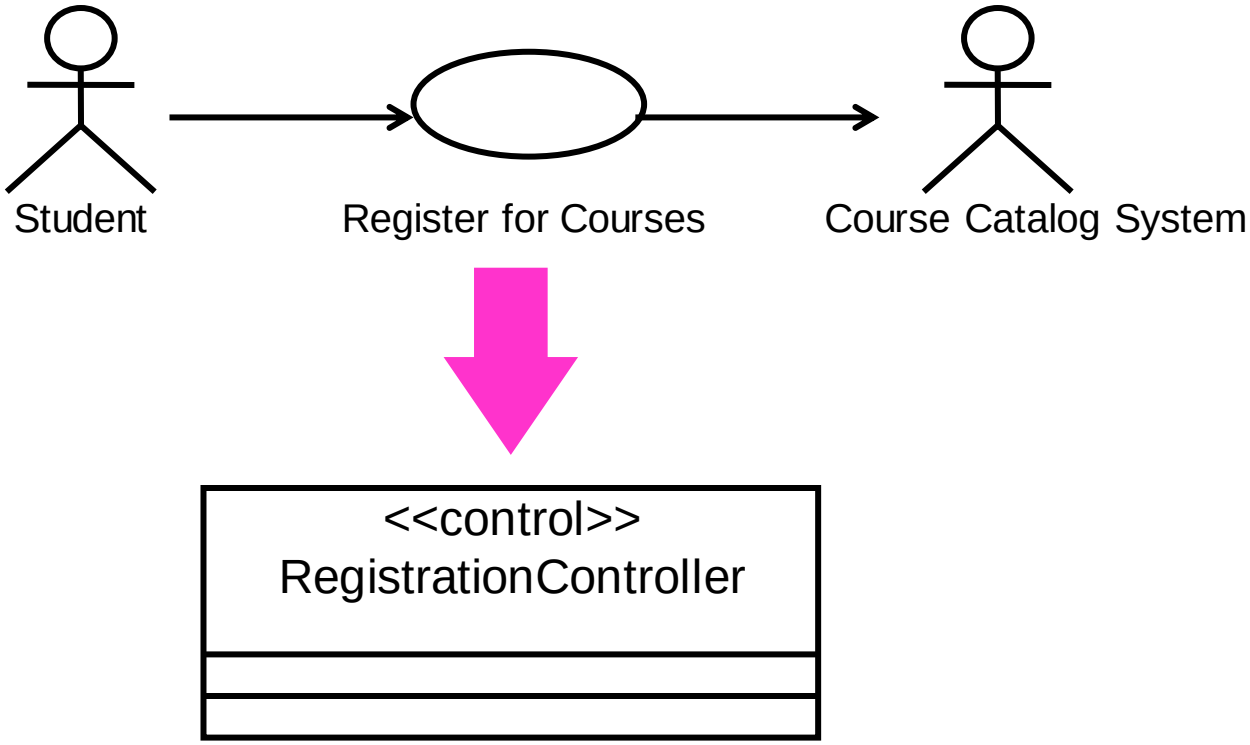
Vai trò của Control Class



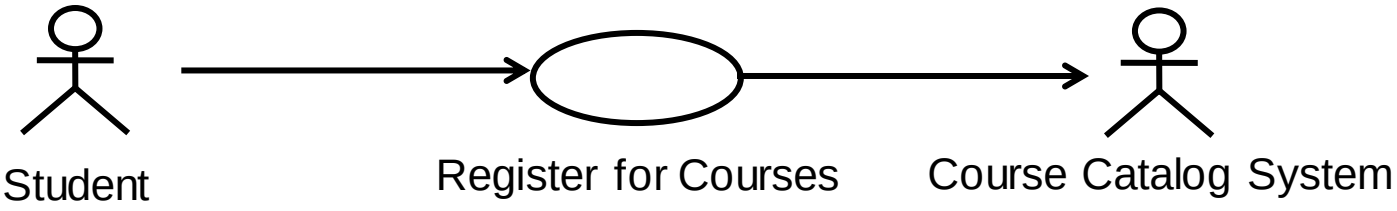
Điều phối các hành vi của use-case

Ví dụ: Tìm các Control Class

- Một control class cho một use case

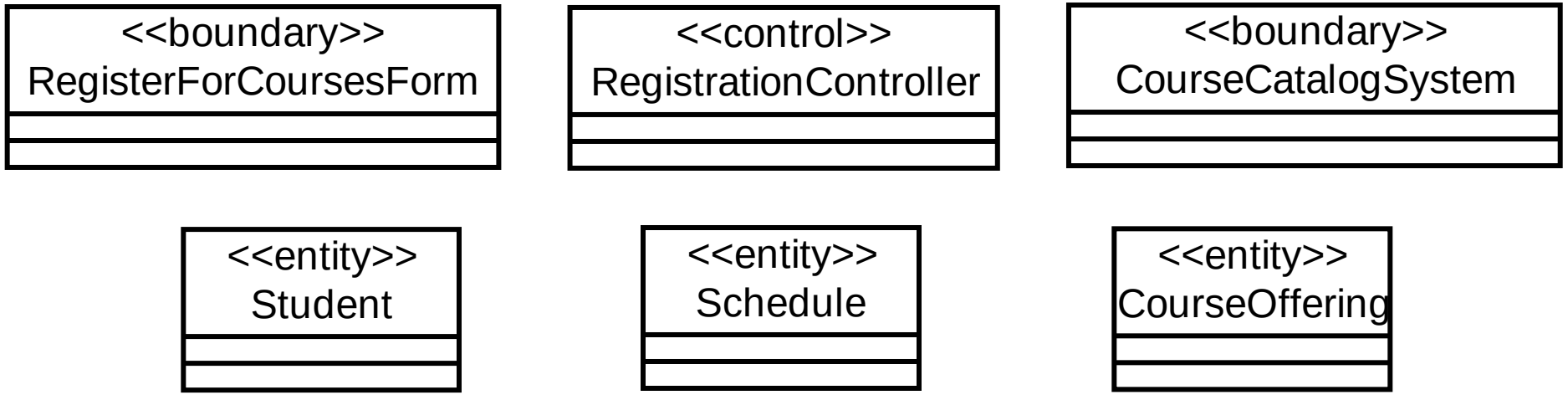


Ví dụ: Analysis Classes



Use-Case Model

Design Model



Nội dung

1. Xác định class
2. Biểu đồ trình tự
   - Định nghĩa
   - Chức năng
   - Cấu trúc và các thành phần
3. Biểu đồ cộng tác

2. Biểu đồ trình tự Sequence Diagram

- Định nghĩa:
  - Lược đồ tuần tự thể hiện các hành vi động hướng thời gian
  - Đặc biệt hữu dụng trong các hệ thống với các chức năng phụ thuộc vào thời gian hoặc yêu cầu trình tự thời gian là quan trọng.

2. Sequence Diagram

- Chức năng
  - Mô hình hóa luồng xử lý
  - Minh họa các kịch bản đặc trưng
  - Mô tả sự tuần tự của các sự kiện, thể hiện khi nào đối tượng được tạo và hủy, mô tả các hành động đồng thời

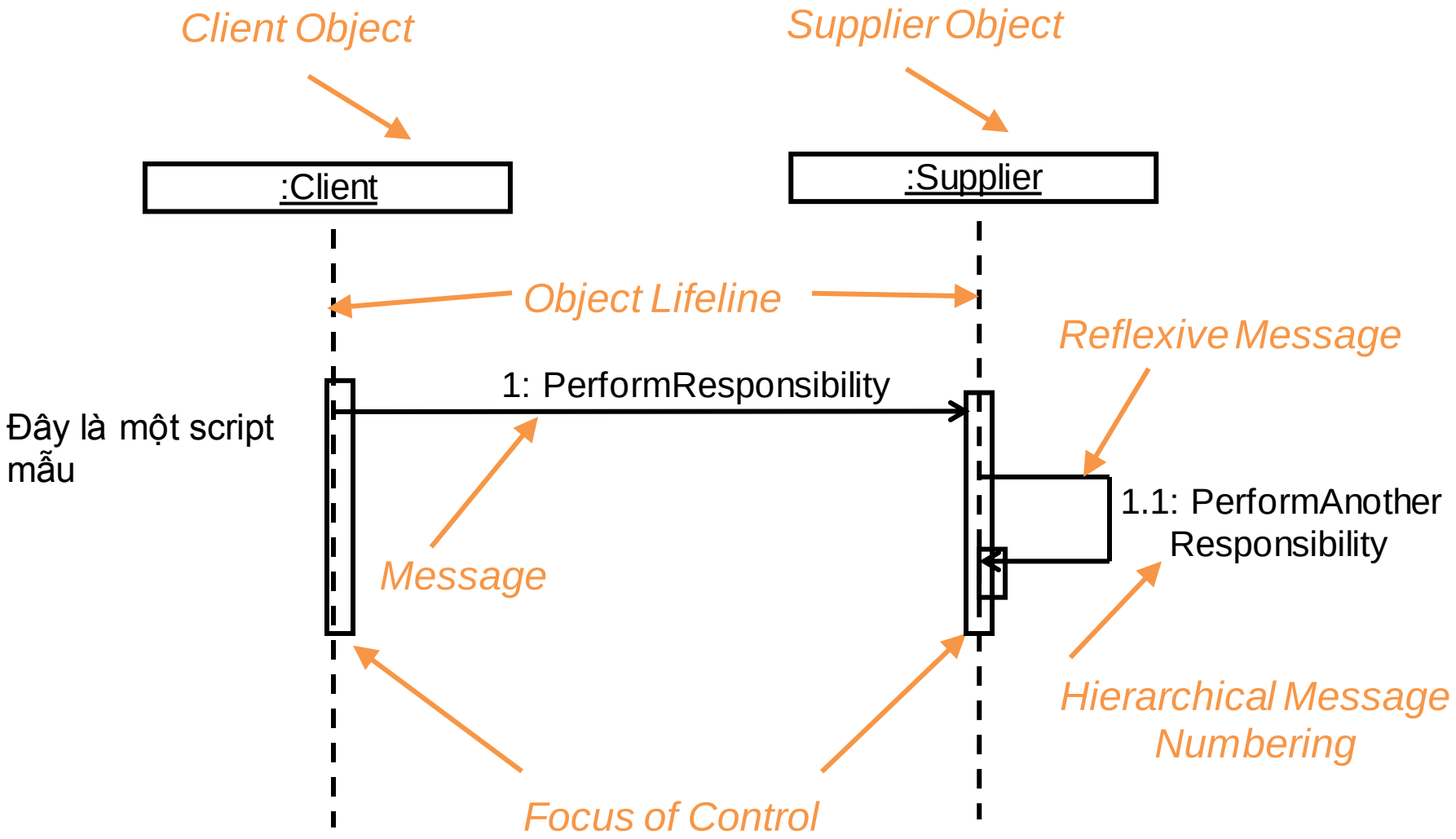
2. Sequence Diagram

- Một lược đồ tuần tự bao gồm có hai phương
  - Phương ngang: các đối tượng khác nhau trong chuỗi hành động tuần tự để thực hiện một chức năng nào đó.
  - Phương thẳng đứng : biểu diễn trục thời gian theo hướng từ trên xuống.

