Bài 13

Tổng quan về UML

Nội dung

1. Phân tích và thiết kế hệ thống HĐT
2. Biểu đồ use case
3. Biểu đồ hoạt động
4. Biểu đồ tương tác
5. Biểu đồ lớp

1

Phân tích và thiết kế hệ thống hướng đối tượng

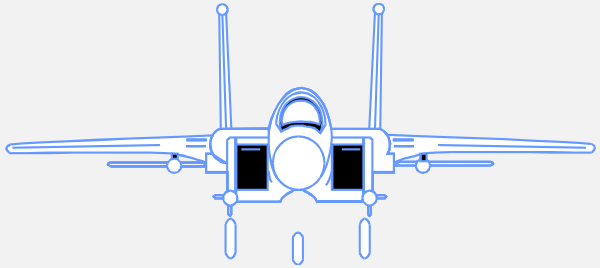
Object-oriented analysis and design (OOAD)

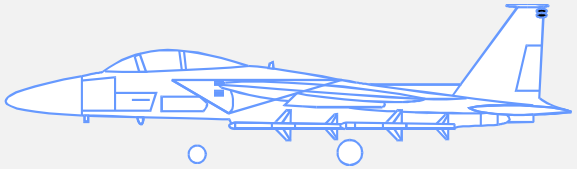
Mô hình hóa

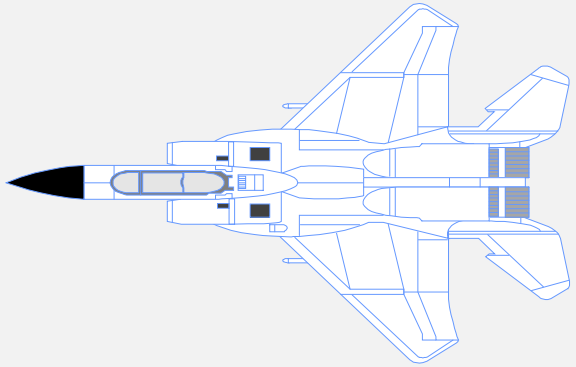
- Hướng tiếp cận "máy bay giấy"?
- Đối với dự án phần mềm
  - Mất rất nhiều thời gian và tạo ra rất nhiều mã nguồn.
  - Không có bất kỳ một kiến trúc nào.
  - Phải chịu khổ với những lỗi phát sinh.

Mô hình hóa

- Mô hình hóa
  - Giúp đơn giản hóa thế giới thực bằng các mô hình
  - Giúp hiểu rõ hơn về hệ thống dướI các góc nhìn khác nhau







UML

- Ngôn ngữ mô hình hóa thống nhất (Unified Modeling Language - UML)
- UML là ngôn ngữ để:
  - trực quan hóa (visualizing)
  - đặc tả (specifying)
  - xây dựng (constructing)
  - tài liệu hóa (documenting)

  các cấu phần (artifact) của một hệ thống phần mềm

UML

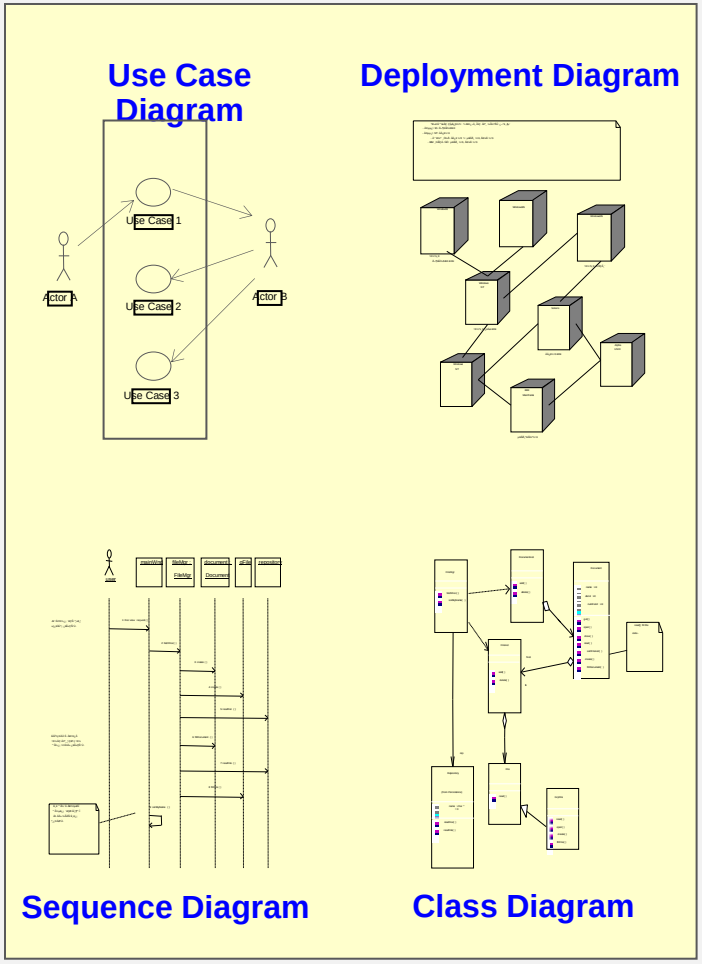
- UML là ngôn ngữ trực quan
- Giúp công việc phát triển được xử lý nhất quán, giảm thiểu lỗi xảy ra
  - Giúp dễ hình dung hơn cấu trúc của hệ thống
  - Hiệu quả hơn trong việc liên lạc, trao đổi
    + Trong tổ chức
    + Bên ngoài tổ chức

UML

- Các mô hình UML có thể kết nối trực tiếp với rất nhiều ngôn ngữ lập trình.
  - Ánh xạ sang Java, C++, Visual Basic…
  - Các bảng trong RDBMS hoặc kho lưu trữ trong OODBMS
  - Cho phép các kỹ nghệ xuôi (chuyển UML thành mã nguồn)
  - Cho phép kỹ nghệ ngược (xây dựng mô hình hệ thống từ mã nguồn)

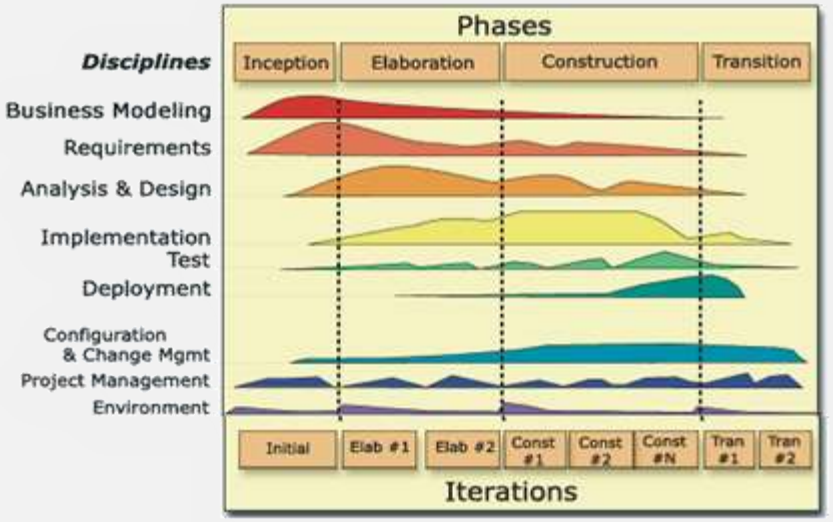
UML

- UML là ngôn ngữ tài liệu hóa
- Tài liệu hóa kiến trúc, yêu cầu, kiểm thử, lập kế hoạch dự án, và quản lý việc bàn giao phần mềm
- Các biểu đồ khác nhau, các ghi chú, ràng buộc được đặc tả trong tài liệu



UML

- UML là ký pháp chứ không phải là phương pháp
  - UML có thể áp dụng cho tất cả các pha của quy trình phát triển phần mềm
  - "Rational Unified Process" - quy trình phát triển cho UML



Lịch sử phát triển

- Vào 1994, có hơn 50 phương pháp mô hình hóa hướng đối tượng:
  - Fusion, Shlaer-Mellor, ROOM, Class-Relation,Wirfs-Brock, Coad-Yourdon, MOSES, Syntropy, BOOM, OOSD, OSA, BON, Catalysis, COMMA, HOOD, Ooram, DOORS …
- "Meta-models" tương đồng với nhau
- Các ký pháp đồ họa khác nhau
- Quy trình khác nhau hoặc không rõ ràng

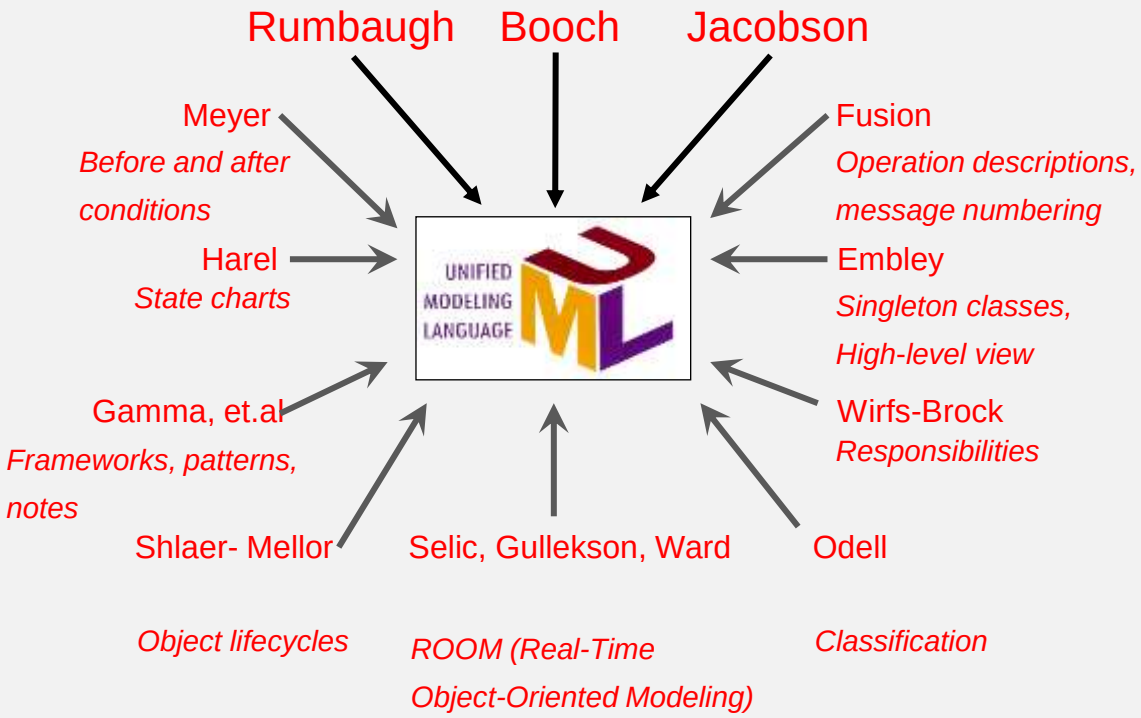
→ Cần chuẩn hóa và thống nhất các phương pháp

Lịch sử phát triển của UML

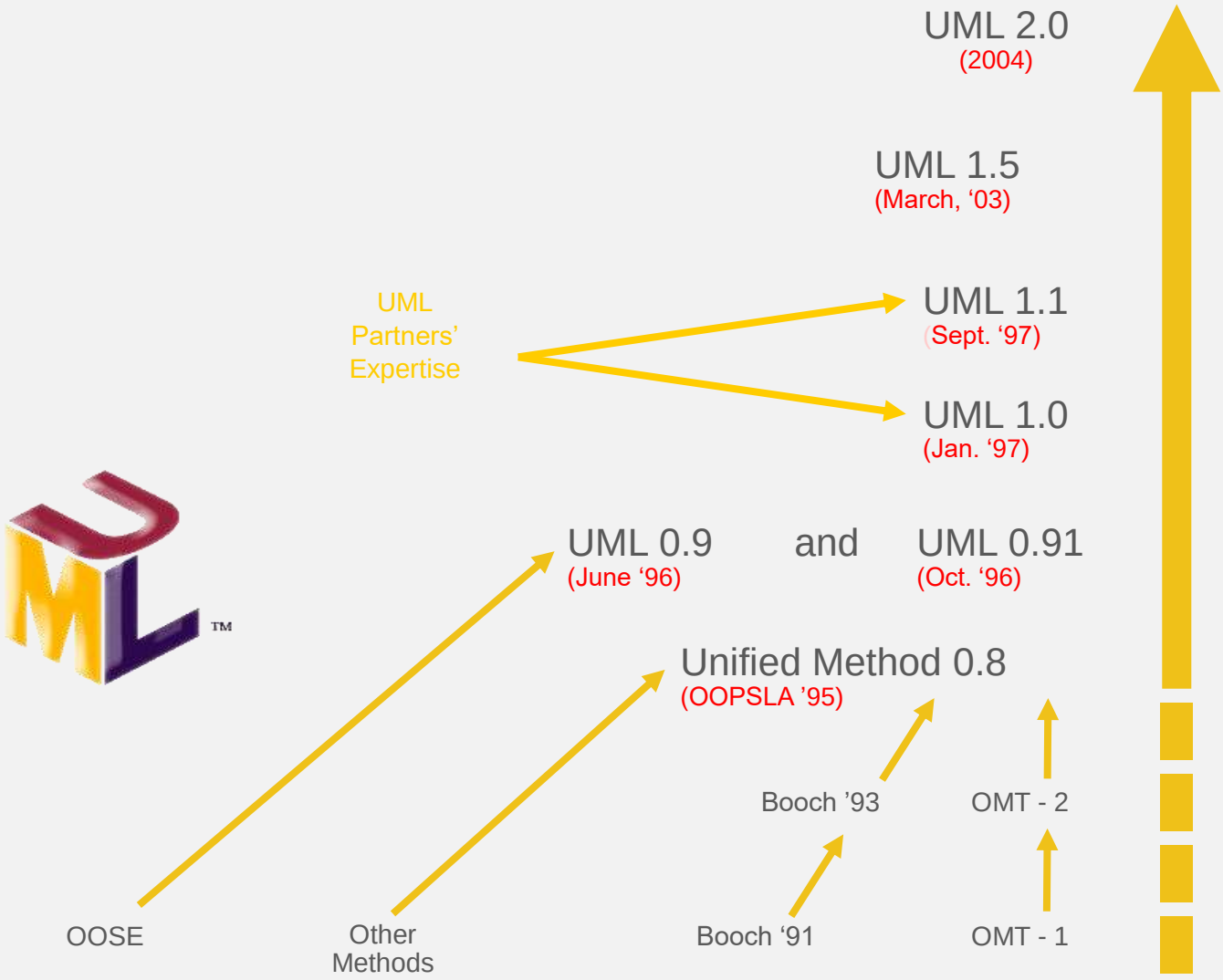
- UML được 3 chuyên gia hướng đối tượng hợp nhất các kỹ thuật của họ vào năm 1994:
  - Booch91 (Grady Booch): Conception, Architecture
  - OOSE (Ivar Jacobson): Use cases
  - OMT (Jim Rumbaugh): Analysis
- Thiết lập một phương thức thống nhất để xây dựng và "vẽ" ra các yêu cầu và thiết kế hướng đối tượng trong quá trình PTTK phần mềm → UML được công nhận là chuẩn chung vào năm 1997.

Lịch sử phát triển của UML

UML là ngôn ngữ hợp nhất các mô hình khác nhau



Lịch sử phát triển của UML



Mục đích của OOAD

- Chuyển các yêu cầu của bài toán thành một bản thiết kế của hệ thống sẽ được xây dựng
- Tập trung vào quá trình phân tích các YÊU CẦU của hệ thống và thiết kế các MÔ HÌNH cho hệ thống đó trước giai đoạn lập trình
- Được thực hiện nhằm đảm bảo mục đích và yêu cầu của hệ thống được ghi lại một cách hợp lý trước khi hệ thống được xây dựng
- Cung cấp cho người dùng, khách hàng, kỹ sư phân tích, thiết kế nhiều cái nhìn khác nhau về cùng một hệ thống

Các công cụ UML

- Công cụ mã nguồn mở:
  - EclipseUML
  - UmlDesigner
  - StarUML
  - Argo UML...
- Công cụ thương mại:
  - Enterprise Architect
  - IBM Rational Software Architect
  - Microsoft Visio
  - Visual Paradigm for UML
  - SmartDraw...