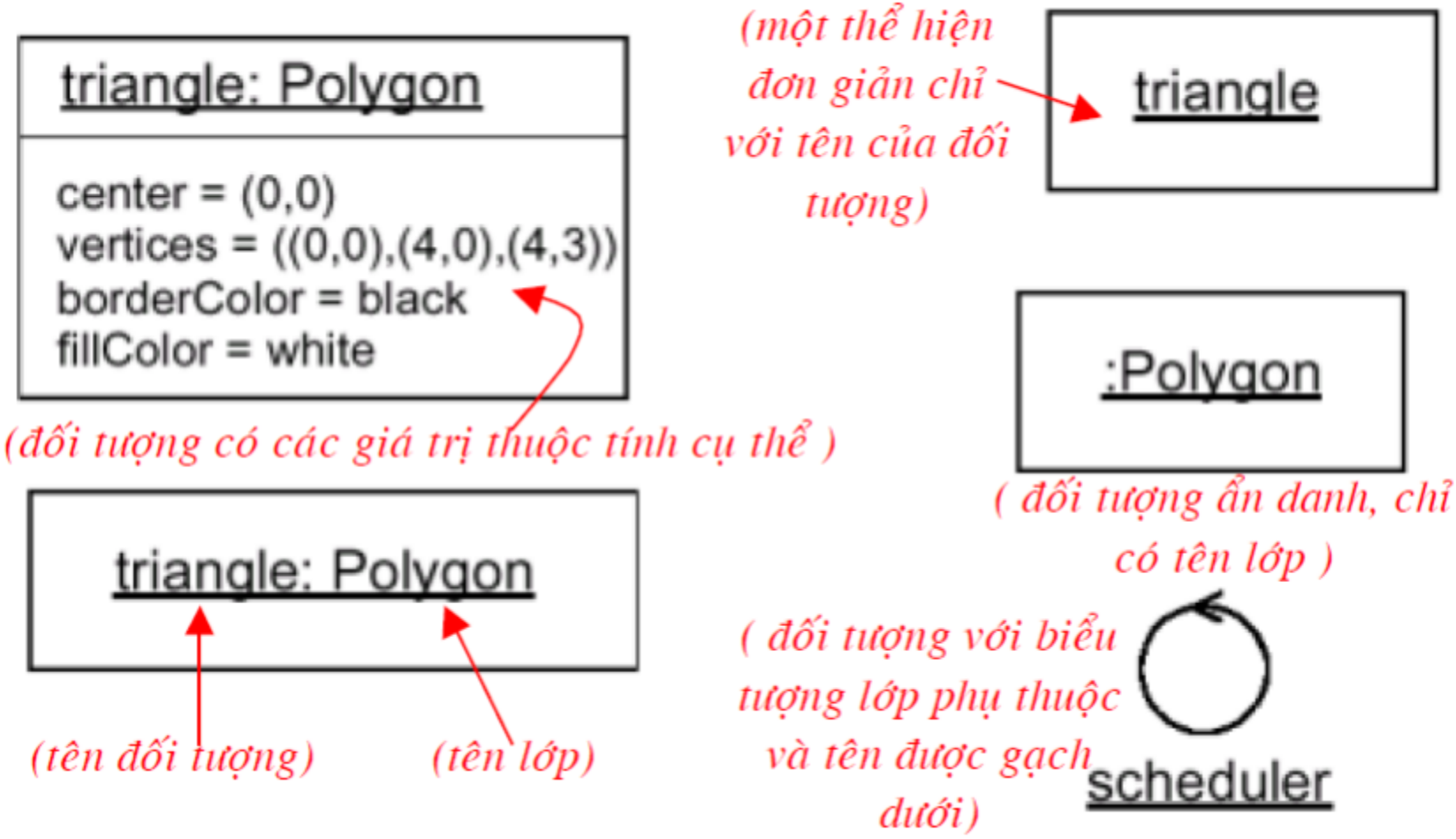
2. Sequence Diagram

- Cấu trúc: Một lược đồ tuần tự bao gồm 4 thành tố chính
  - ***Object***: các đối tượng khác nhau liên quan đến lược đồ
  - ***Lifeline***: thể hiện sự tồn tại của đối tượng trên trục thời gian
  - ***Activation – focus of control***: thể hiện bằng hình chữ nhật nằm trên lifeline. Độ dài của focus of control cho biết thời gian mà đối tượng tồn tại.
  - ***Message***: thể hiện sự liên lạc giữa các đối tượng

2. Sequence Diagram

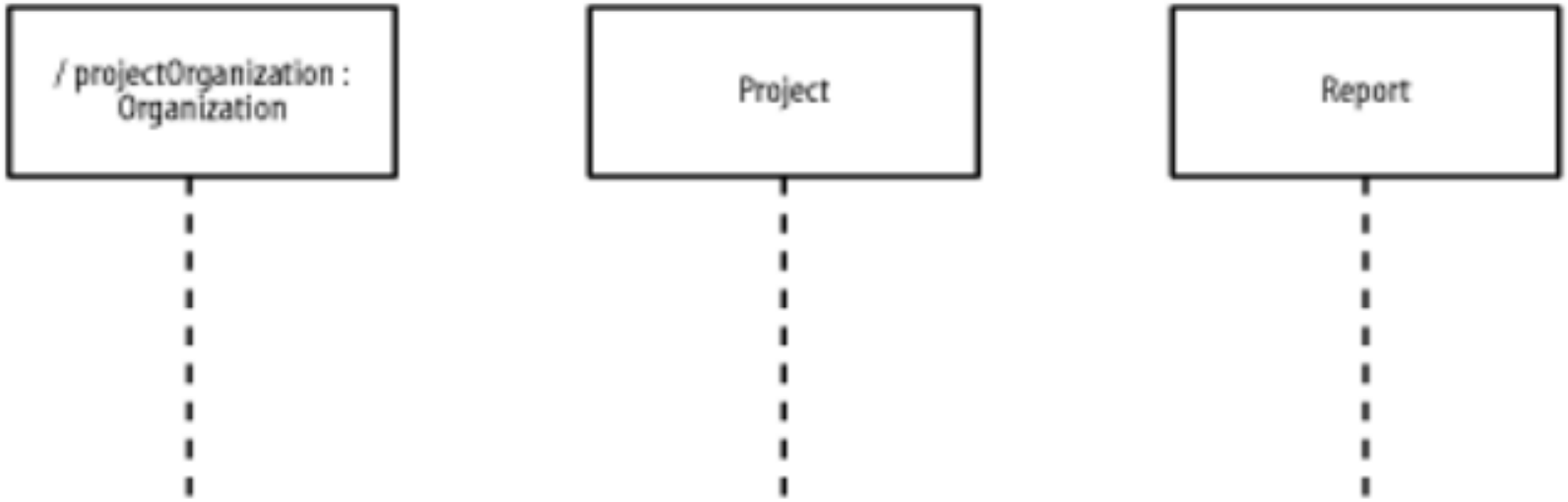


2.1 Object


triangle: Polygon
center = (0,0)
vertices = ((0,0),(4,0),(4,3))
borderColor = black
fillColor = white
(đối tượng có các giá trị thuộc tính cụ thể)
(một thể hiện đơn giản chỉ với tên của đối tượng)
triangle
:Polygon
(đối tượng ẩn danh, chỉ có tên lớp)
triangle: Polygon
(tên đối tượng)
(tên lớp)
(đối tượng với biểu tượng lớp phụ thuộc và tên được gạch dưới)
scheduler

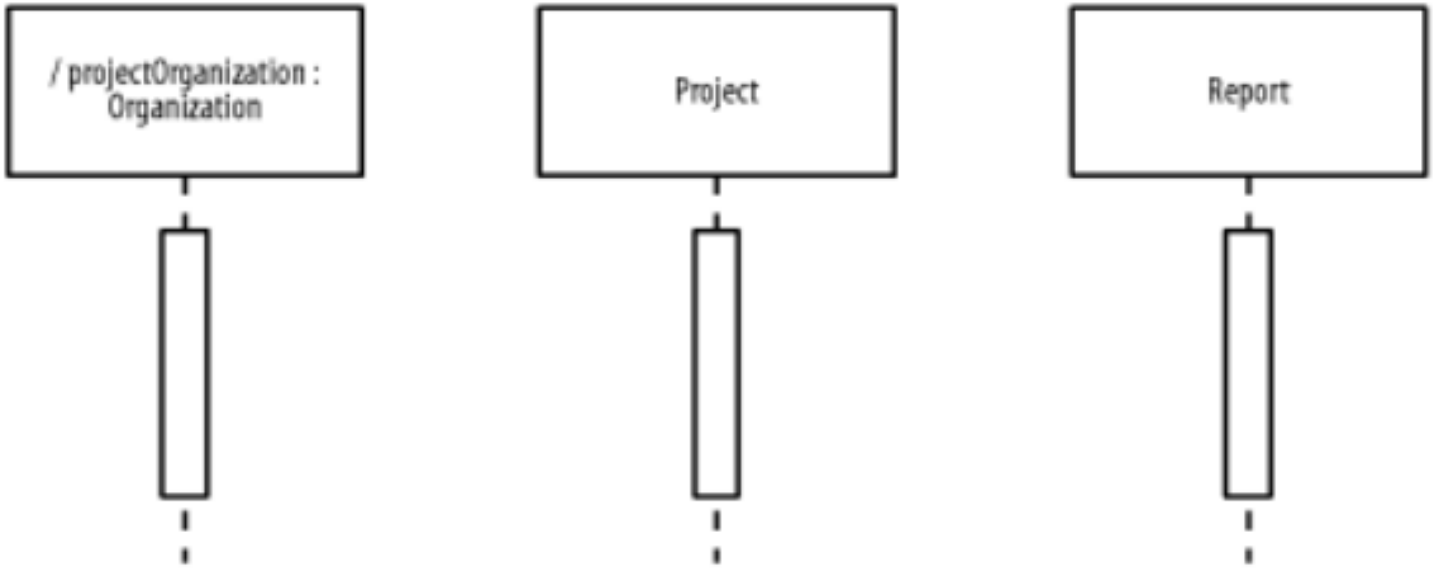
2.2 Lifeline

- Được thể hiện bằng một đường nét đứt dọc từ một object, đại diện cho sự tồn tại của các object theo thời gian.



2.3 Activations – Focus of control

- Hiển thị như một hình chữ nhật nằm trên lifeline đại diện cho các giai đoạn.



2.4 Message

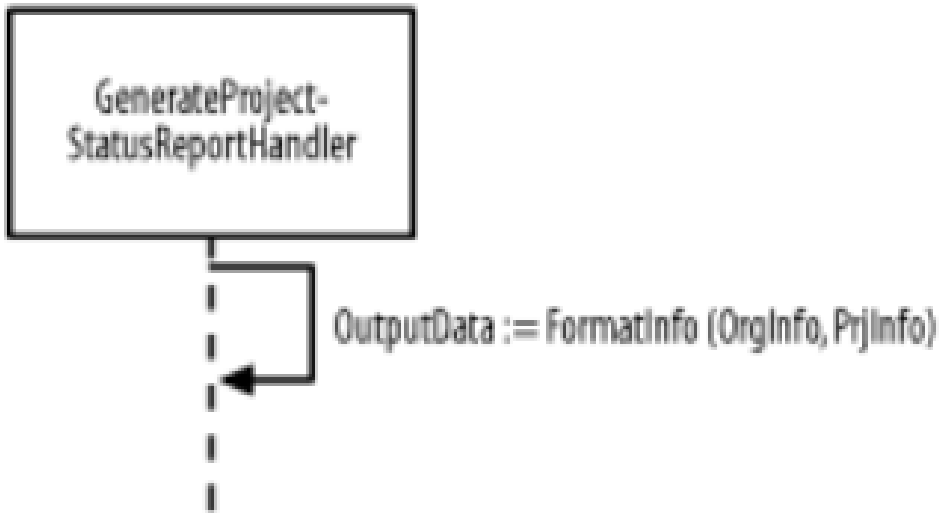
- Mô tả bằng cú pháp như sau:

[guard] * [iteration] *sequence number:*

return variable := operation_name
(argument list)

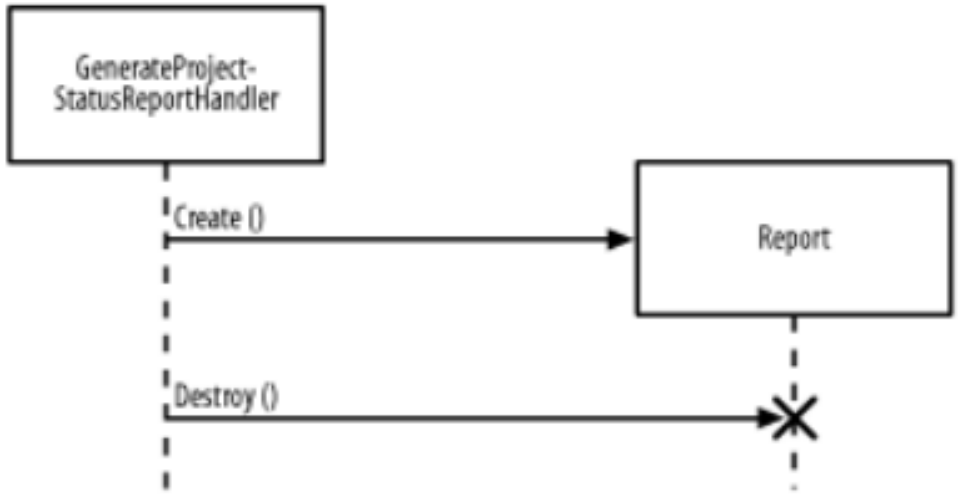
Reflexive message

- Được hiển thị như một mũi tên liền nét từ lifeline hoặc focus of control vòng quay trở lại lifelines hoặc focus of control đó



Message – Creation and Destruction

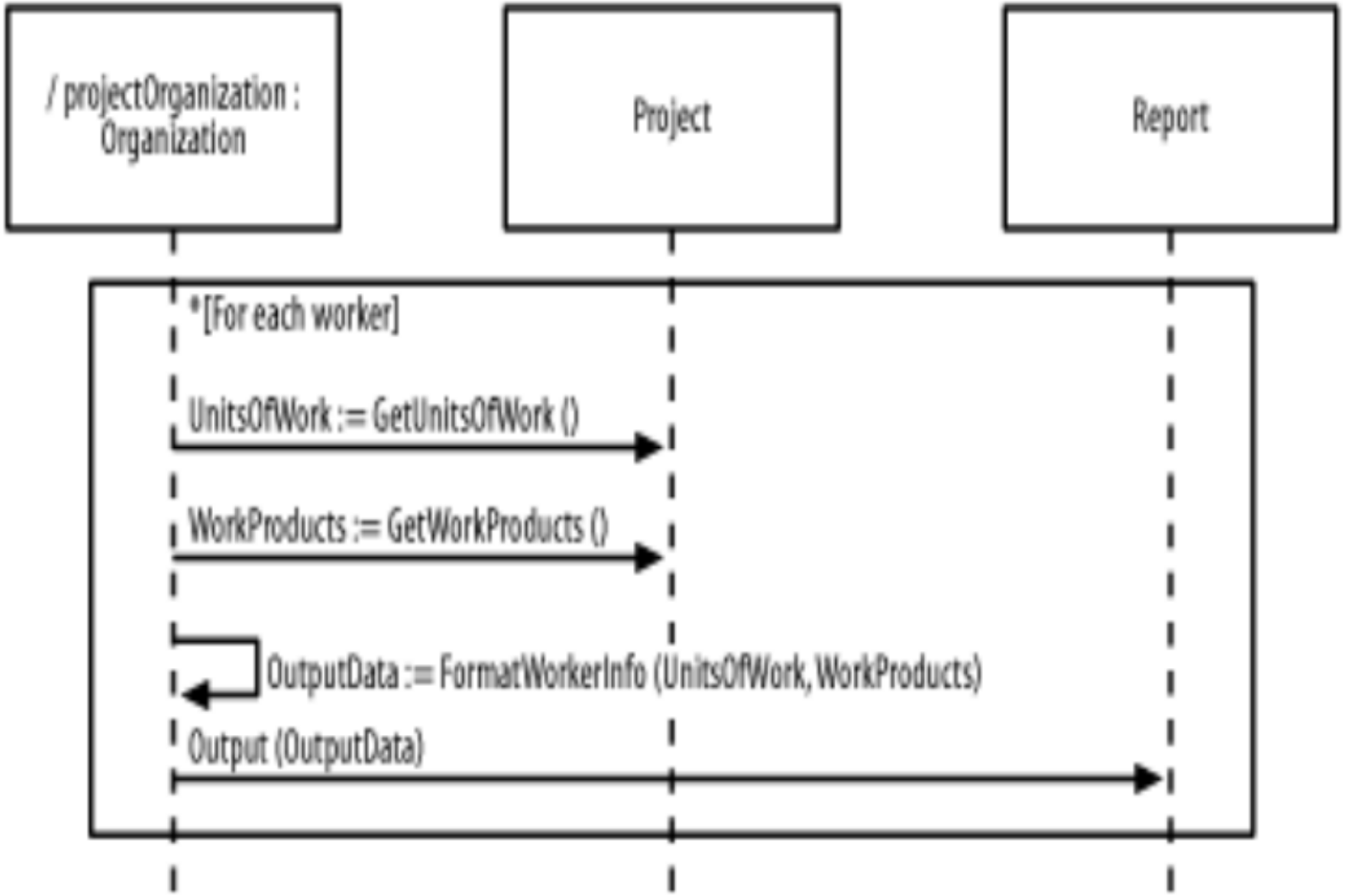
- Việc tạo một đối tượng được thể hiện bằng mũi tên thông điệp chỉ đến đối tượng cần tạo.
- Việc hủy một đối tượng được ký hiệu bằng một mũi tên thông điệp chỉ đến dấu "X" nằm cuối lifeline của đối tượng