Các biểu đồ UML

- Biểu đồ use case (Use Case Diagram)
- Biểu đồ hoạt động (Activity Diagram)
- Biểu đồ tương tác (Interaction Diagrams)
  - Biểu đồ trình tự (Sequence Diagram)
  - Biểu đồ giao tiếp/cộng tác (Communication/Collaboration Diagram)
- Biểu đồ trạng thái (Statechart Diagram)
- Biểu đồ cấu trúc tĩnh (Static Structure Diagrams)
  - Biểu đồ lớp (Class Diagram)
  - Biểu đồ đối tượng (Object Diagram)
- Biểu đồ thực thi (Implementation Diagrams)
  - Biểu đồ thành phần (Component Diagram)
  - Biểu đồ triển khai (Deployment Diagram)

2

Biểu đồ use case

Use case diagram

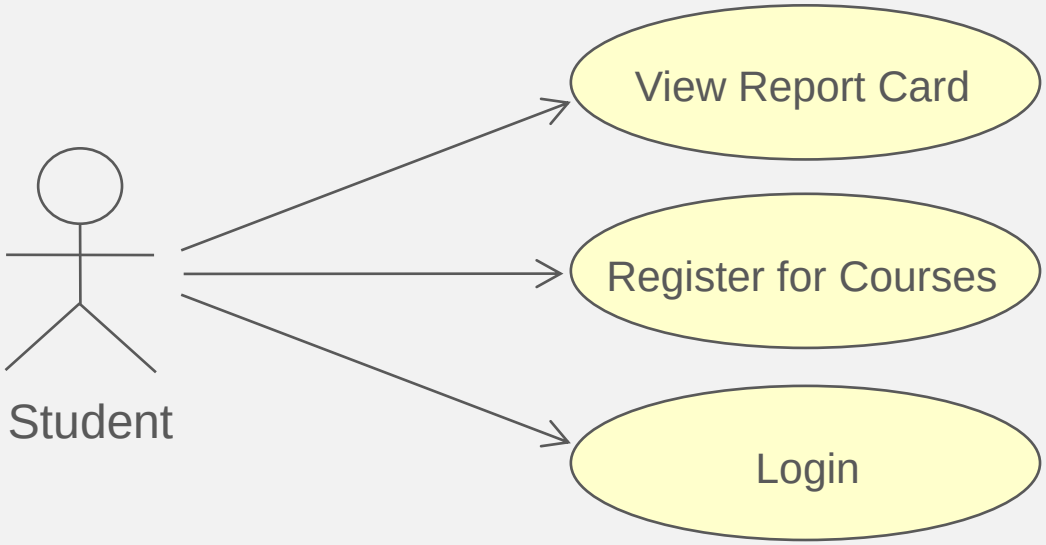
Tổng quan

- Mỗi hệ thống tương tác với con người hoặc các hệ thống khác để thực hiện nhiệm vụ
- Các hành vi của hệ thống có thể được mô tả trong các use case.
  - What, not How
  - Các use case mô tả các tương tác giữa hệ thống và môi trường của nó

    → Biểu đồ use case

Tổng quan về biểu đồ use case

- Biểu đồ mô tả các yêu cầu chức năng của hệ thống dưới dạng các use case.
- Bao gồm các chức năng mong đợi của hệ thống (use case) và môi trường (actor) của nó.

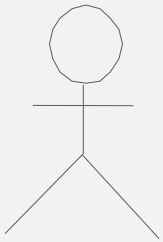


Mục đích

- Giống như một bản hợp đồng giữa người phát triển phần mềm và khách hàng.
- Là công cụ mạnh mẽ cho việc lập kế hoạch → Được dùng trong tất cả các giai đoạn trong quy trình phát triển hệ thống
  - Khách hàng phê chuẩn biểu đồ use-case
  - Sử dụng biểu đồ use case để thảo luận với khách hàng.
  - Các thành viên tham gia vào dự án, sử dụng mô hình này để hiểu rõ hơn về hệ thống

Các thành phần chính

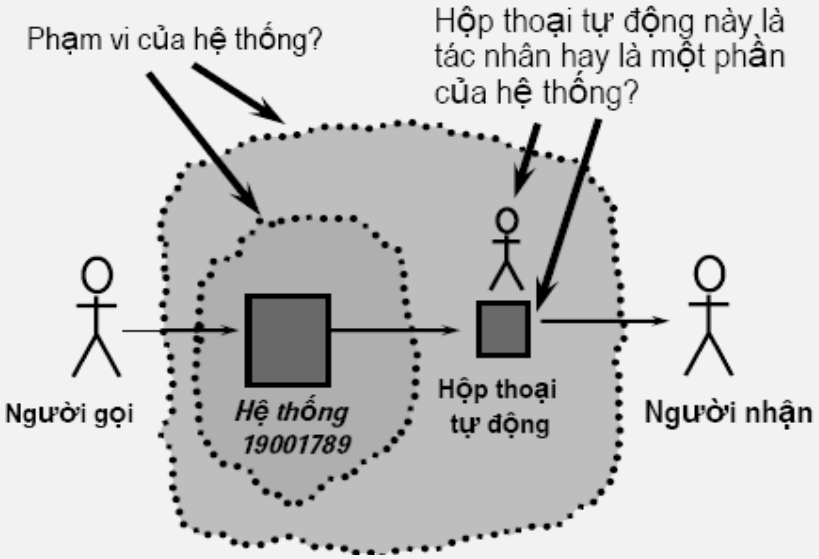
- Tác nhân
  - là bất kỳ thứ gì tương tác với hệ thống, có sự trao đổi dữ liệu với hệ thống, có thể là:
    + Người dùng,
    + Thiết bị phần cứng
    + Hệ thống phần mềm khác
  - Là một lớp/loại người dùng chứ không phải một người cụ thể
  - Một người dùng cụ thể có thể đóng vai trò là các tác nhân khác nhau, có nghĩa là người đó có nhiều vai trò khác nhau trong hệ thống
  - Không phải là một phần của hệ thống



Ví dụ

- Tác nhân trao đổi thông tin với hệ thống:
  - Gửi thông tin tới hệ thống
  - Nhận thông tin từ hệ thống
- Tác nhân KHÔNG phải là một phần của hệ thống

- Ví dụ: Hệ thống trả lời tự động

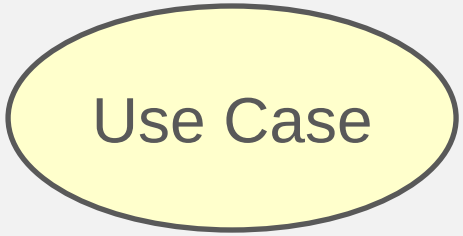


Xác định tác nhân của hệ thống

- Đặt các câu hỏi sau
  - Nhóm người nào yêu cầu hệ thống làm việc giúp họ?
  - Nhóm người nào kích hoạt chức năng của hệ thống?
  - Nhóm người nào sẽ duy trì và quản trị hệ thống hoạt động?
  - Hệ thống có tương tác với các thiết bị hay phần mềm ngoại vi nào khác hay không?
- Thông tin về tác nhân
  - Tên tác nhân phải mô tả vai trò của tác nhân đó một cách rõ ràng
  - Tên nên là danh từ
  - Cần mô tả khái quát khả năng của tác nhân đó

Các thành phần chính (tiếp)

- Use case
  - Mô tả chức năng của hệ thống, là một chuỗi các hành động của hệ thống thực hiện nhằm thu được một kết quả dễ thấy tới một tác nhân nào đó.
  - Một use case mô hình hóa một hội thoại giữa một hoặc nhiều tác nhân với hệ thống
  - Một use case mô tả hành động của hệ thống thực hiện nhằm mang đến một giá trị nào đó cho tác nhân.



Xác định use case của hệ thống

- Xem các yêu cầu chức năng
- Đối với mỗi tác nhân tìm được, đặt các câu hỏi:
  - Các tác nhân yêu cầu những gì từ hệ thống
  - Các công việc chính mà tác nhân đó muốn HT thực thi?
  - Tác nhân đó có tạo ra hay thay đổi dữ liệu gì của HT?
  - Tác nhân đó có phải thông báo gì cho HT?
  - Tác nhân đó có cần thông tin thông báo gì từ HT?
- Thông tin về use case
  - Tên của UC nên chỉ rõ kết quả của quá trình tương tác với tác nhân
  - Tên nên là động từ
  - Mô tả ngắn gọn về mục đích của UC

Những điều nên tránh khi tạo UC

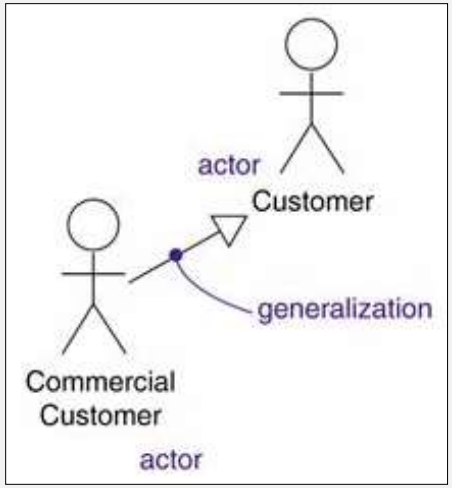
- Tạo ra các UC quá nhỏ
  - Hành động quá đơn giản mà chỉ cần mô tả bởi vài dòng
- Tạo ra quá nhiều Use case (hàng chục)
  - Nhóm các Use case liên quan thành một Use case tổng quát (mức 1)
  - Mô tả các Use Case tổng quát ở một sơ đồ khác (mức 2)
    - Ví dụ: "Quản lý sách" bao gồm "Nhập sách", "Xuất sách", "..."
- Sử dụng các Use-case quá cụ thể, hoặc làm việc với dữ liệu quá cụ thể. Ví dụ:
  - "Tìm sách theo tên" (nên là "Tìm sách")
  - "Nhập Pin vào máy ATM" (nên là "Nhập PIN")
  - "Thêm sách" (nên là "Quản lý sách" bao gồm "Thêm sách")

Các thành phần chính (tiếp)

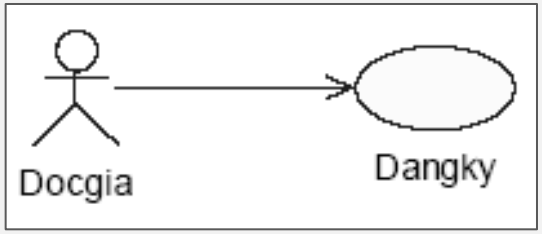
- Liên hệ (relationship)
  - Mối liên hệ giữa các actor với nhau
    + Khái quát hóa
    + Giao tiếp
  - Mối liên hệ giữa actor và use case
    + Giao tiếp
  - Mối liên hệ giữa các use case với nhau
    + Generalization: Khái quát hóa
    + Include: Bao hàm
    + Extend: Mở rộng

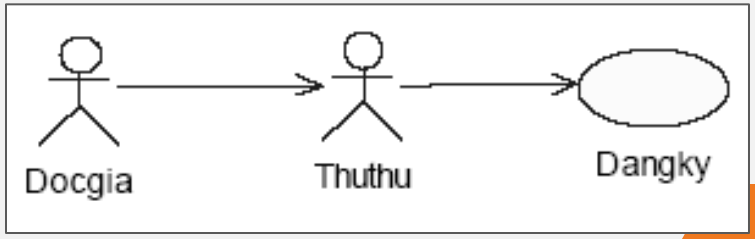
Giữa các actor

- Khái quát hóa (Generalization)
  - Tác nhân con kế thừa tính chất và hành vi của tác nhân cha
  - Ví dụ



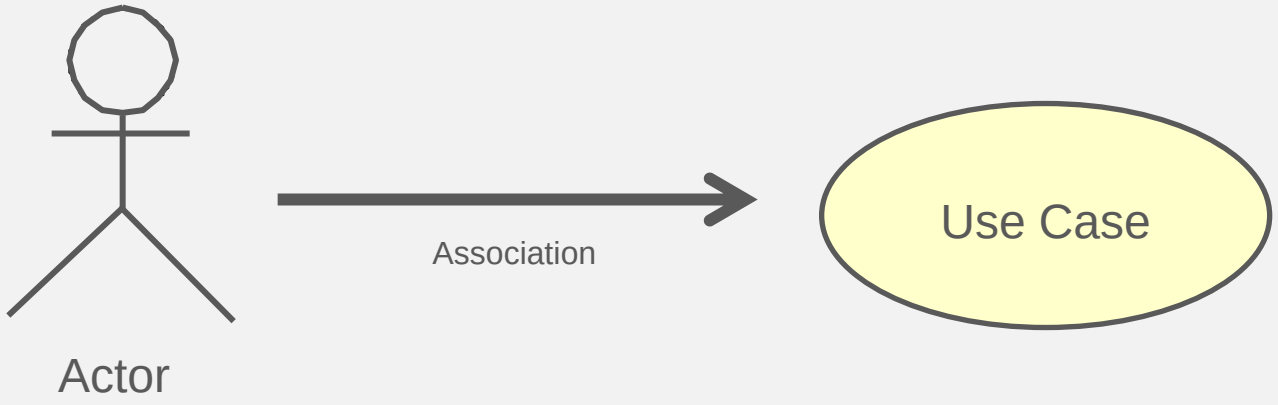
- Giao tiếp (Association)
  - Các tác nhân tương tác với nhau (gửi và nhận thông điệp)
  - Ví dụ





Giữa actor với use case

- Thiết lập quan hệ giữa Tác nhân và Use Case
  - Chúng tương tác bằng cách gửi các tín hiệu cho nhau
- Một use case mô hình hóa một hội thoại giữa các tác nhân và hệ thống
- Một use case được bắt đầu bởi một tác nhân để gọi một chức năng nào đó trong hệ thống.

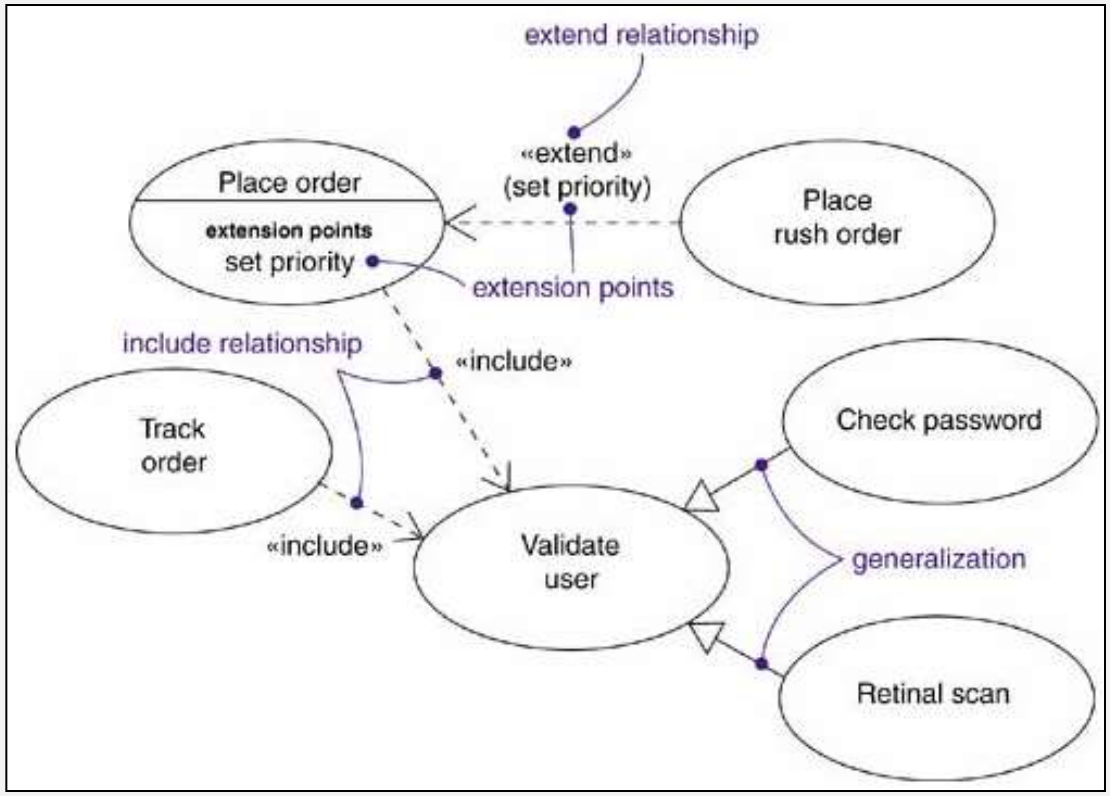


Giữa actor với use case (tiếp)

- Chiều của quan hệ chính là chiều của tín hiệu gửi đi
- Từ tác nhân tới Use Case
  - Kích hoạt Use case
  - Hỏi thông tin nào đó trong hệ thống
  - Thay đổi thông tin nào đó trong hệ thống
  - Thông báo cho UC về một sự kiện đặt biệt nào đó xảy ra với hệ thống
- Từ Use Case tới tác nhân:
  - Nếu như có một điều gì đó xảy ra với HT và tác nhân đó cần được biết sự kiện đó
  - UC đôi khi cần hỏi thông tin nào đó từ một tác nhân trước khi UC đó đưa ra một quyết định