Repetition

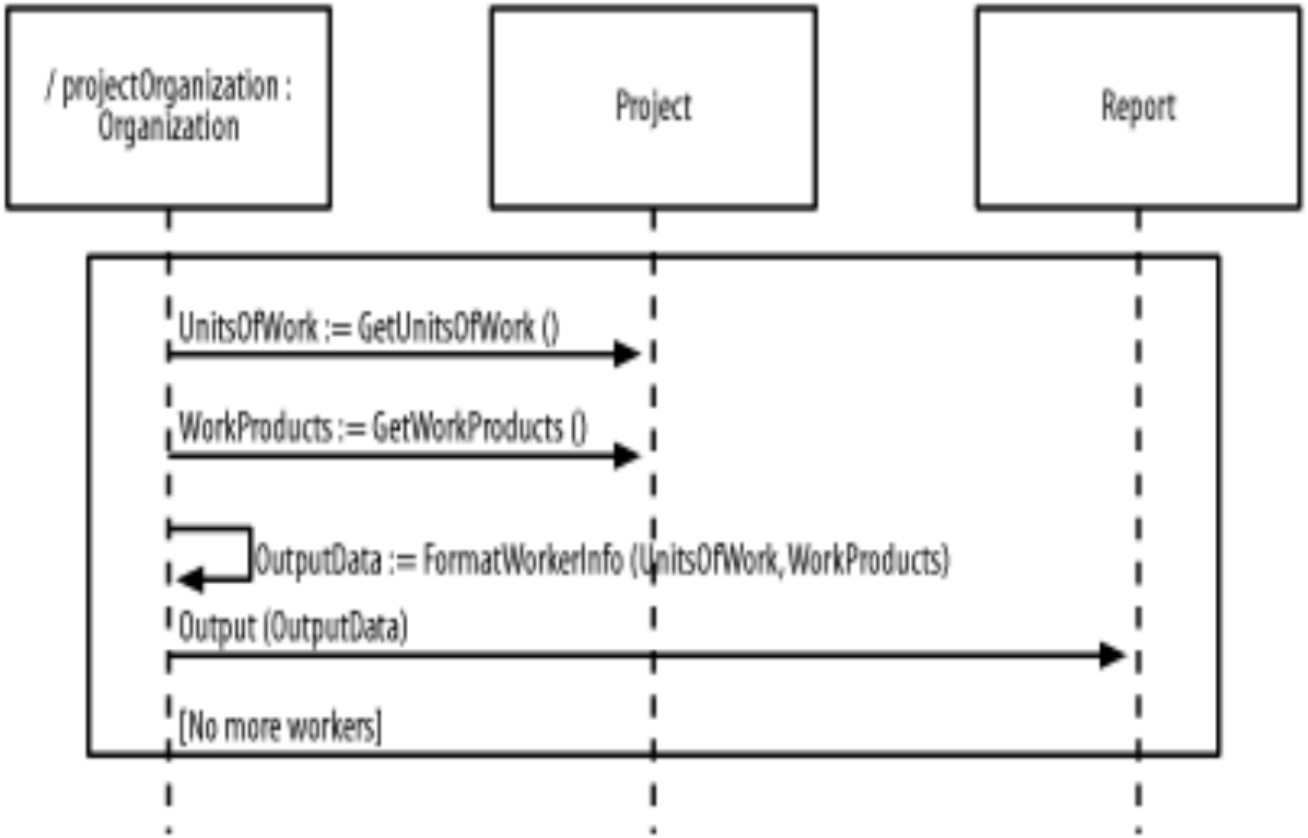
- Được thể hiện bằng thông tin kèm theo bên trong hình chữ nhật.
- Được thể hiện ở góc trên hay góc bên dưới của hình chữ nhật cho biết số lần các thông tin liên lạc với nhau bên trong hình chữ nhật.

Repetition


/ projectOrganization : Organization
Project
Report
*[For each worker]
UnitsOfWork := GetUnitsOfWork ()
WorkProducts := GetWorkProducts ()
OutputData := FormatWorkerInfo (UnitsOfWork, WorkProducts)
Output (OutputData)

Repetition

- Một biểu hiện Guard được sử dụng ở góc bên trên hay bên dưới của hình chữ nhật chỉ ra điều kiện được chấp nhận để chấm dứt sự lặp lại

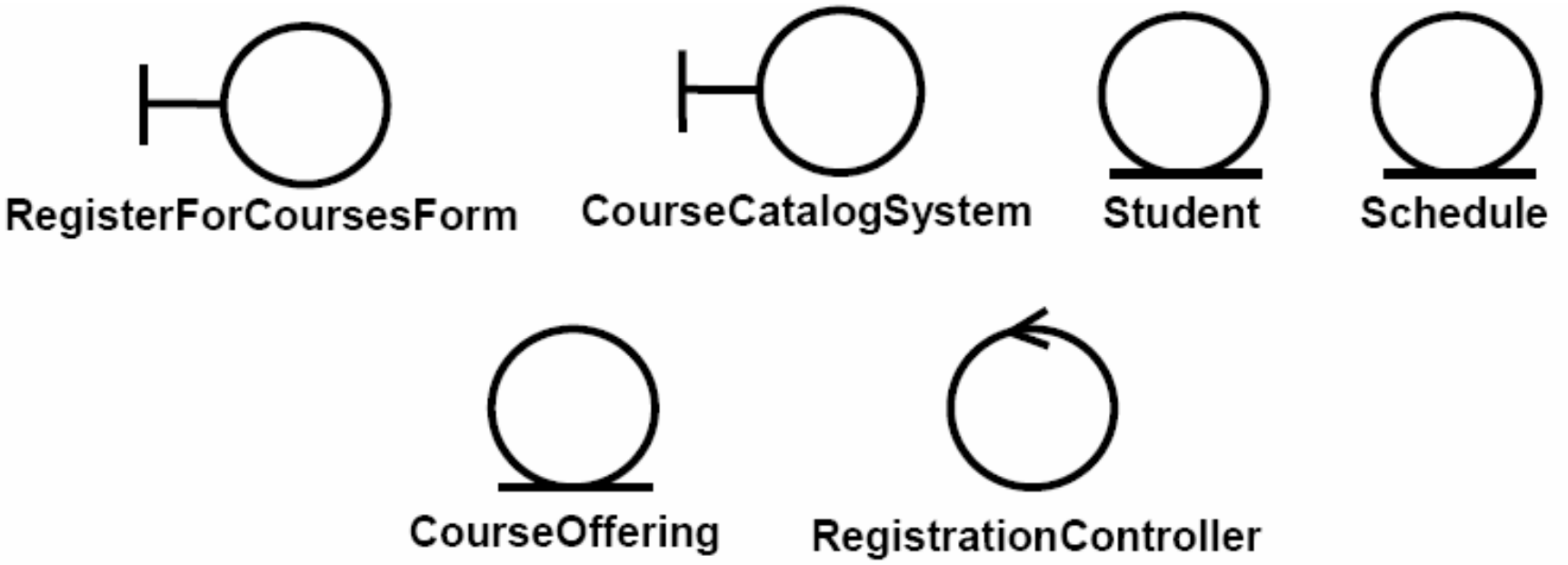


Use Case Đăng ký học phần

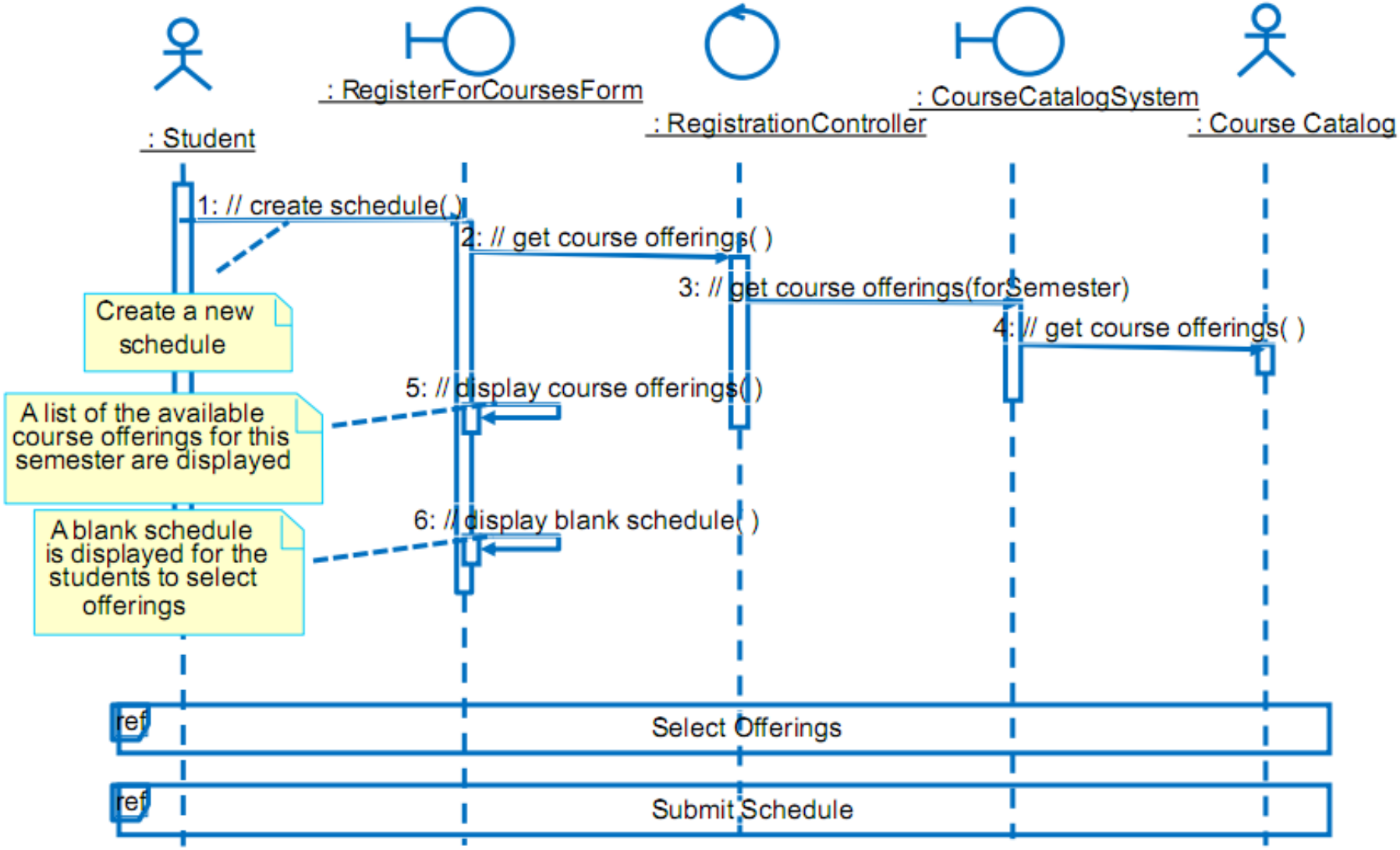
Basic flow: Create a Schedule

1. SV chọn chức năng "create schedule."
2. Hệ thống yêu cầu danh sách các lớp được mở (course offerings) từ Course Catalog System.
3. Hệ thống hiện thị giao diện thời khóa biểu trống.
4. SV chọn 4 học phần chính và 2 học phần thay thế từ danh sách các học phần được mở. Khi việc lựa chọn hoàn tất, SV chọn "submit."
5. Hệ thống tạo một lịch học chứa những học phần sinh viên đã đăng ký.
6. Sinh viên kiểm tra và xác nhận lịch học, Submit Schedule được thực thi.

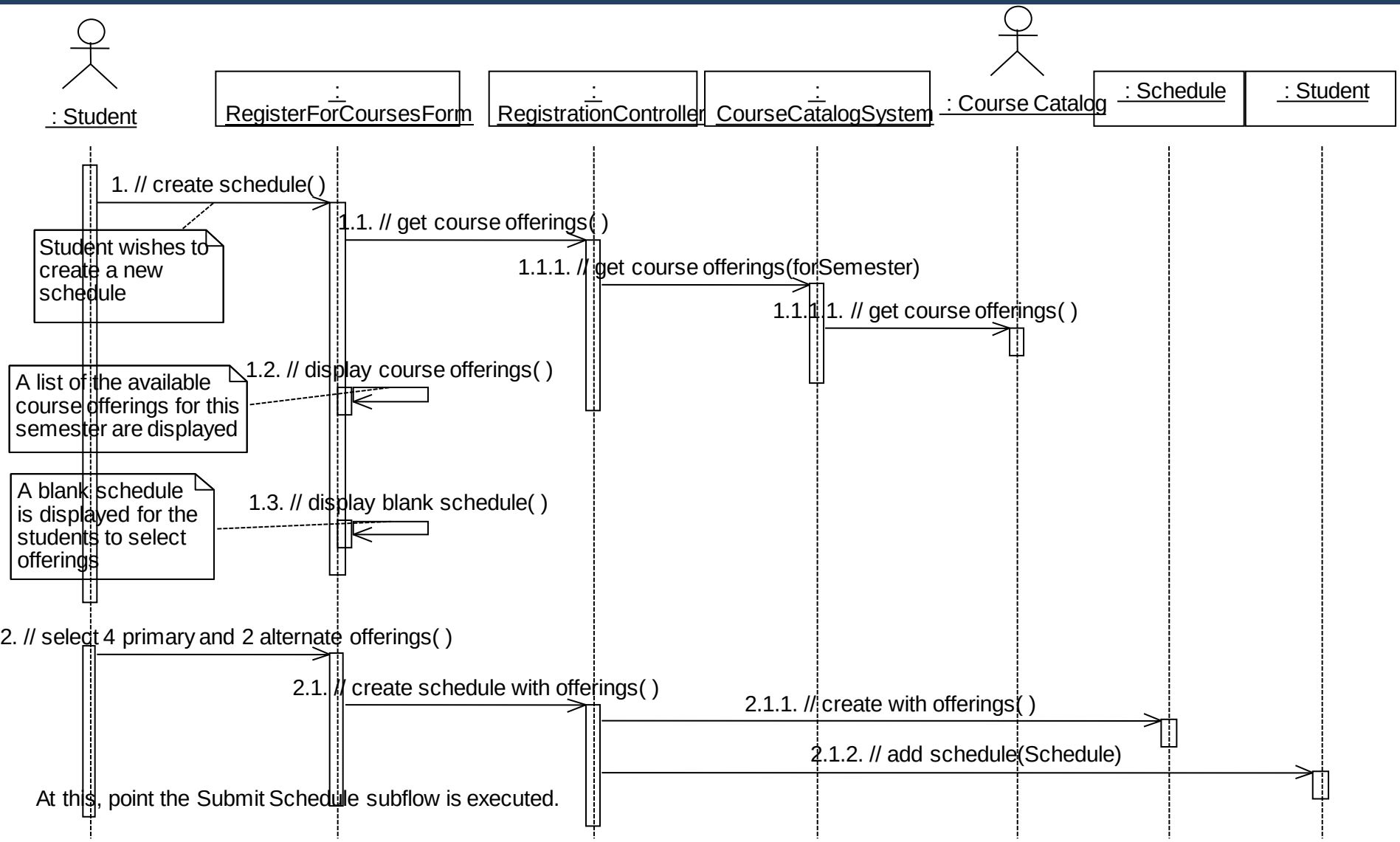
Analysis Classes



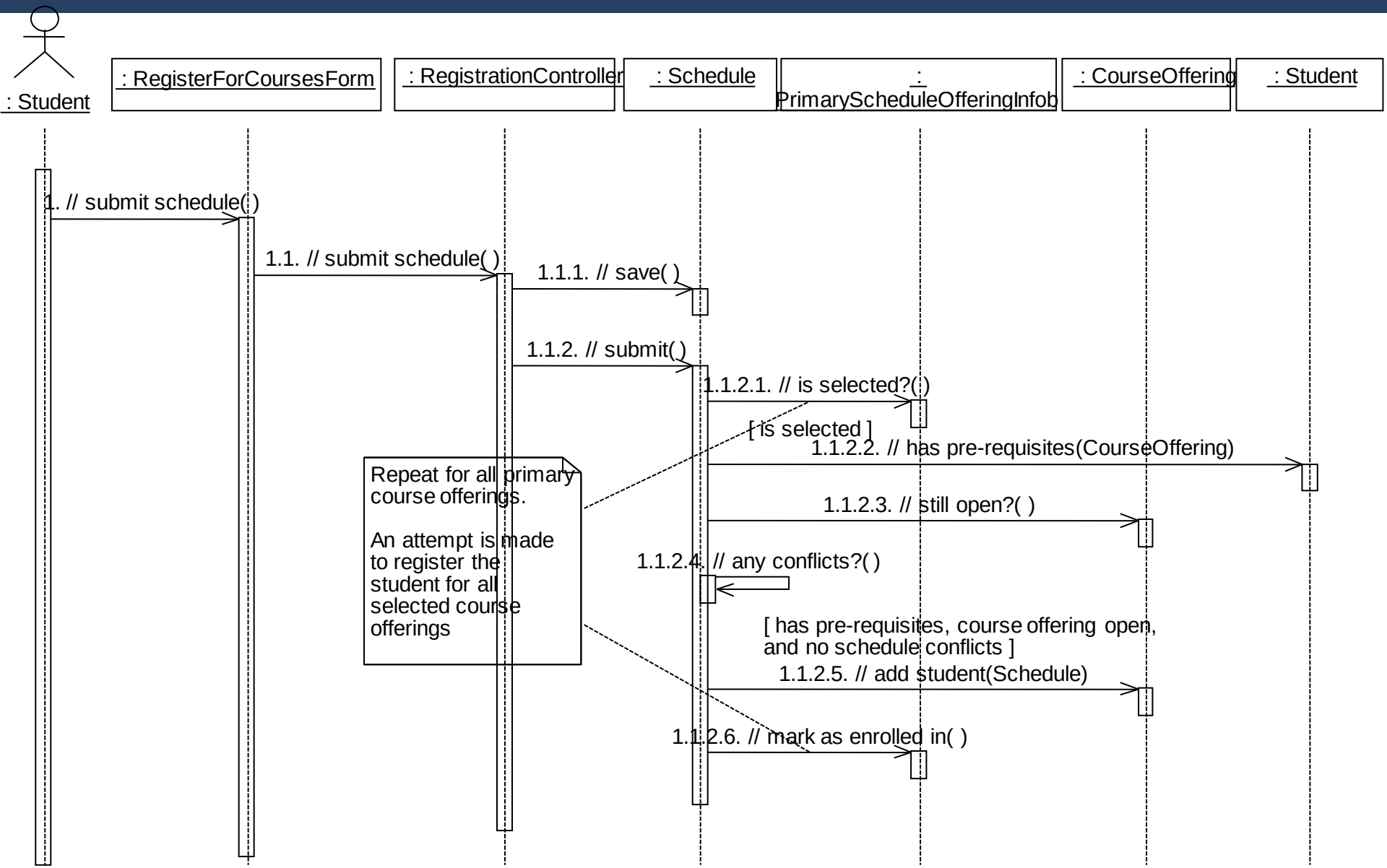
Ví dụ: Sequence Diagram



Ví dụ : Sequence Diagram

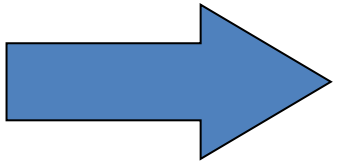


Ví dụ: Sequence Diagram (cont.)