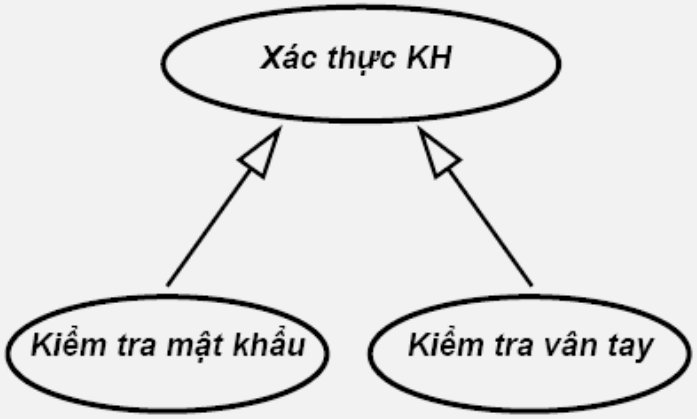
Giữa các use case

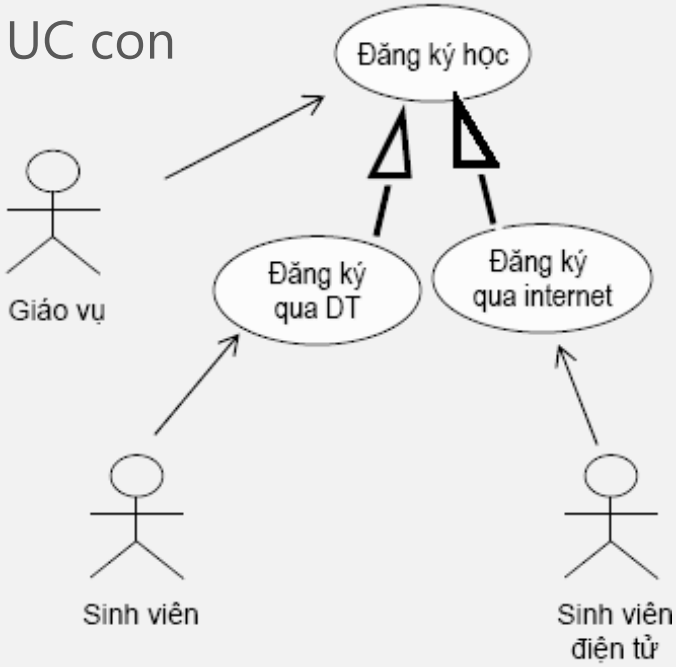
- Generalization
- <<include>>
  - always use
- <<extend>>
  - sometime use



a. Quan hệ generalization

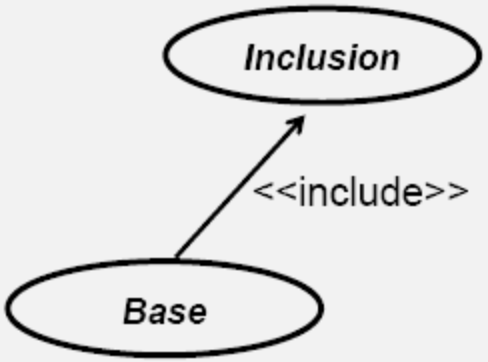
- Được sử dụng để chỉ ra một vài tính chất chung của một nhóm tác nhân hoặc UC
- Sử dụng khái niệm kế thừa
  - Mô tả hành vi chung (chia sẻ) trong UC cha
  - Mô tả hành vi riêng trong (các) UC con

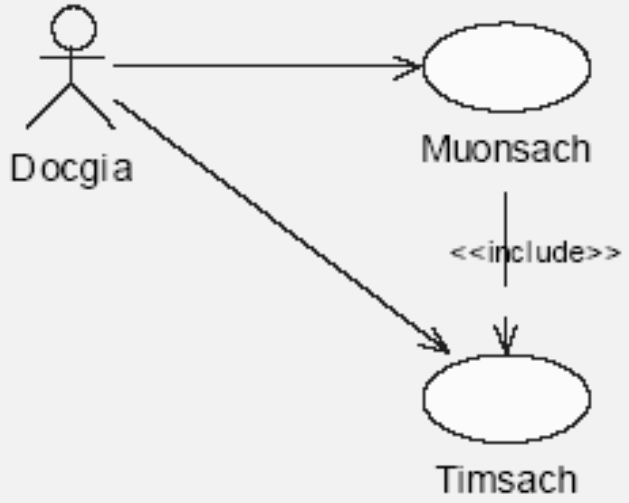




b. Quan hệ <<include>>

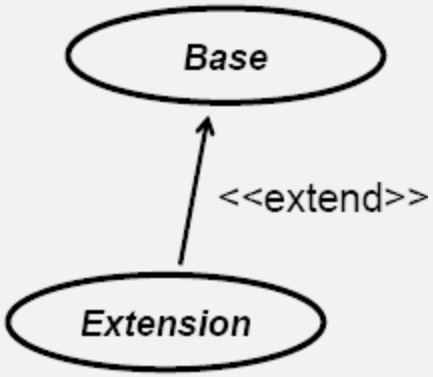
- Cho phép một UC sử dụng chức năng của UC khác
- Chức năng của UC Inclusion sẽ được gọi trong UC Base
- Sử dụng stereotype là <<include>>

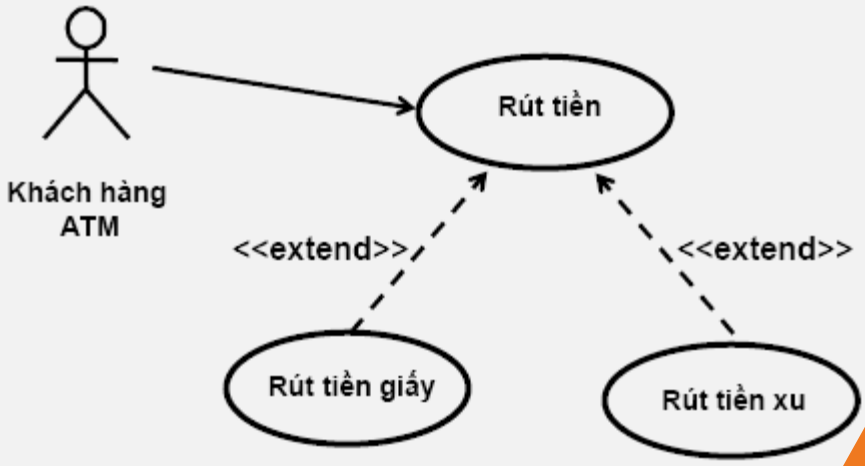




c. Quan hệ <<extend>>

- Cho phép mở rộng chức năng của một UC
- Chèn hành vi của UC Extension vào UC Base
  - Chỉ chèn khi điều kiện extend đúng (mở rộng, phát sinh)
  - Chèn vào lớp cơ sở tại điểm phát sinh (extension point)
- Sử dụng stereotype là <<extend>>

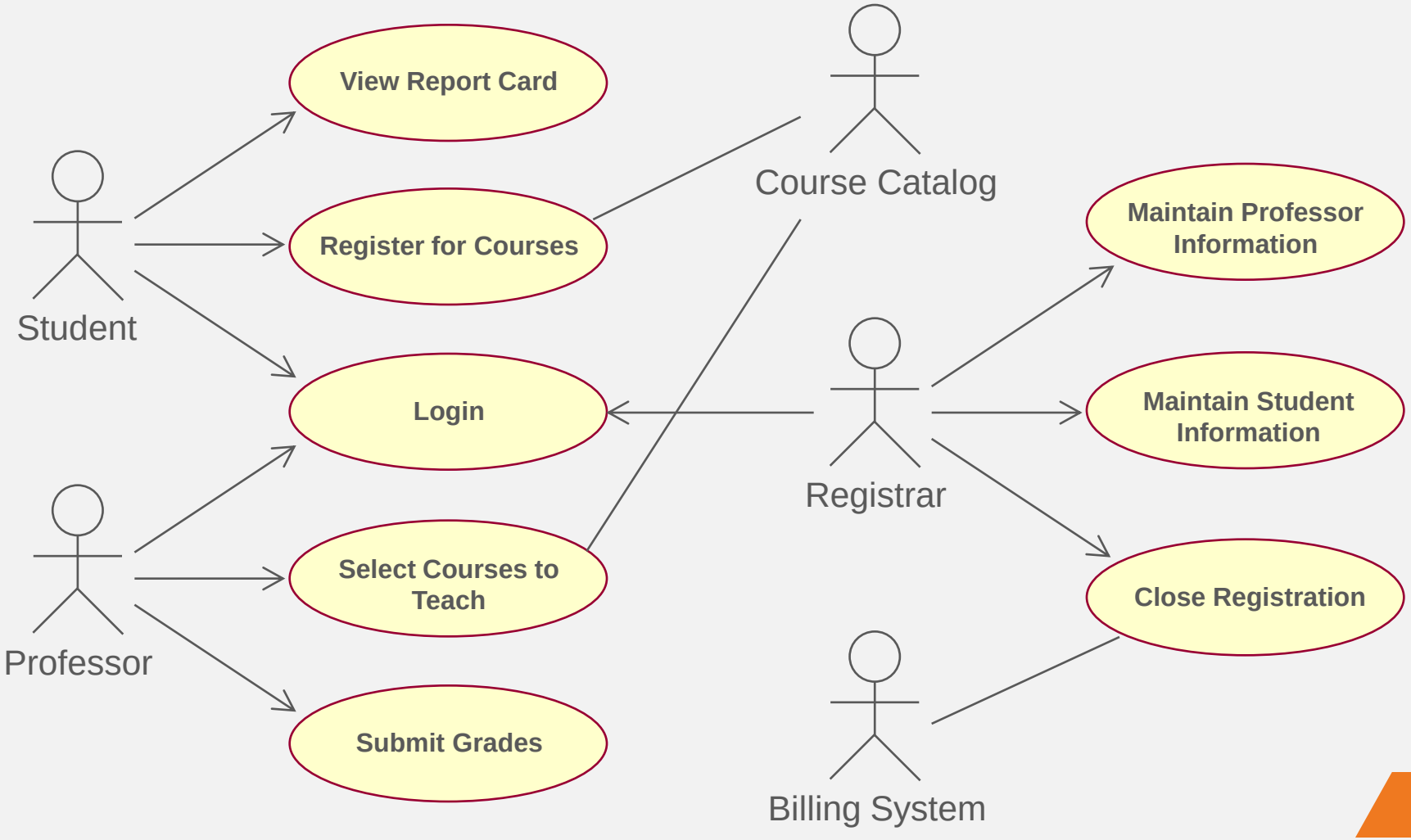




Đọc biểu đồ use case

- Trả lời các câu hỏi sau:
  - Mô tả các chức năng của hệ thống
  - Sinh viên có thể tác động lên những use-case nào?
  - Giáo viên có thể tác động lên những use-case nào?
  - Nếu A vừa là sinh viên vừa là giáo viên, anh ta có thể thực hiện được những use-case nào?
  - Sơ đồ này không nói lên được những gì?
  - Những use-case nào cần thiết thực hiện đầu tiên?

Ví dụ


View Report Card
Register for Courses
Login
Select Courses to Teach
Submit Grades
Student
Professor
Course Catalog
Registrar
Billing System
Maintain Professor Information
Maintain Student Information
Close Registration

3

Biểu đồ hoạt động

Activity diagram

Biểu đồ hoạt động

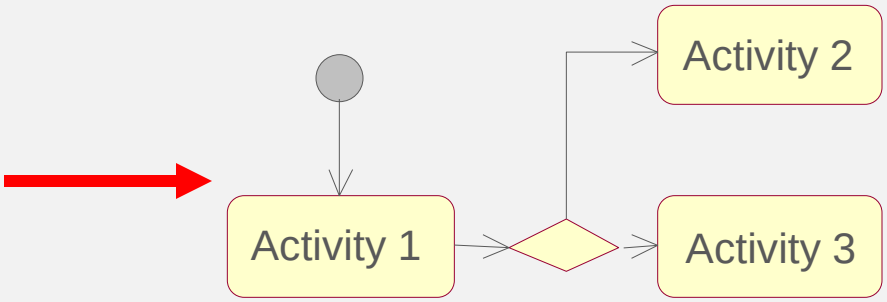
- Biểu đồ hoạt động (Activity Diagram – AD) được sử dụng để mô tả các hoạt động và các hành động được thực hiện trong một use case
  - Biểu đồ luồng (flow chart): Chỉ ra luồng điều khiển từ hoạt động/hành động này đến hoạt/hành động khác.

Flow of Events

This use case starts when the Registrar requests that the system close registration.

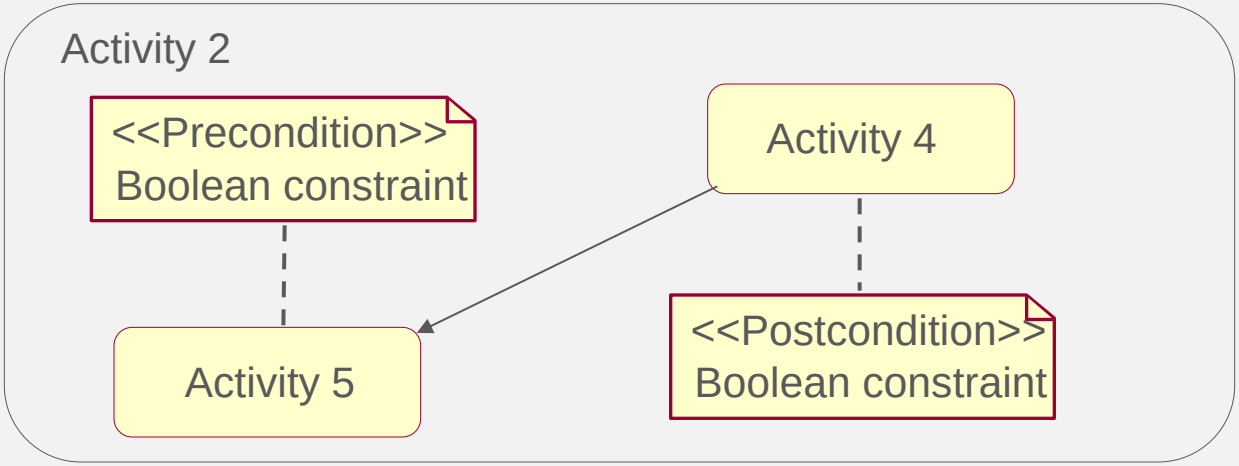
1. The system checks to see if registration is in progress. If it is, then a message is displayed to the Registrar and the use case terminates. The Close Registration processing cannot be performed if registration is in progress.

2. For each course offering, the system checks if a professor has signed up to teach the course offering and at least three students have registered. If so, the system commits the course offering for each schedule that contains it.