Ví dụ: Use case Rút tiền

Xác định các danh từ



KH
User
Tài khoản
Số tiền rút
Thẻ

Use case: Rút tiền
Actor: Khách hàng
Mô tả: Cho phép khách hàng rút tiền
Pre-conditions: User **phải login**
Post-conditions: **Tài khoản của user bị trừ ứng với số tiền rút**
Main flow:

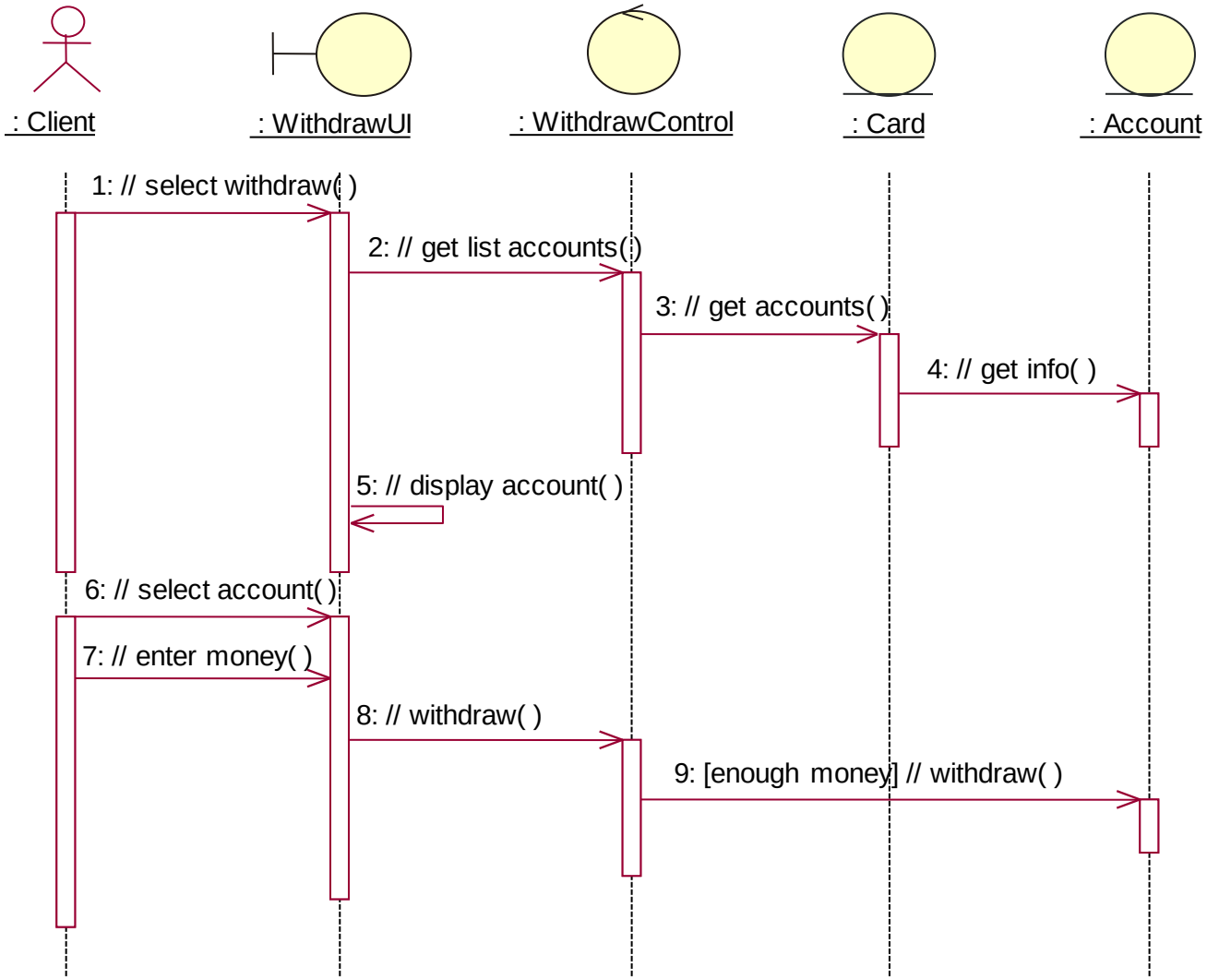
1) Usecase bắt đầu khi **KH** chọn "Rút tiền"
2) HT hiển thị tất cả **tài khoản** tương ứng với **thẻ** và yêu cầu KH chọn một tài khoản.
3) KH chọn một tài khoản.
4) HT yêu cầu nhập **số tiền muốn rút**
5) KH nhập số tiền muốn rút.

…

Alternative flows:

…

Sequence Diagram – Use case Rút tiền



Nội dung

1. Xác định class
2. Biểu đồ trình tự
3. Biểu đồ cộng tác
   - Định nghĩa
   - Chức năng
   - Cấu trúc và các thành phần

3. Biểu đồ cộng tác Collaboration Diagram

- Mục đích: biểu diễn mối quan hệ tương tác giữa
  - Các đối tượng
  - Đối tượng và tác nhân
  - Nhấn mạnh đến vai trò của các đối tượng trong tương tác
  - Biểu diễn các hoạt động như các luồng công việc hoặc tiến trình khác nhau trong hệ thống được xây dựng.

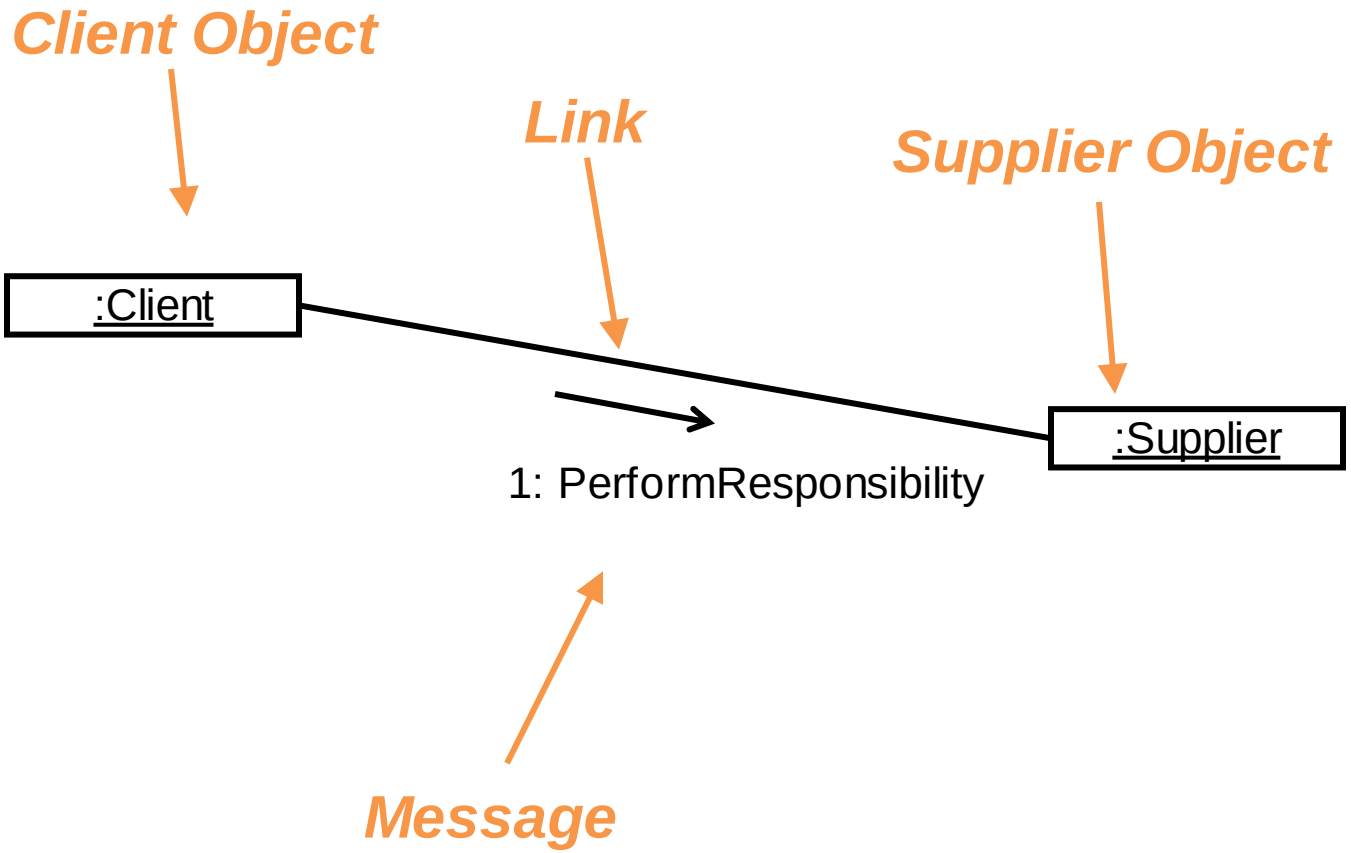
3. Collaboration Diagram

- Đặc điểm:
  - Đối tượng được đặt một cách tự do trong không gian
  - Không có đường lifeline cho mỗi đối tượng
  - Các message được đánh số theo thứ tự thời gian

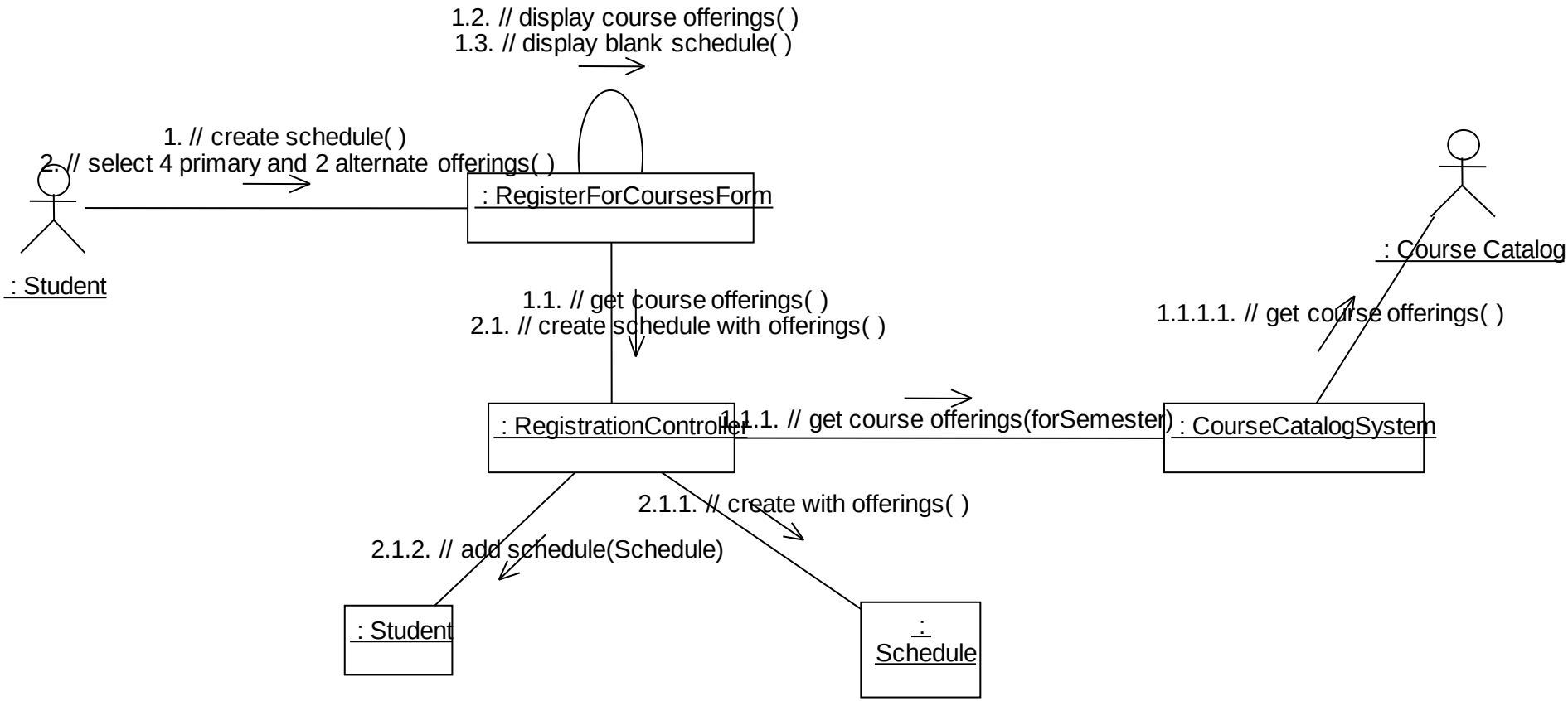
3. Collaboration Diagram

- Các thành phần
  - ***Đối tượng (participants):*** hình chữ nhật, luôn xuất hiện tại vị trí xác định, tên đối tượng:tên lớp
  - ***Liên kết (links):*** đường nối 2 đối tượng có tương tác, không có chiều
  - ***Thông điệp (messages):*** mũi tên từ đối tượng gửi sang đối tượng nhận, đánh số theo thứ tự xuất hiện

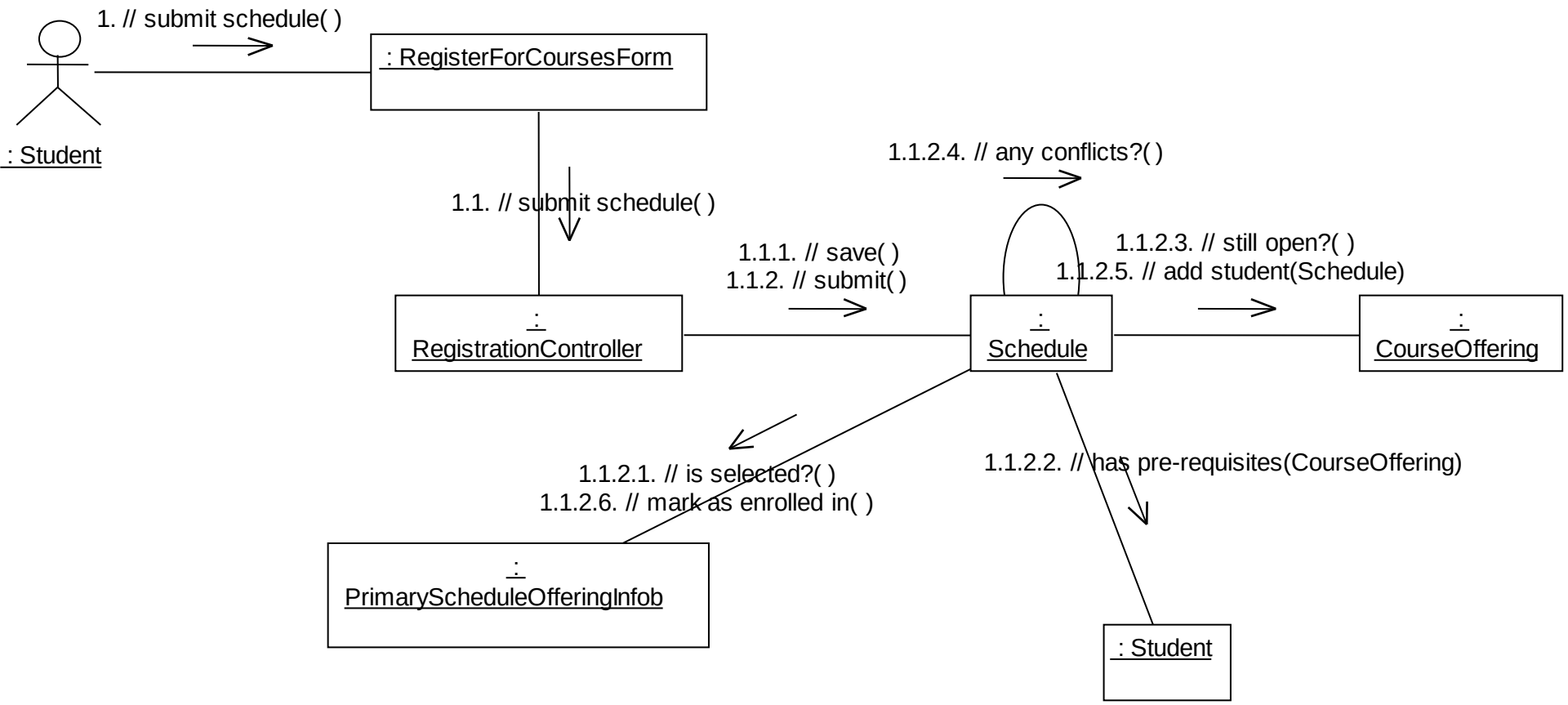
3. Collaboration Diagram



Ví dụ: Collaboration Diagram

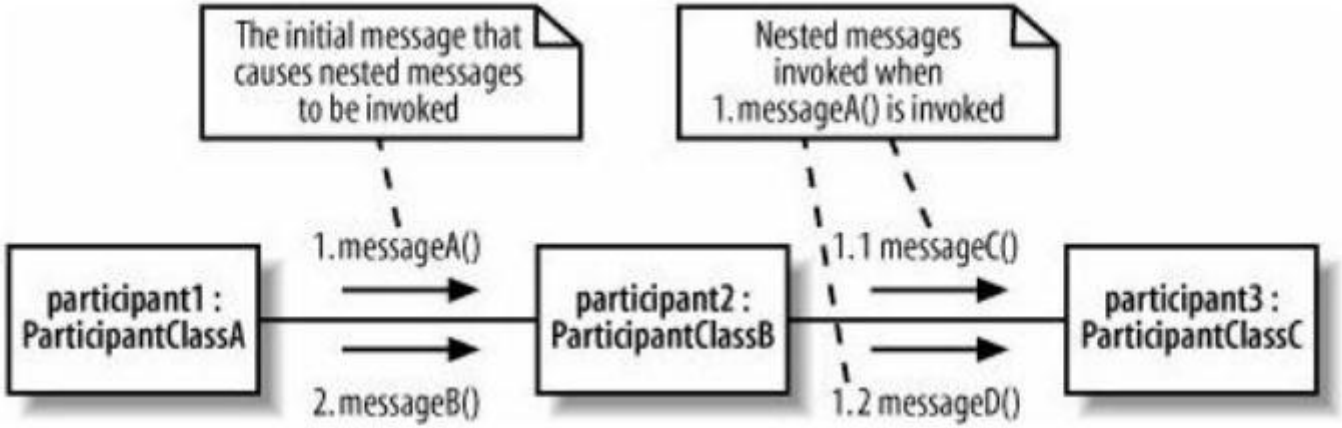


Ví dụ: Collaboration Diagram



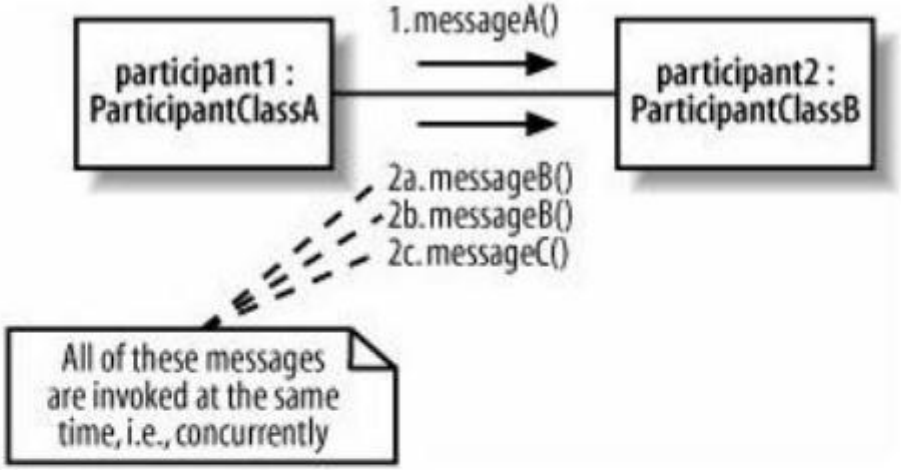
Các loại thông điệp

- Thông điệp lồng nhau (nested messages): là thông điệp được gọi (invoke) bởi đối tượng theo yêu cầu của 1 thông điệp trước đó