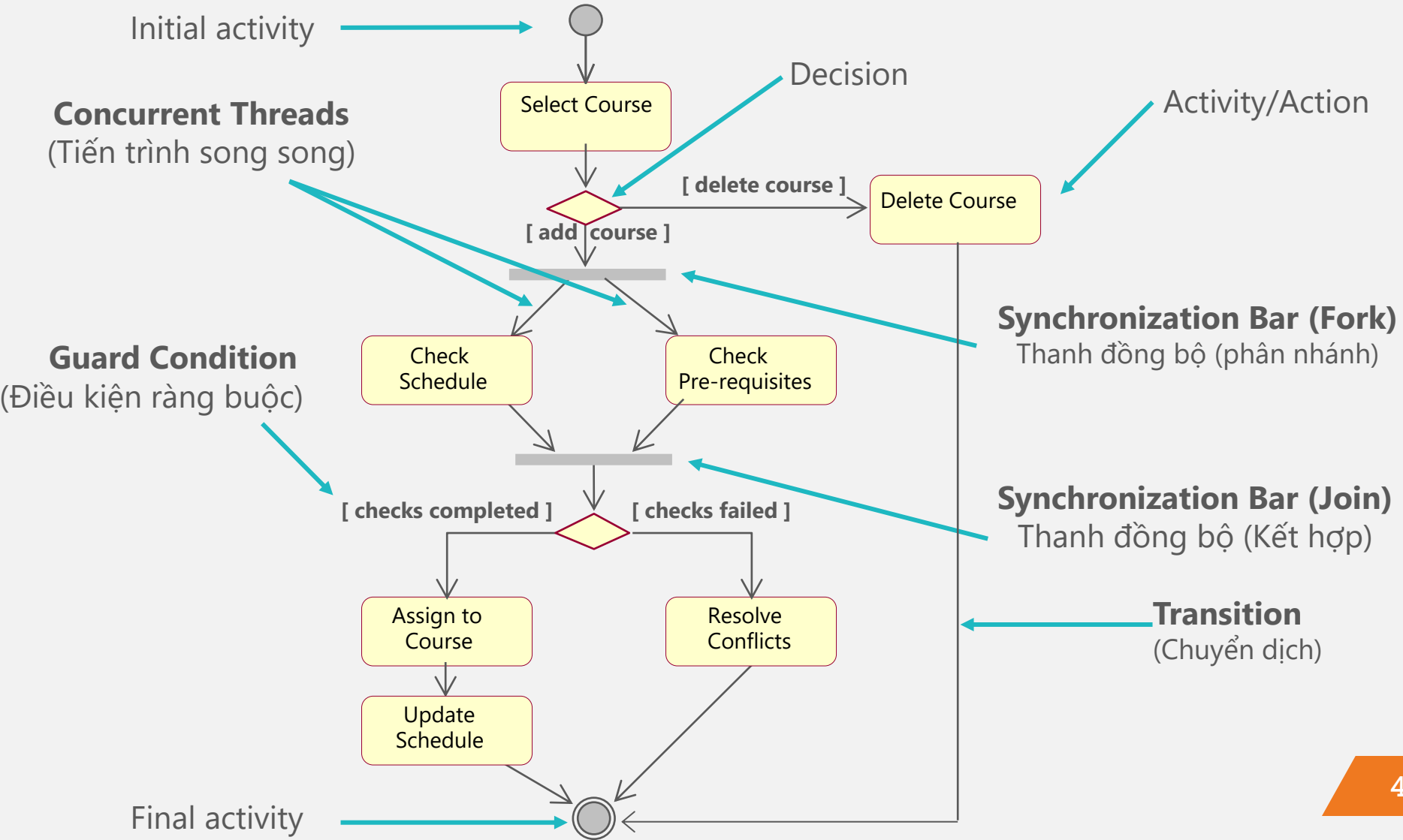


Biểu đồ hoạt động

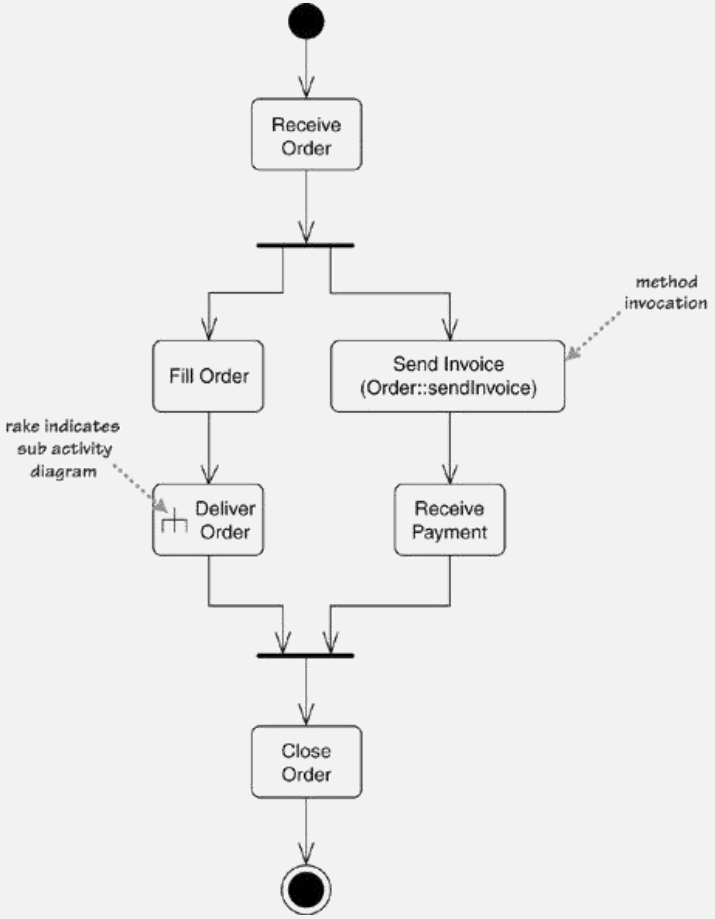
- Hoạt động
  - Đặc tả cho hành vi được diễn tả như một luồng thực thi thông qua sự sắp xếp thứ tự của các đơn vị nhỏ hơn.
  - Các đơn vị nhỏ hơn bao gồm các hoạt động lồng nhau và các hành động riêng lẻ cơ bản
- Có thể chứa các ràng buộc biểu thức logic khi hoạt động được gọi hoặc kết thúc

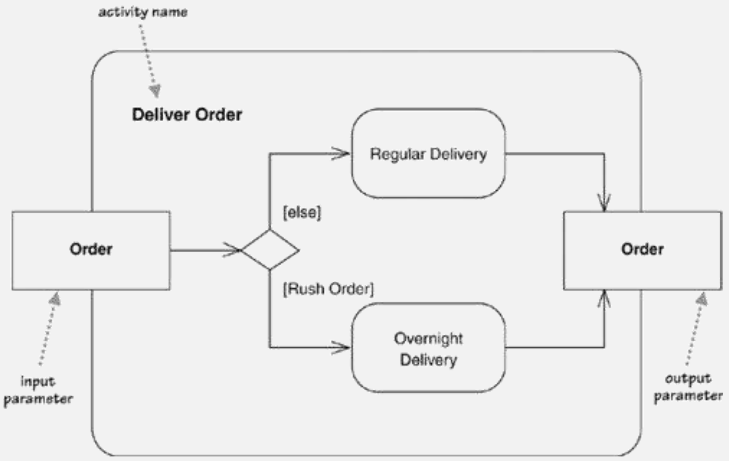


Ví dụ: Đăng ký khóa học



Gọi một AD khác

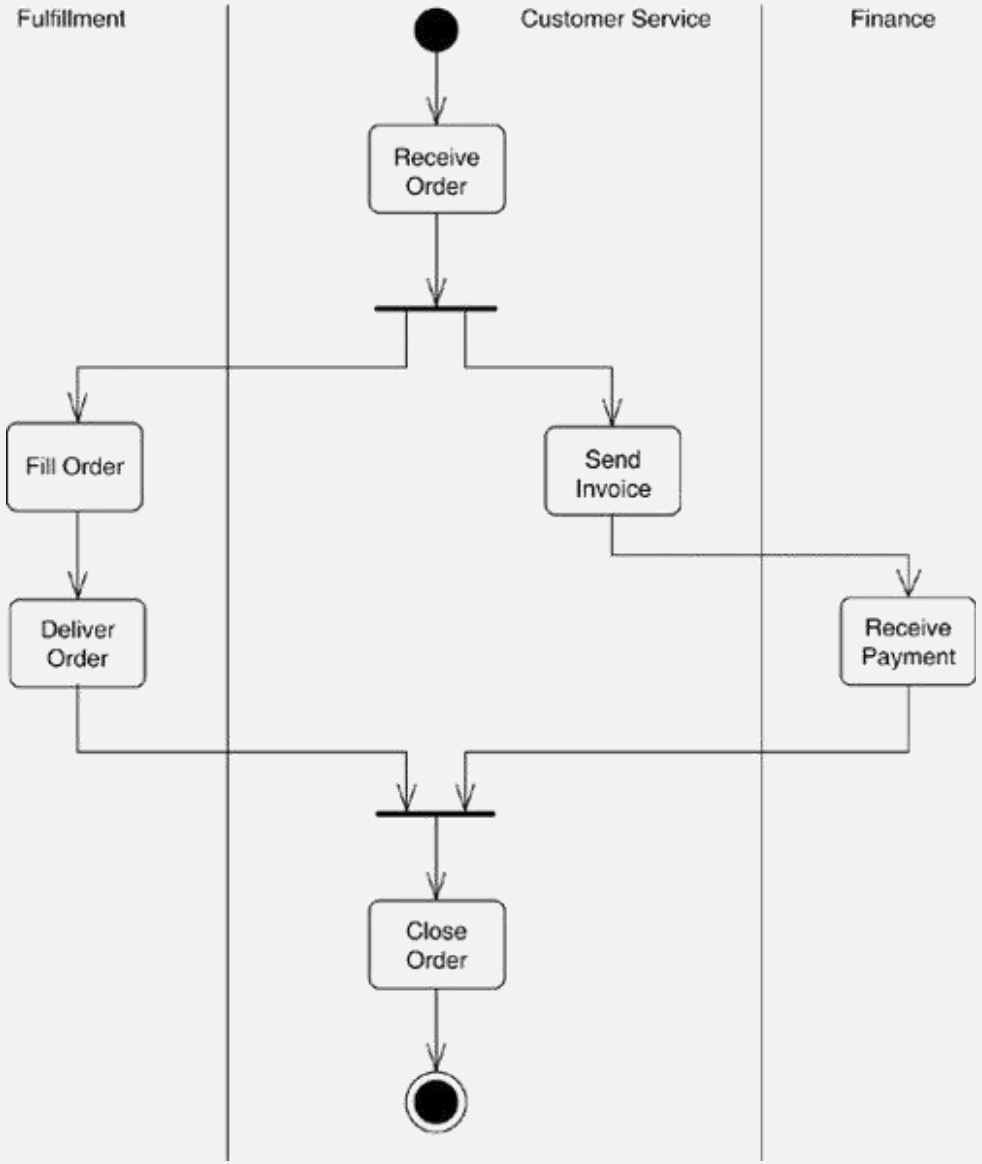




Phân chia (Partition)

- Biểu đồ hoạt động chỉ mô tả điều gì xảy ra chứ không mô tả ai làm gì
- Nếu muốn chỉ ra ai làm gì thì có thể phân chia thành các phần bao gồm các hoạt động do ai làm
- Có thể phân chia theo một chiều (hàng hoặc cột) hoặc hai chiều (cả hàng và cột)

Phân chia một chiều



Bài tập

- Cho:
  - Các tác nhân: Người mua, Hệ thống E-mail, Hệ thống cho vay và Hệ thống báo cáo tín dụng
  - Các use case: Tìm người môi giới, Quản lý hồ sơ cá nhân, Tìm kiếm nhà và Yêu cầu vay
  - Các mối liên kết:
    - Từ người mua tới Tìm người môi giới
    - Từ người mua tới Quản lý hồ sơ cá nhân
    - Từ người mua tới Tìm kiếm nhà
    - Từ người mua tới Yêu cầu vay
    - Quản lý hồ sơ cá nhân tới Hệ thống e-mail
    - Tìm kiếm nhà tới Hệ thống e-mail
    - Yêu cầu vay tới Hệ thống e-mail, Hệ thống cho vay
    - Yêu cầu vay tới Hệ thống báo cáo tín dụng
- Hãy vẽ:
  - Biểu đồ use-case

Bài tập

- Cho:
  - Các trạng thái hành động:
    - Chọn hồ sơ
    - Tìm hồ sơ người mua
    - Tạo hồ sơ mới
    - Đăng nhập
  - Luồng hoạt động:
    - Bắt đầu từ Chọn hồ sơ tới Tìm hồ sơ người mua rồi đi từ Tìm hồ sơ người mua đến Tạo hồ sơ mới nếu hồ sơ không tồn tại. Nếu hồ sơ tồn tại thì có thể Đăng nhập
- Hãy vẽ:
  - Biểu đồ hoạt động

4

Biểu đồ tương tác

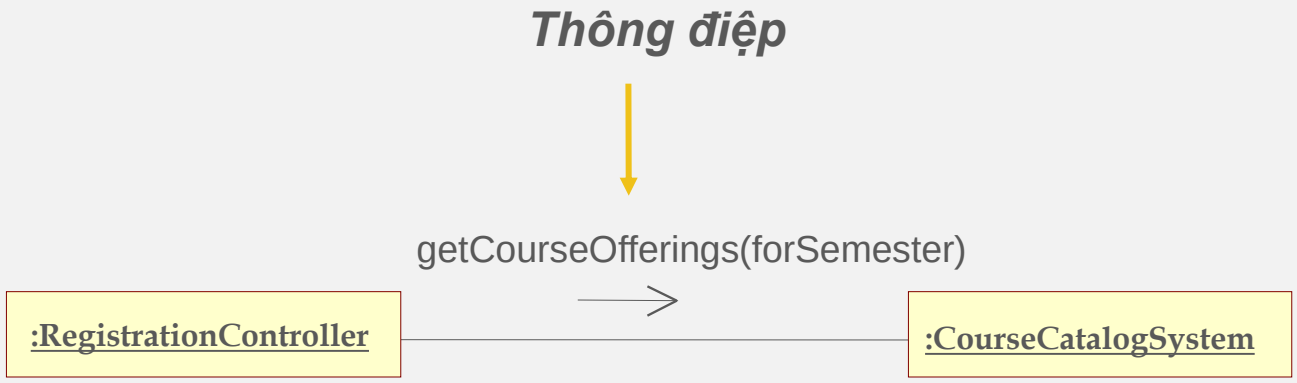
Interaction diagram

Các đối tượng cần phải cộng tác

- Các đối tượng sẽ trở nên vô nghĩa nếu chúng không cộng tác với nhau để giải quyết vấn đề.
  - Mỗi đối tượng có trách nhiệm quản lý hành vi và trạng thái của nó.
  - Không một ai, không một đối tượng nào lại tự mình làm được mọi việc.
- Các đối tượng tương tác với nhau như thế nào?
  - Chúng tương tác với nhau thông qua các thông điệp.
  - Cho biết làm thế nào mà một đối tượng yêu cầu một đối tượng khác thực hiện hành động.

Ví dụ

- Tương tác giữa đối tượng đăng ký khóa học với hệ thống thông tin khóa học

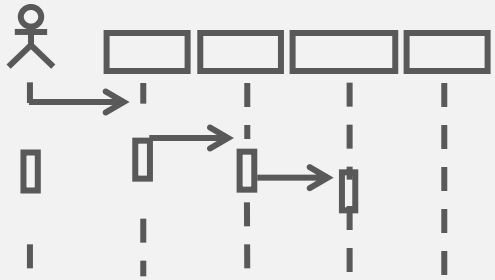


Biểu đồ tương tác

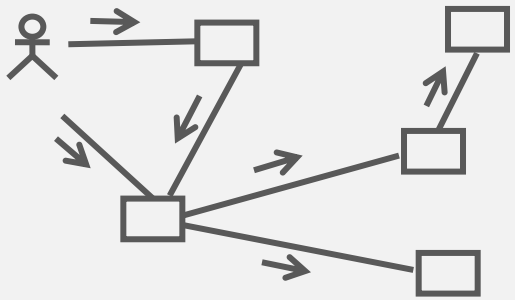
- Biểu đồ tương tác - Interaction diagram
- Mô hình hóa phương diện động của hệ thống, mô tả tương tác giữa các đối tượng
- Thường dùng để mô tả kịch bản của use case
- Các loại biểu đồ tương tác
  - Biểu đồ tuần tự (Sequence diagram)
  - Biểu đồ giao tiếp (Communication diagram)
- Các biến thể chuyên dụng
  - Biểu đồ thời gian (Timing Diagram)
  - Biểu đồ tương tác tổng quát (Interaction Overview Diagram)

Các biểu đồ tương tác

- Biểu đồ trình tự
  - Một cách nhìn hướng về trình tự thời gian tương tác giữa các đối tượng.

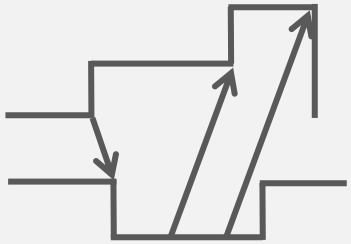


- Biểu đồ giao tiếp
  - Một cách nhìn thông điệp giữa các đốhướng về cấu trúc của quá trình truyền thông điệp giữa các đối tượng.

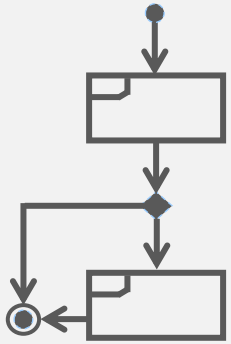


Các biểu đồ tương tác (tiếp)

- Biểu đồ thời gian
  - Một cách nhìn về sự ràng buộc thời gian của các thông điệp trong một tương tác.
  - Thường sử dụng trong các ứng dụng thời gian thực, vì trong các ứng dụng này yếu tố thời gian mang tính quyết định



- Biểu đồ tương tác tổng quan
  - Một cách nhìn tương tác ở mức cao bằng cách kết hợp các biểu đồ tương tác theo một trình tự logic nào đó.



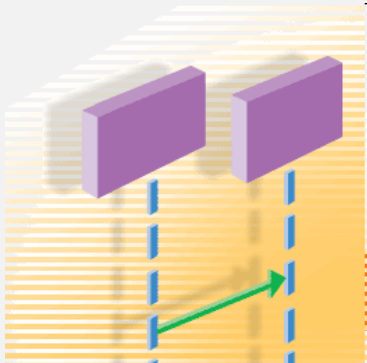
4.1

Biểu đồ tương tác
Biểu đồ trình tự

Sequence diagram (SD)

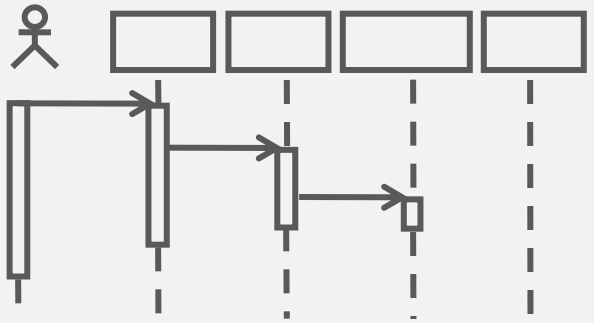
Biểu đồ trình tự

- Biểu đồ trình tự (Sequence diagram – SD)
- Được sử dụng để xác định và chỉ rõ vai trò của các đối tượng tham gia vào luồng sự kiện của use case
- Là một loại biểu đồ tương tác, mô tả mô hình tương tác giữa các đối tượng, trong đó nhấn mạnh vào trình tự thời gian của các thông điệp trao đổi giữa các đối tượng đó.



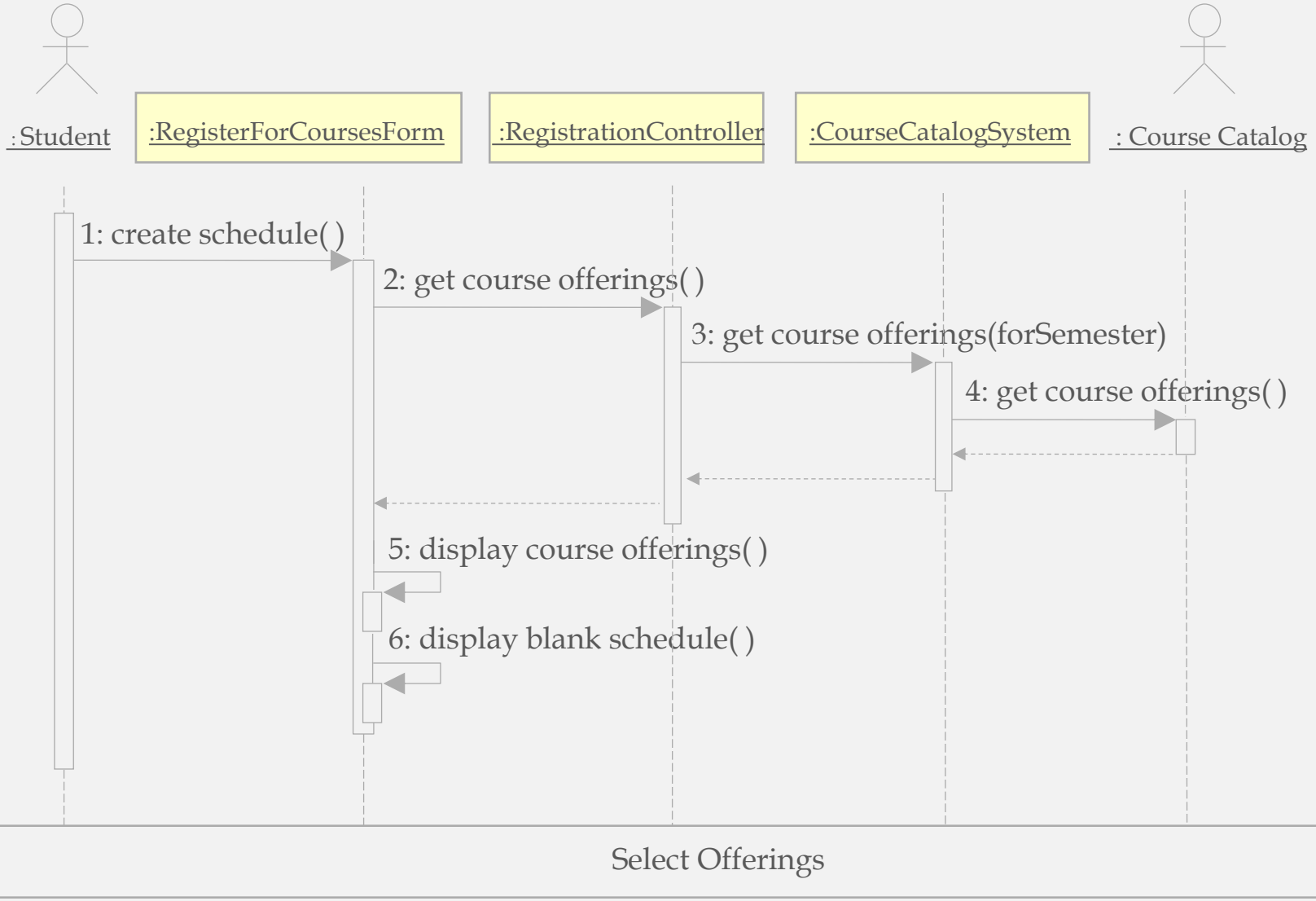
Biểu đồ trình tự

- Biểu đồ trình tự chỉ ra:
  - Các đối tượng tham gia vào tương tác.
  - Thời gian sống của các đối tượng
  - Trình tự các thông điệp được trao đổi.