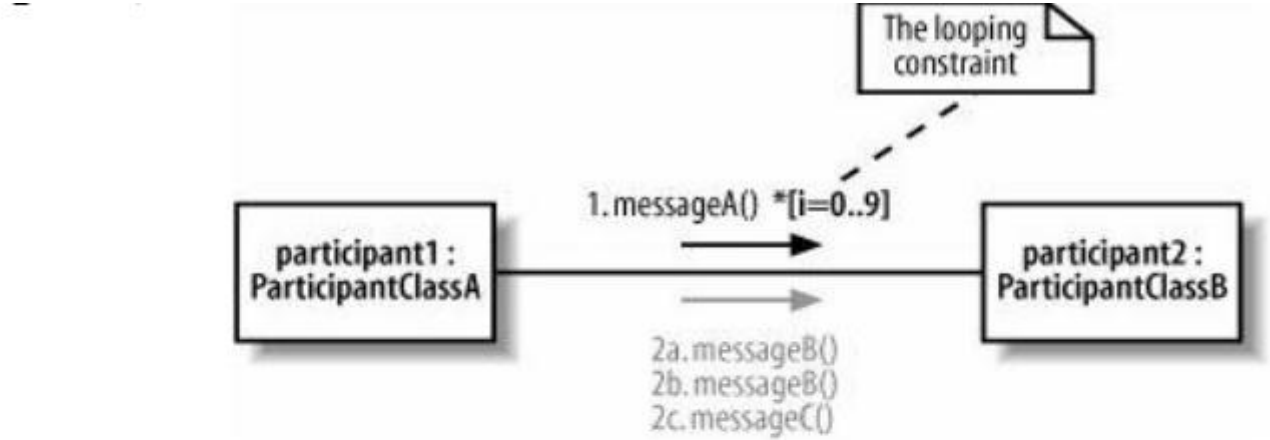
Các loại thông điệp

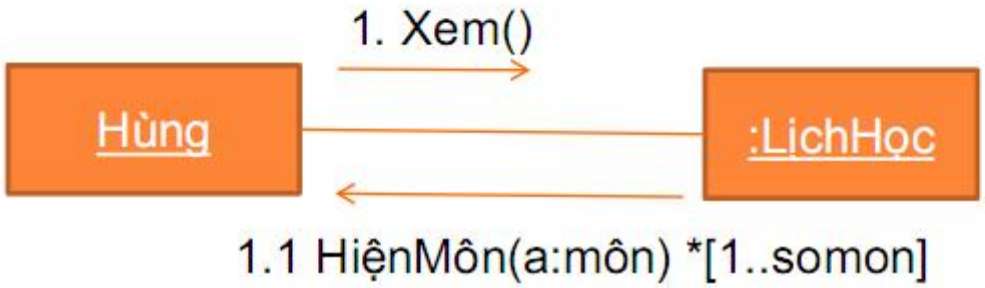
- Thông điệp xảy ra đồng thời: ký hiệu bằng cách thêm ký tự (a,b,c,…) sau số chỉ thứ tự thực hiện của nó.



Các loại thông điệp

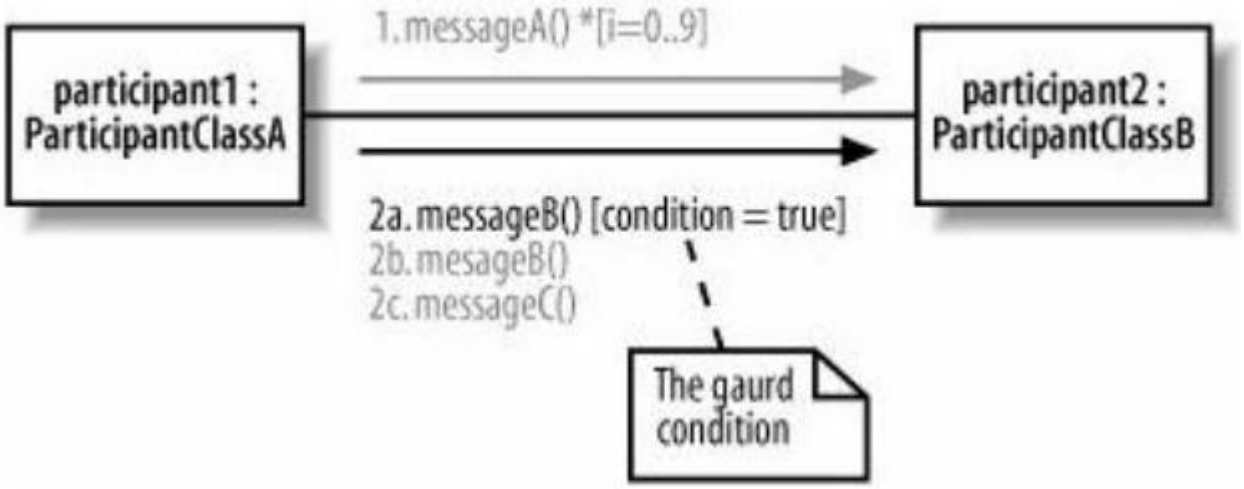
- Gọi một thông điệp nhiều lần (vd: thông điệp được gọi bởi vòng for)

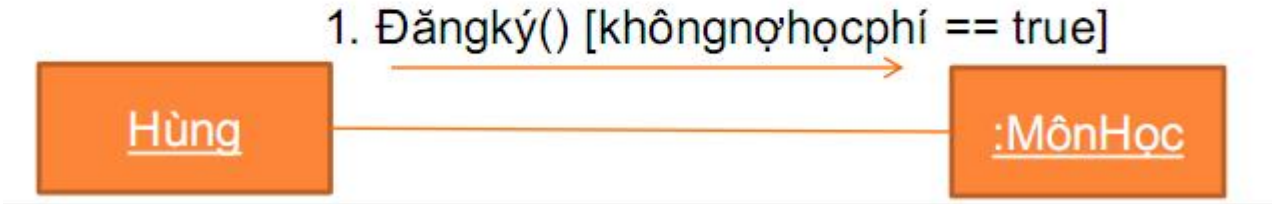




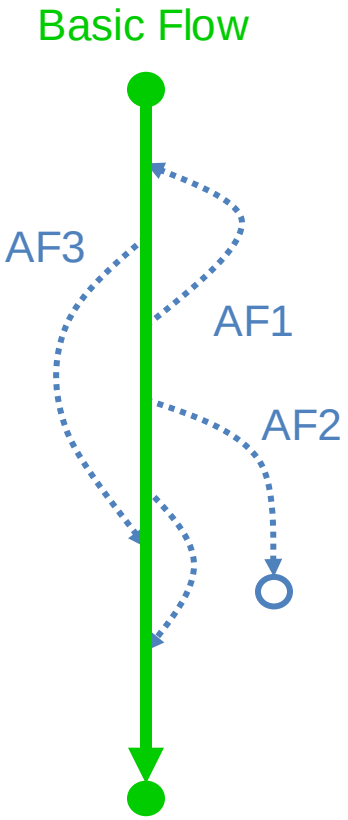
Các loại thông điệp

- Thông điệp chỉ được thực hiện khi thỏa 1 điều kiện cụ thể

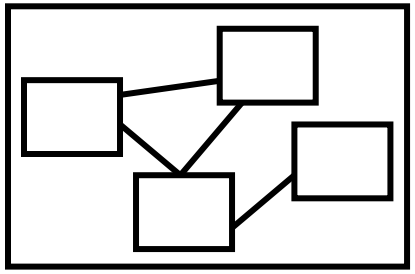




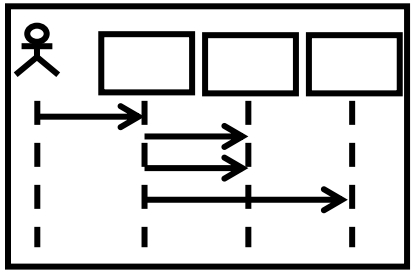
Một lược đồ tương tác là chưa đủ



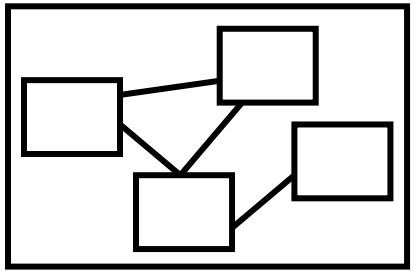
Alternate Flow 1



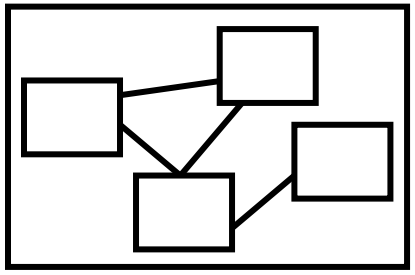
Alternate Flow 2



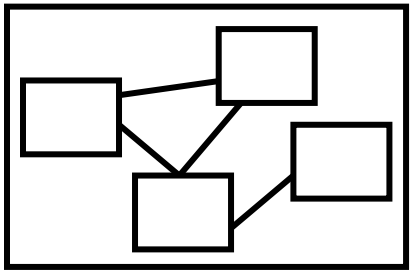
Alternate Flow 3



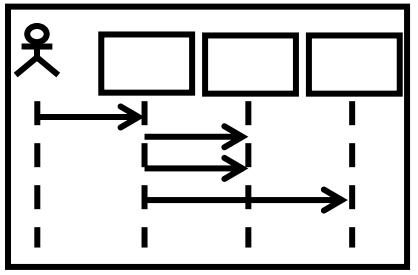
Alternate Flow 4



Alternate Flow 5



Alternate Flow n



Collaboration Diagrams Vs Sequence Diagrams

- Collaboration Diagrams
  - Chỉ ra các mối quan hệ bổ sung cho các tương tác
  - Trực quan hóa tốt hơn các mẫu cộng tác
  - Trực quan hóa tốt hơn các hiệu ứng tác động lên một đối tượng cụ
  - Dễ sử dụng hơn trong các vấn đề cần giải quyết tập thể

- Sequence Diagrams
  - Chỉ ra rõ ràng chuỗi các thông điệp
  - Trực quan hóa tốt hơn toàn bộ luồng sự kiện
  - Tốt hơn cho các đặc tả real-time và cho các scenario phức tạp