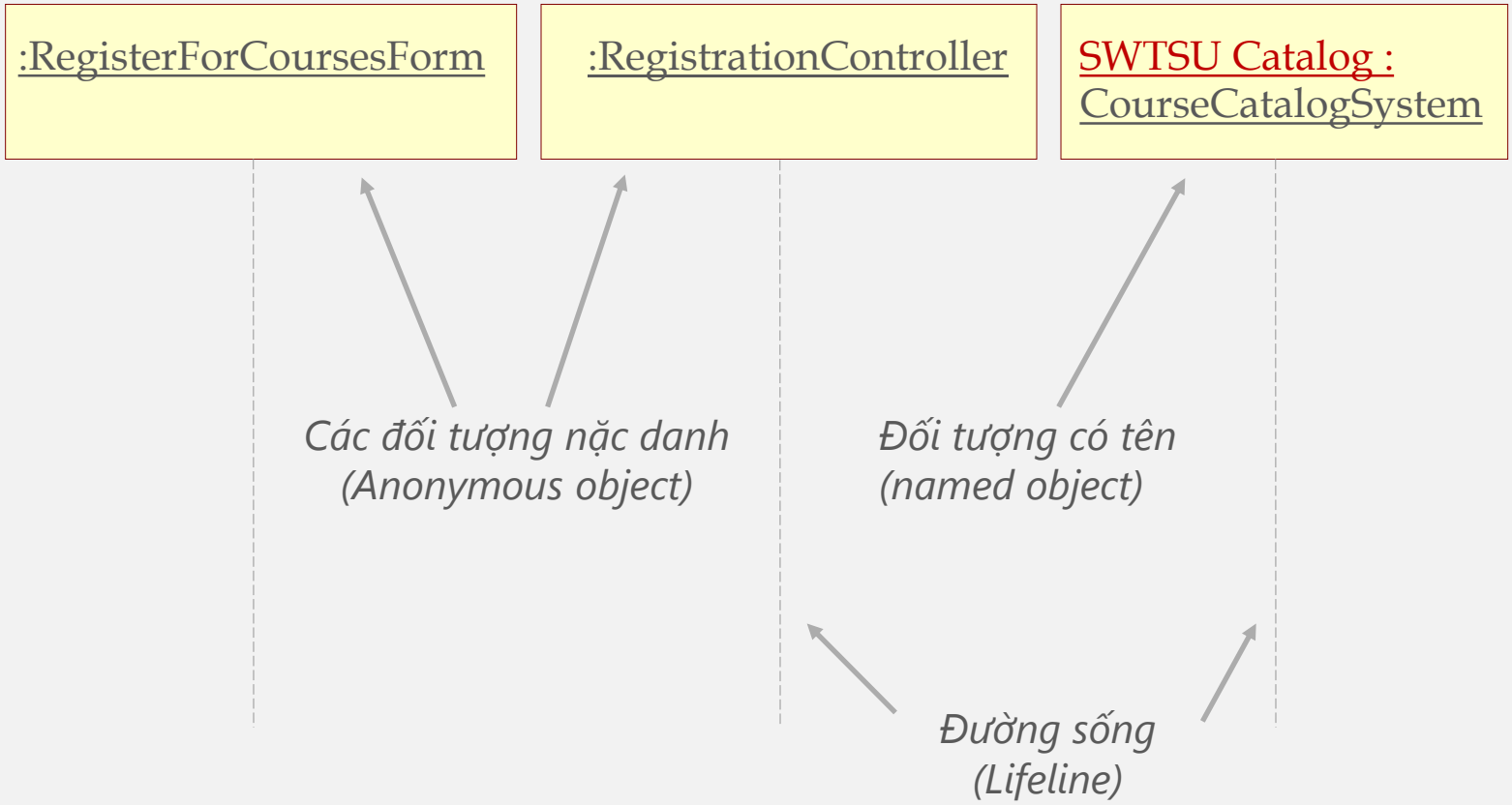


Biểu đồ trình tự

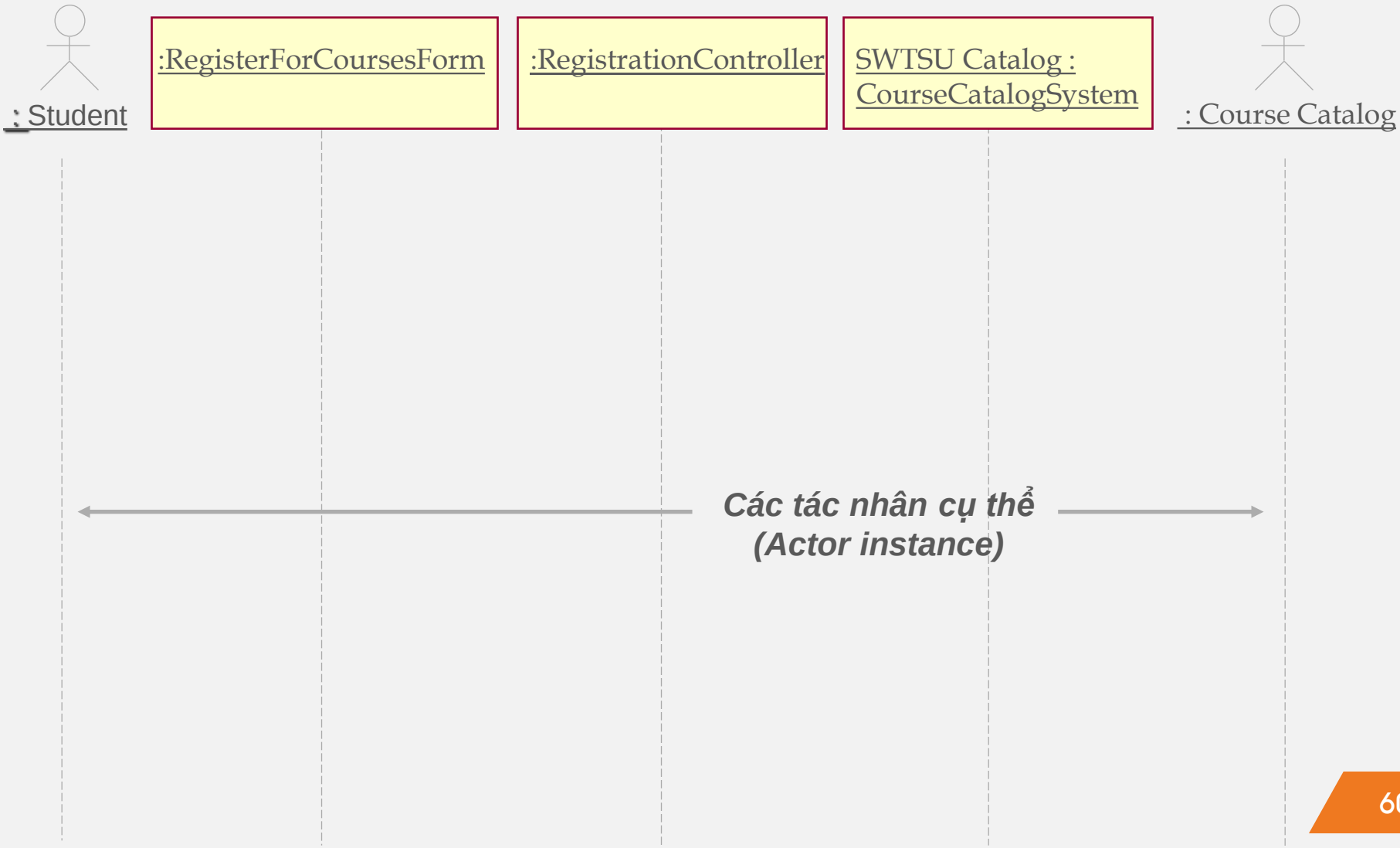
Ví dụ: Đăng ký khóa học



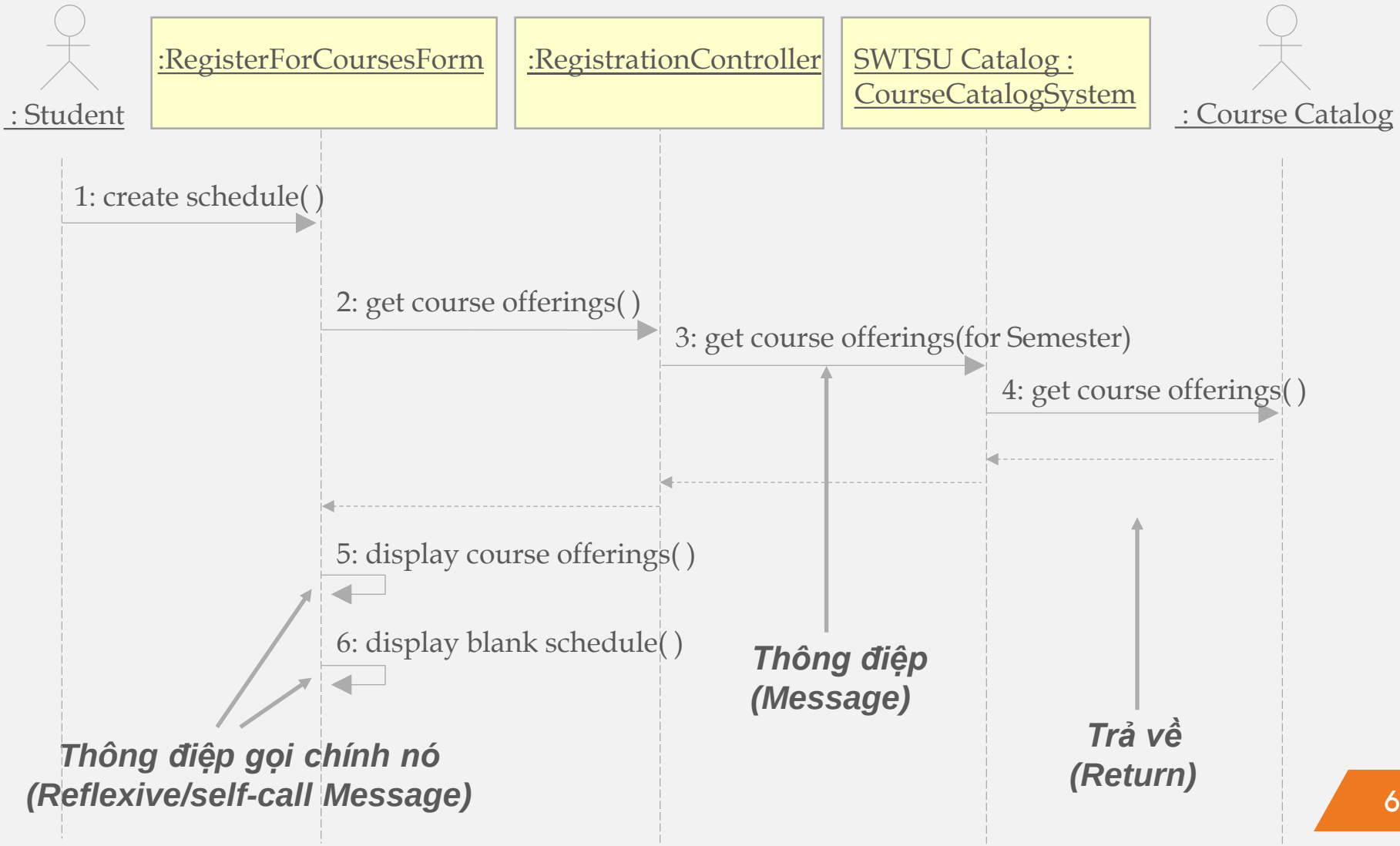
Biểu đồ trình tự: Đối tượng



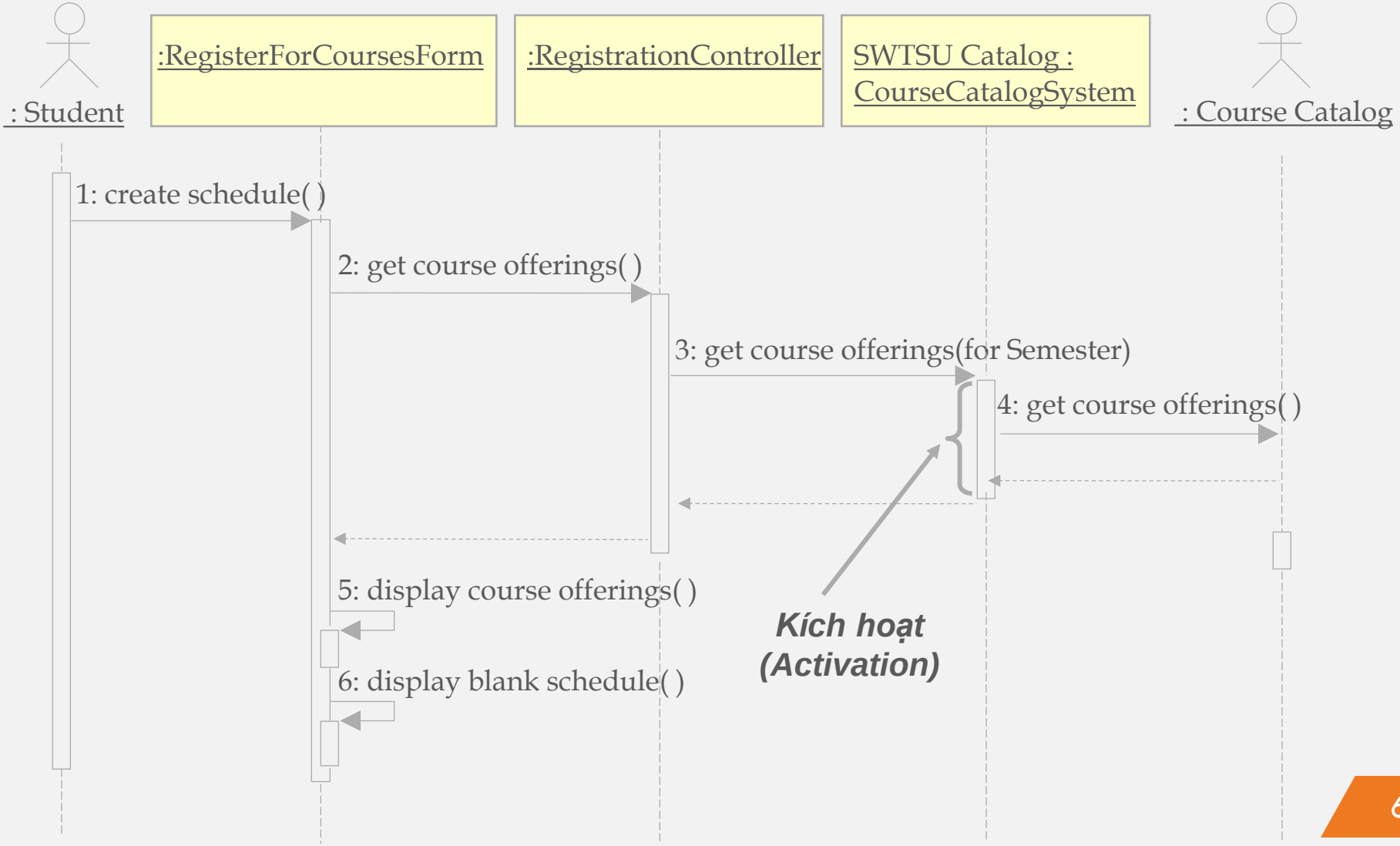
Biểu đồ trình tự: Tác nhân



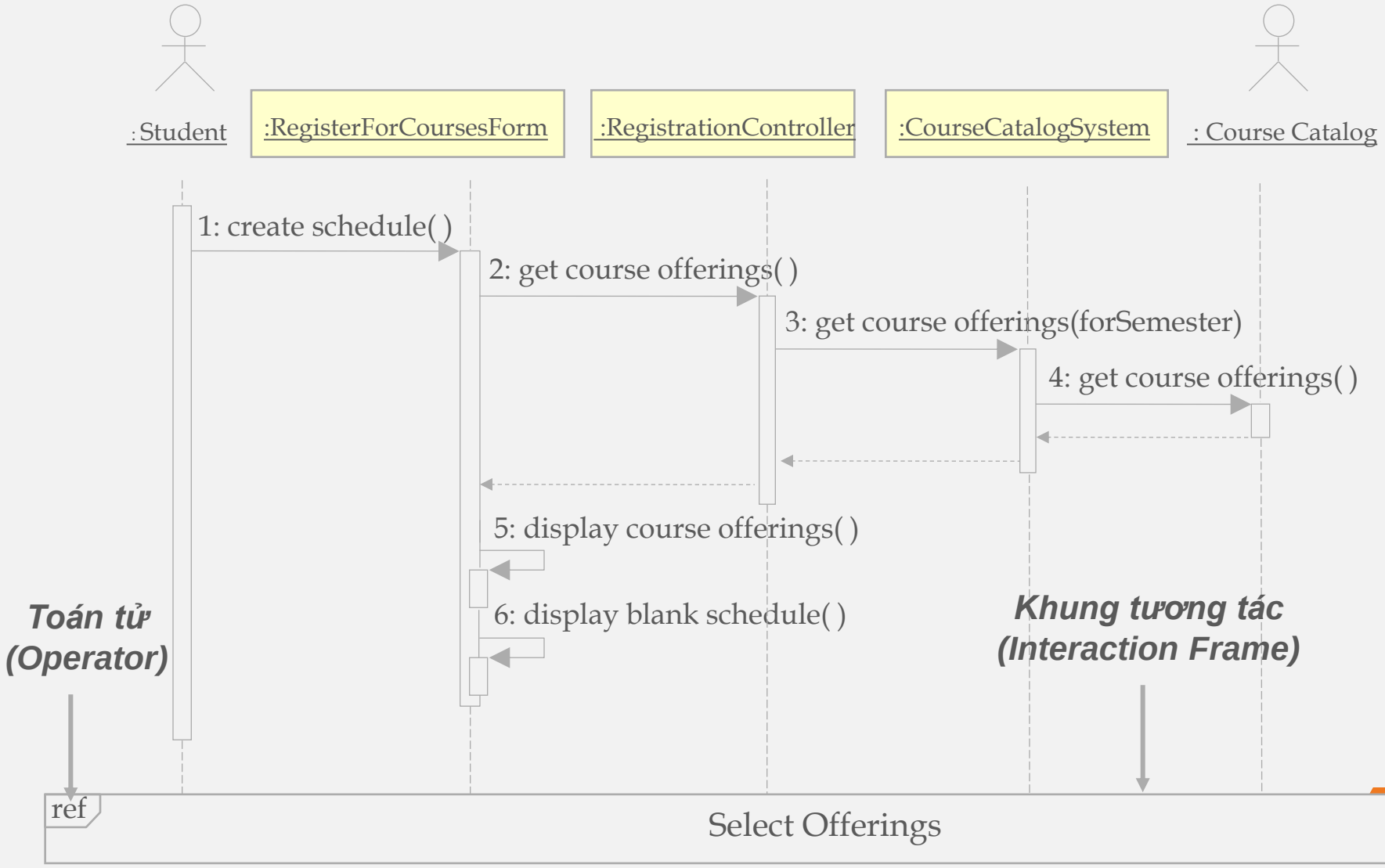
Biểu đồ trình tự: Thông điệp



Biểu đồ trình tự: Kích hoạt



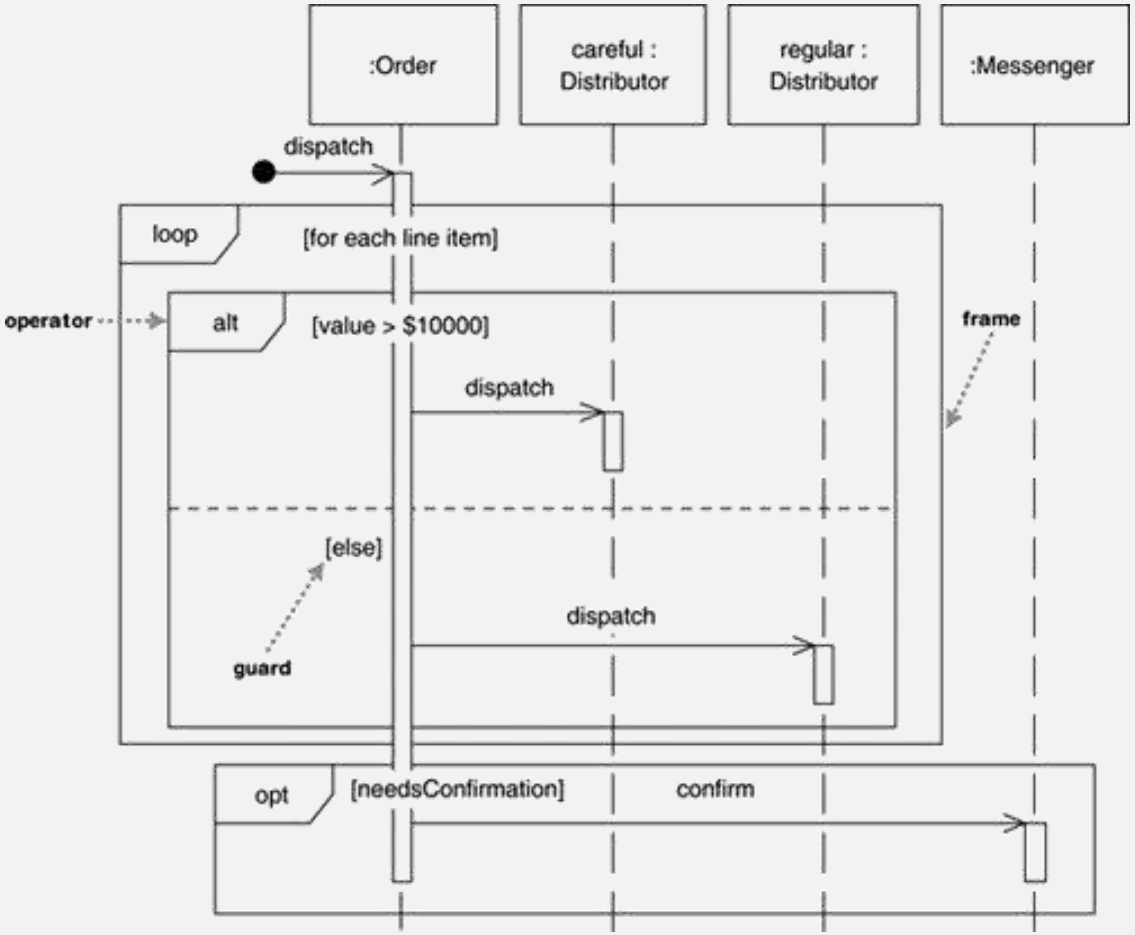
Biểu đồ trình tự: Khung tương tác



Biểu đồ trình tự: Khung tương tác

Toán tử	*Ý nghĩa*
alt	Khung lựa chọn nhiều, chỉ có lựa chọn có điều kiện đúng sẽ được thực hiện
opt	Tùy chọn, chỉ thực hiện khi điều kiện thỏa mãn
par	Song song, mỗi khung chạy song song
loop	Lặp lại, khung có thể được thực hiện nhiều lần
region	Vùng then chốt, tại một thời điểm chỉ có một luồng chạy nó
ref	Tham chiếu đến một tương tác khác trong biểu đồ khác, vẽ trùm trên các lifetime liên quan, có thể có tham số và giá trị trả về
sd	Vẽ xung quanh 1 biểu đồ biểu đồ trình tự nếu cần

Ví dụ




```
procedure dispatch
 foreach (lineitem)
   if
(product.value>$10K)
     careful.dispatch
   else
     regular.dispatch
   end if
  end for
  if (needsConfirmation)
   messenger.confirm
end procedure
```

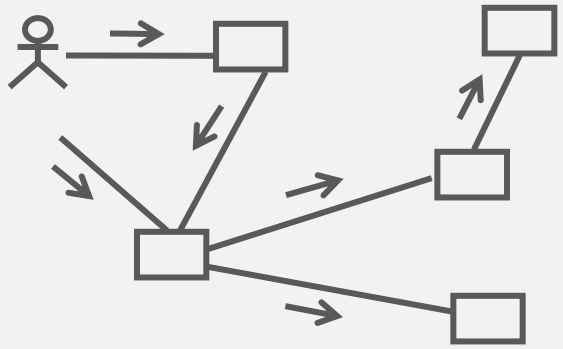
4.2

Biểu đồ tương tác
Biểu đồ giao tiếp

Communication diagram (CD)

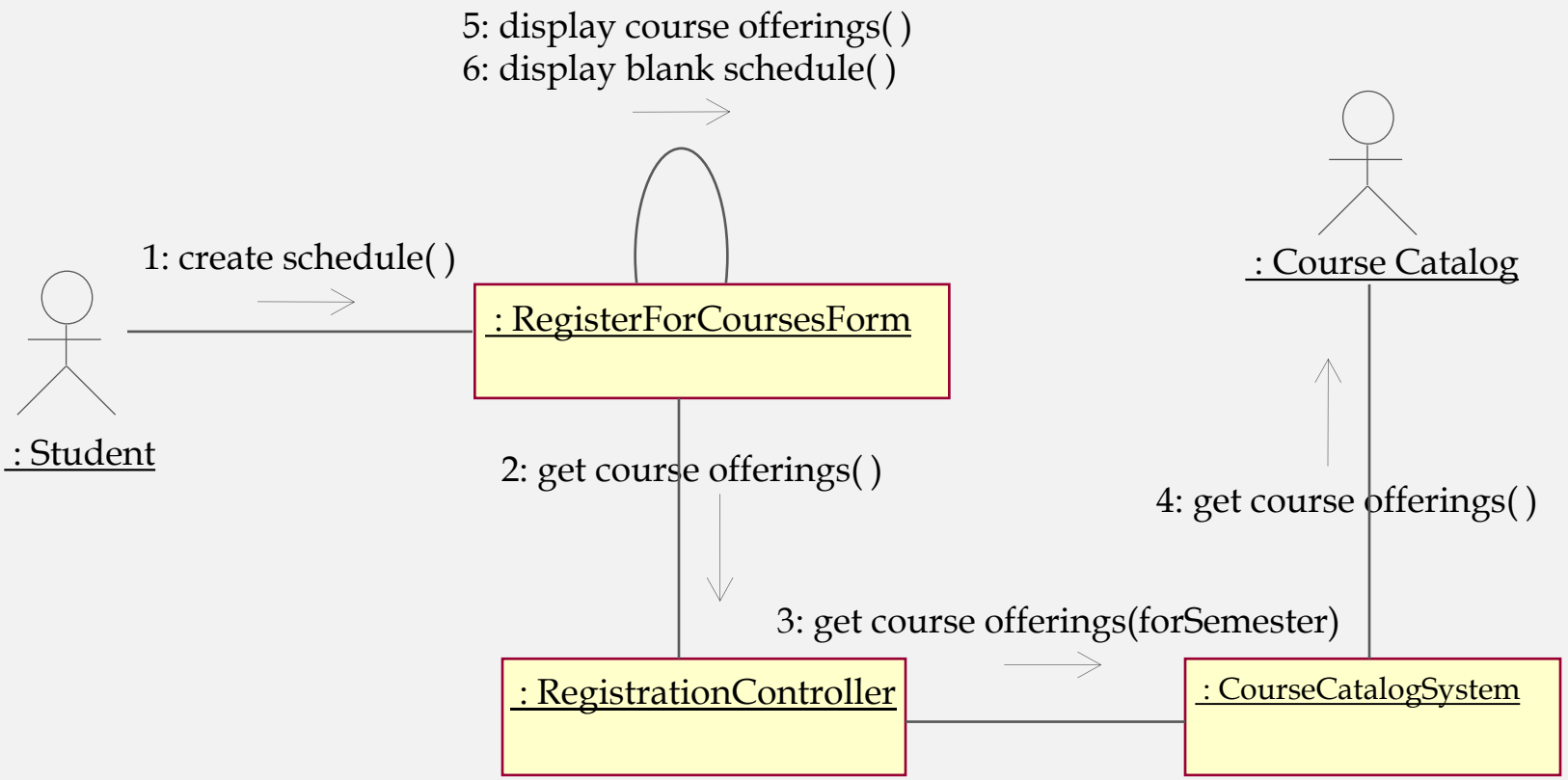
Biểu đồ giao tiếp

- Biểu đồ giao tiếp nhấn mạnh vào việc tổ chức các đối tượng tham gia vào tương tác.
- Biểu đồ giao tiếp chỉ ra:
  - Các đối tượng tham gia vào tương tác.
  - Các liên kết giữa các đối tượng.
  - Các thông điệp trao đổi giữa các đối tượng.

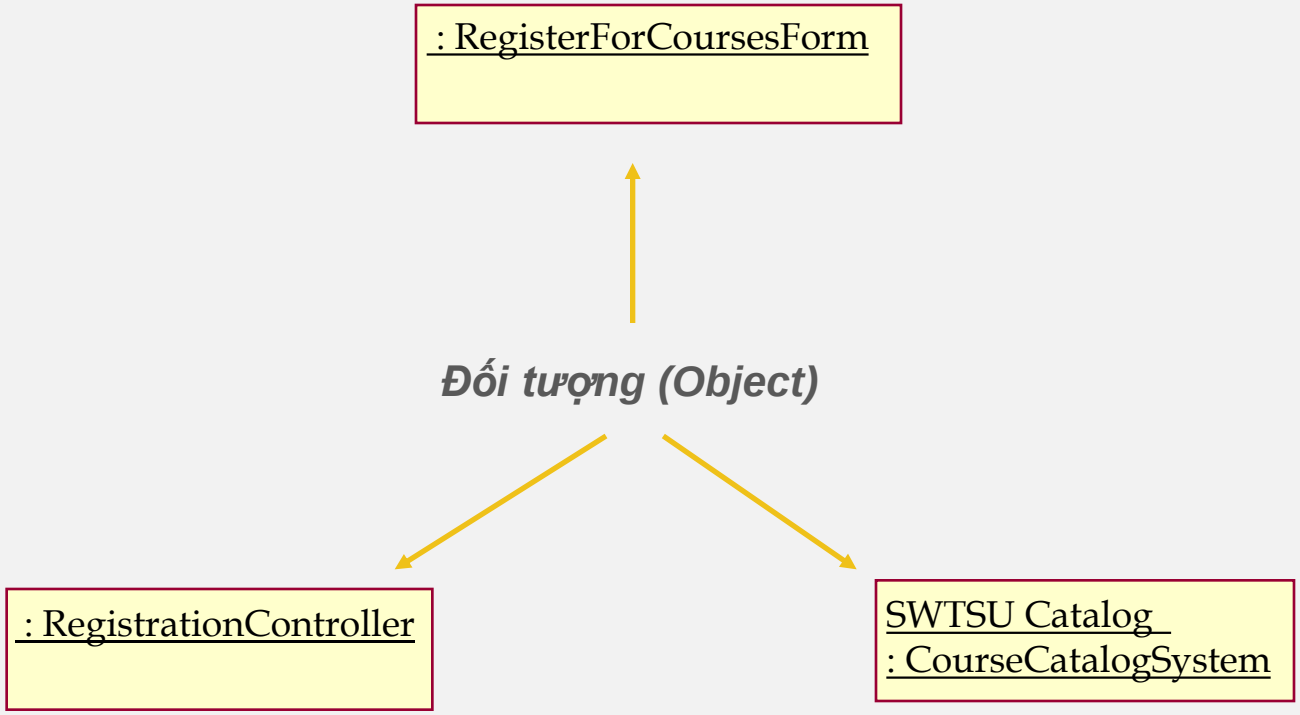


Biểu đồ giao tiếp

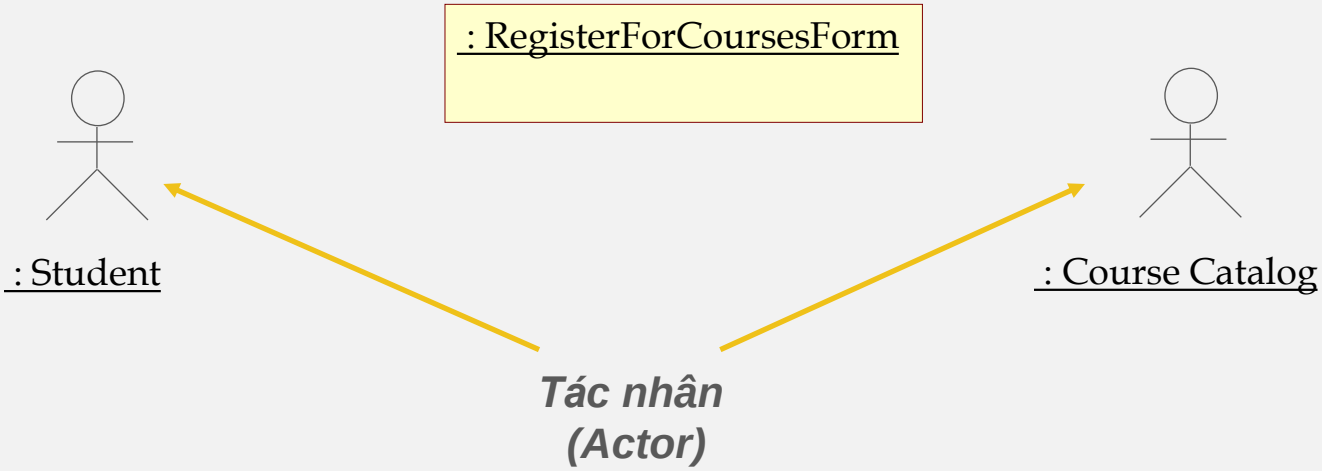
Ví dụ


5: display course offerings( )
6: display blank schedule( )
1: create schedule( )
: Course Catalog
: RegisterForCoursesForm
: Student
2: get course offerings( )
4: get course offerings( )
3: get course offerings(forSemester)
: RegistrationController
: CourseCatalogSystem

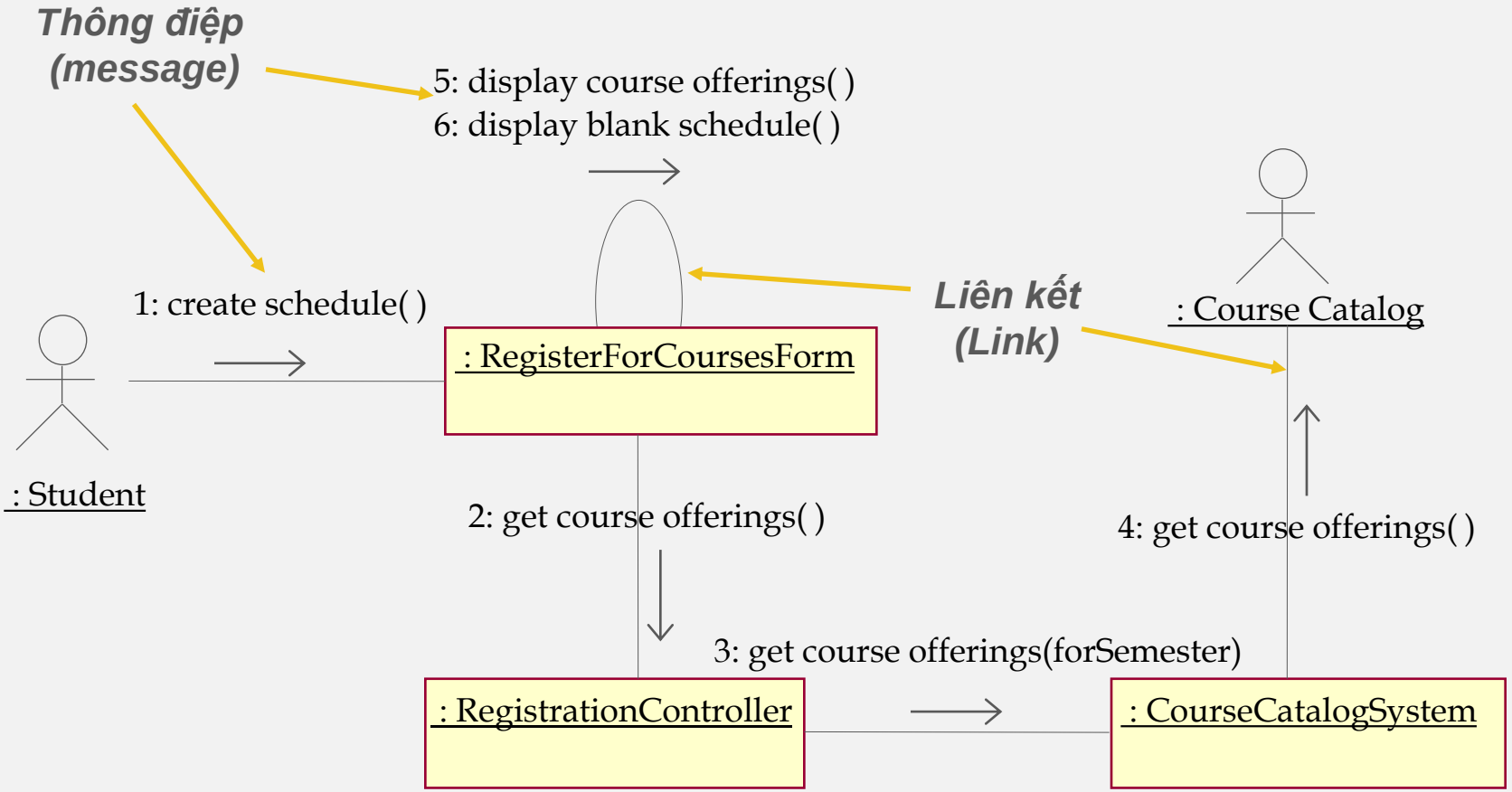
Đối tượng


: RegisterForCoursesForm
Đối tượng (Object)
: RegistrationController
SWTSU Catalog
: CourseCatalogSystem

Tác nhân


: RegisterForCoursesForm
: Student
: Course Catalog
Tác nhân
(Actor)
: RegistrationController
SWTSU Catalog
: CourseCatalogSystem

Liên kết và thông điệp



Biểu đồ trình tự và giao tiếp

- Giống nhau:
- Tương đương về ngữ nghĩa
  - Cùng đưa ra thông tin về sự tương tác giữa các đối tượng qua các thông điệp
  - Có thể chuyển đổi giữa hai biểu đồ mà không mất mát thông tin
- Mô hình hóa phương diện động của hệ thống
- Mô hình hóa kịch bản use case.

Biểu đồ trình tự và giao tiếp

Biểu đồ tuần tự

- Chỉ ra thứ tự rõ ràng của các thông điệp
- Thể hiện tốt hơn luồng công việc
- Mô hình hóa trực quan hơn toàn bộ luồng thực thi (theo thời gian)
- Thể hiện tốt hơn đối với các đặc tả thời gian thực và các kịch bản phức tạp

Biểu đồ giao tiếp

- Chỉ ra mối quan hệ rõ ràng giữa các đối tượng
- Thể hiện tốt hơn quá trình giao tiếp
- Mô hình hóa trực quan hơn cho tất cả các ảnh hưởng của đối tượng
- Thể hiện rõ hơn hiệu quả của quá trình tương tác trên từng đối tượng, dễ hiểu hơn cho các buổi brainstorming

5

Biểu đồ lớp

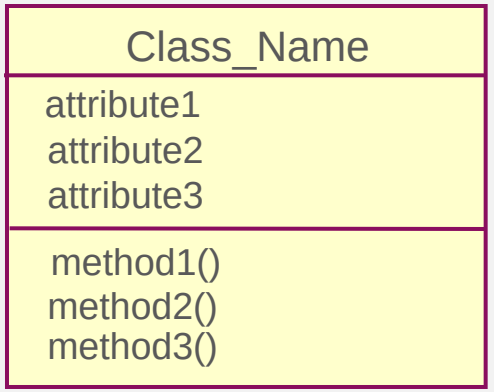
Class diagram

Biểu đồ lớp

- Biểu đồ lớp (Class diagram – CD) chỉ ra sự tồn tại của các lớp và mối quan hệ giữa chúng trong bản thiết kế logic của một hệ thống
  - Chỉ ra cấu trúc tĩnh của mô hình như lớp, cấu trúc bên trong của chúng và mối quan hệ với các lớp khác.
  - Chỉ ra tất cả hoặc một phần cấu trúc lớp của một hệ thống.
  - Không đưa ra các thông tin tạm thời.
- Khung nhìn tĩnh của một hệ thống chủ yếu hỗ trợ các yêu cầu chức năng của hệ thống.

Lớp

- Sử dụng hình chữ nhật gồm 3 thành phần
  - Tên lớp
  - Các thuộc tính
  - Các phương thức



Biểu diễn thuộc tính

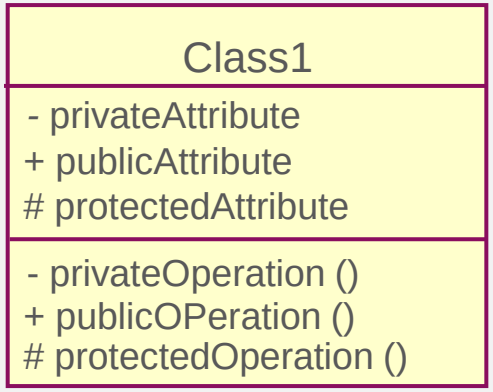
- Chỉ ra tên, kiểu và giá trị mặc định nếu có
  - `attributeName : Type = Default`
- Tuân theo quy ước đặt tên của ngôn ngữ cài đặt và của dự án.
- Kiểu (type) nên là kiểu dữ liệu cơ bản trong ngôn ngữ thực thi
  - Kiểu dữ liệu có sẵn, kiểu dữ liệu người dùng định nghĩa, hoặc lớp tự định nghĩa.

Mô tả phương thức

- Tên phương thức:
  - Mô tả kết quả
  - Sử dụng góc nhìn của đối tượng khách (client – đối tượng gọi)
  - Nhất quán giữa các lớp
- Chữ ký của phương thức:
  - `operationName([direction] parameter:class,...):returnType`
    - Direction: in (mặc định), out hoặc inout

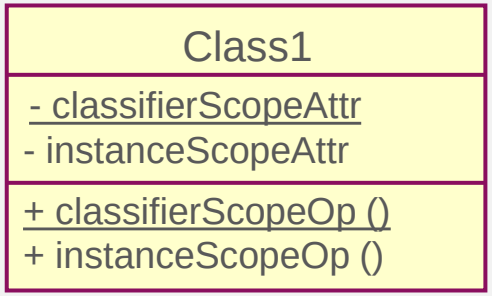
Phạm vi truy cập

- Phạm vi truy cập được sử dụng để thực hiện khả năng đóng gói
- Sử dụng các ký hiệu
  - \+ Public access
  - # Protected access
  - \- Private access



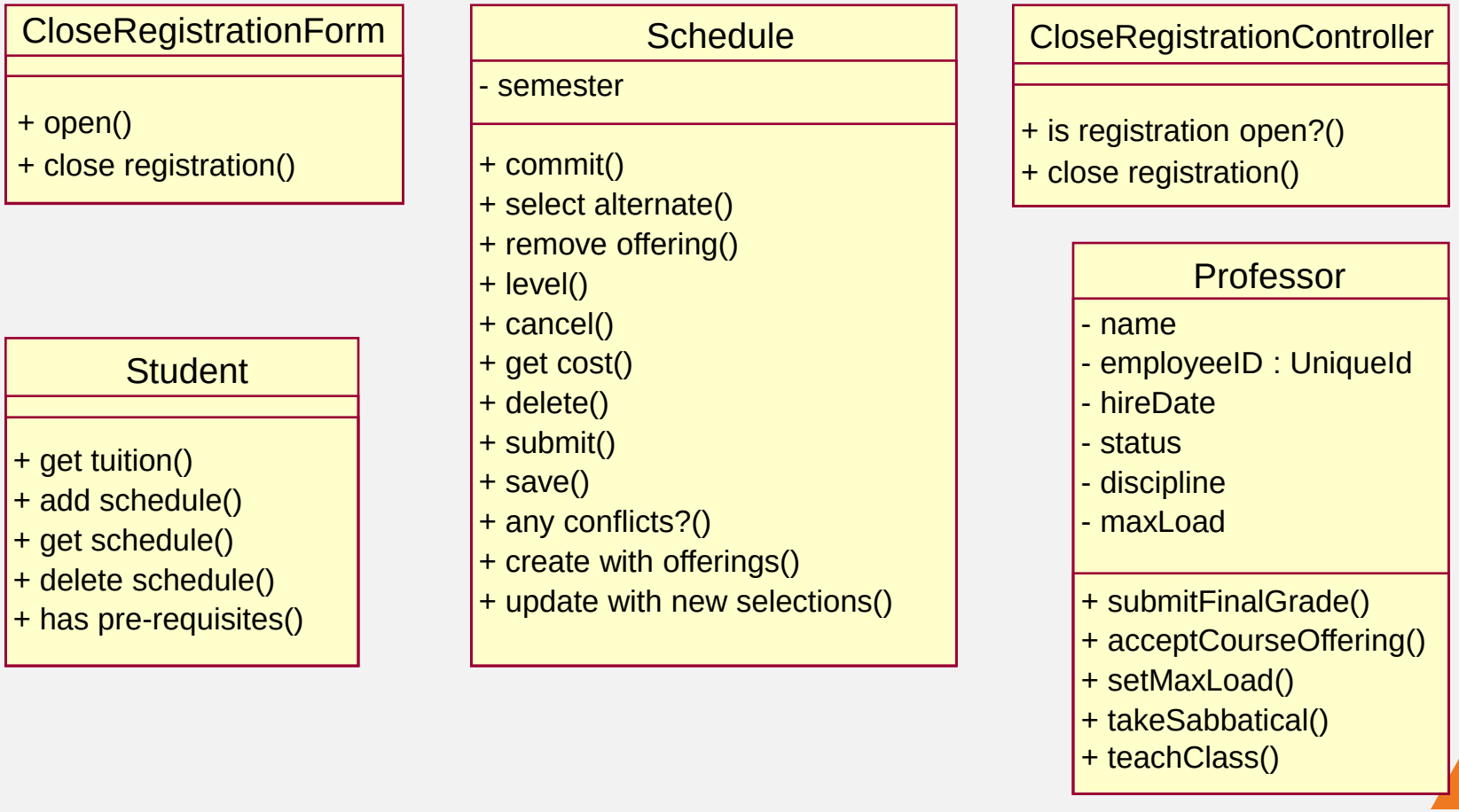
Thành viên lớp

- Phạm vi truy cập xác định số lượng thể hiện của thuộc tính/thao tác:
  - Instance (Thành viên đối tượng): Một thể hiện cho mỗi thể hiện của mỗi lớp
  - Classifier (Thành viên lớp): Một thể hiện cho tất cả các thể hiện của lớp
- Thành viên lớp (thành viên tĩnh) được ký hiệu bằng cách gạch dưới tên thuộc tính/thao tác.



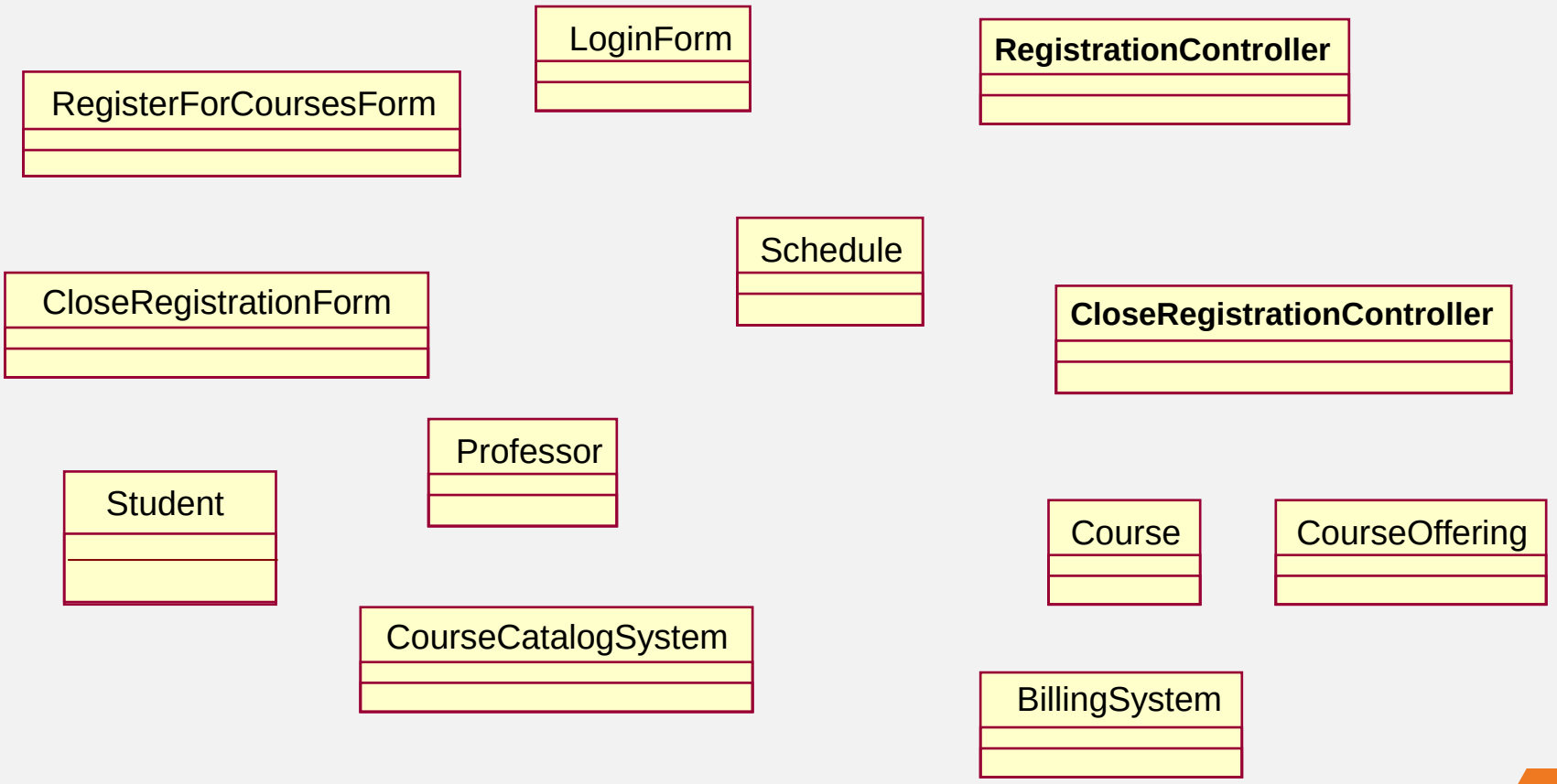
Ví dụ

- Hệ thống đăng ký khóa học



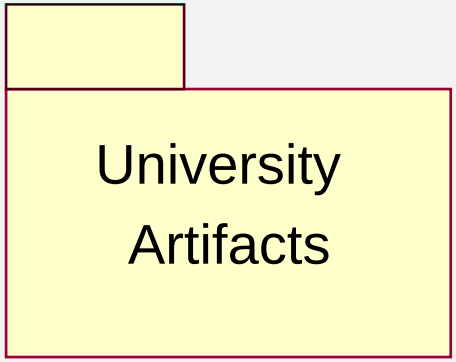
Ví dụ

- Biểu đồ lớp sơ lược

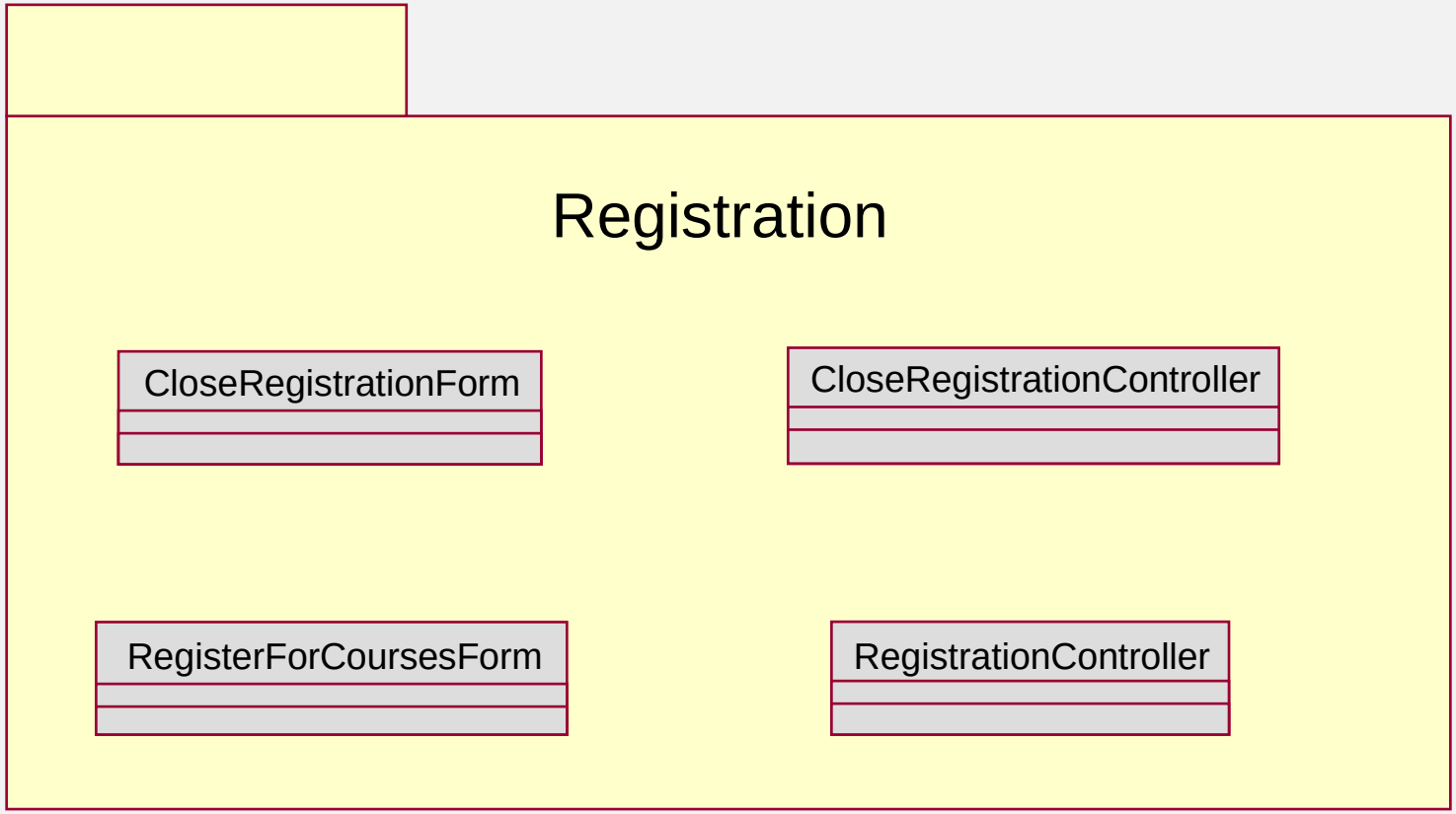


Gói

- Một cơ chế chung để tổ chức các phần tử thành nhóm.
- Một phần tử trong mô hình có thể chứa các phần tử khác.

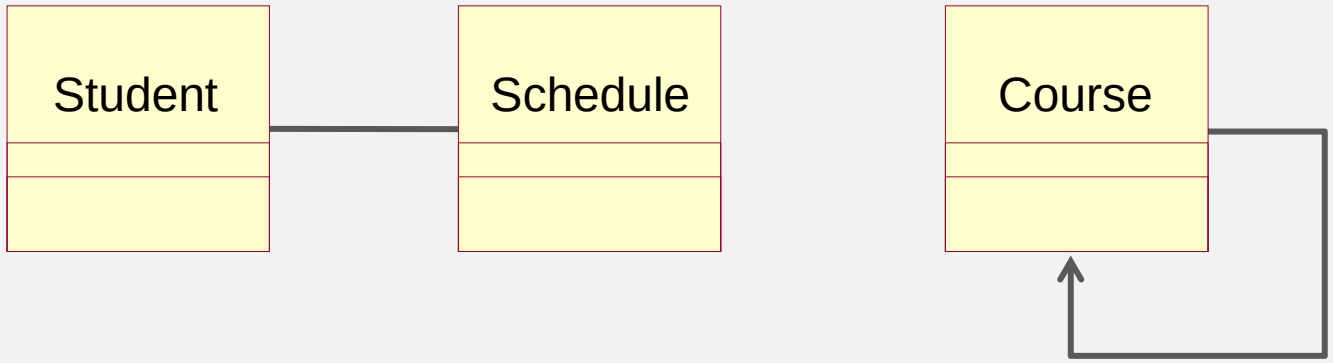


Ví dụ


Registration
CloseRegistrationForm
CloseRegistrationController
RegisterForCoursesForm
RegistrationController

Liên kết

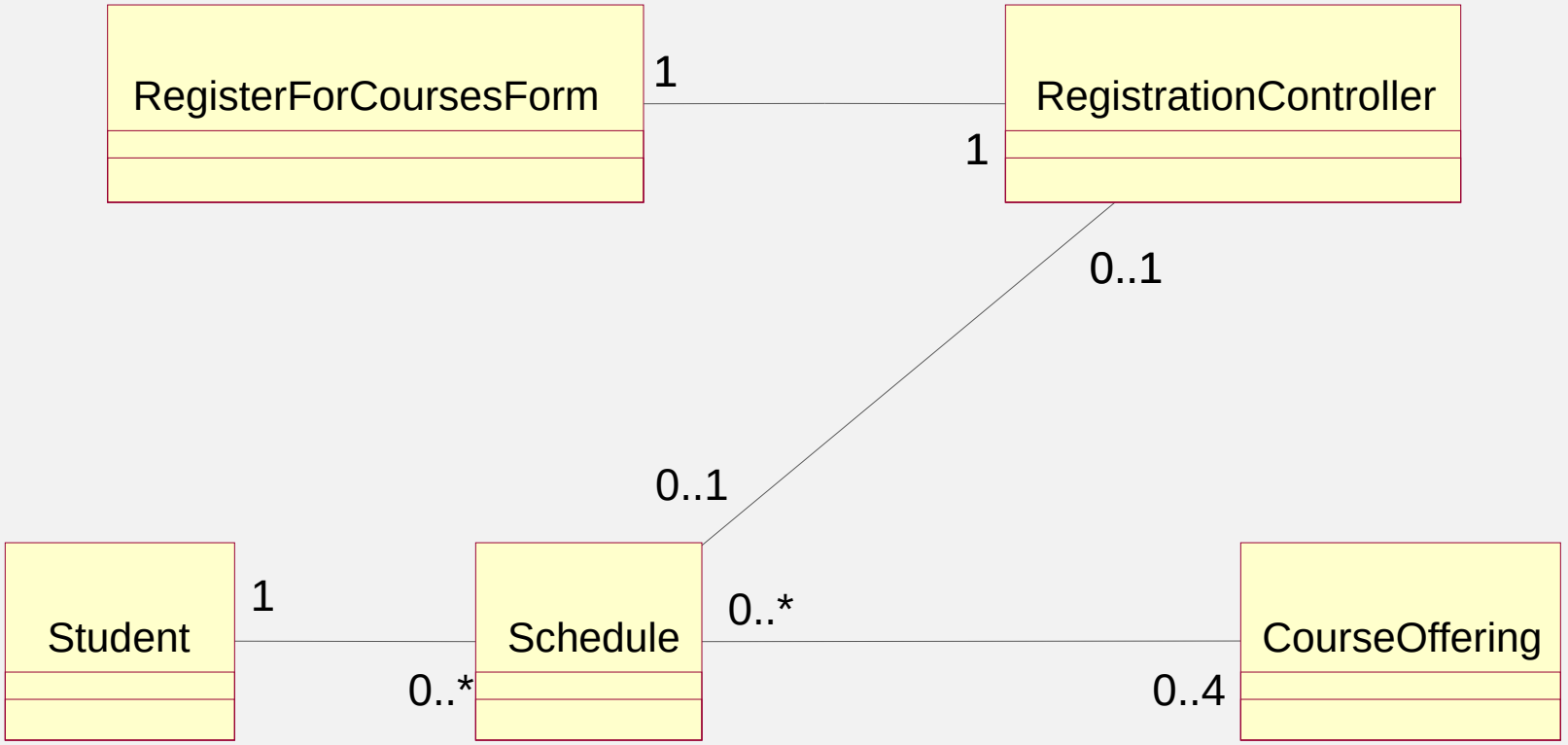
- Mối liên hệ ngữ nghĩa giữa hai hay nhiều lớp chỉ ra sự liên kết giữa các thể hiện của chúng
- Mối quan hệ về mặt cấu trúc chỉ ra các đối tượng của lớp này có kết nối với các đối tượng của lớp khác.



Bội số quan hệ

- Bội số quan hệ là số lượng thể hiện của một lớp liên quan tới MỘT thể hiện của lớp khác.
- Với mỗi liên kết, có hai bội số quan hệ cho hai đầu của liên kết.
  - Với mỗi đối tượng của Professor, có nhiều Course Offerings có thể được dạy.
  - Với mỗi đối tượng của Course Offering, có thể có 1 hoặc 0 Professor giảng dạy.

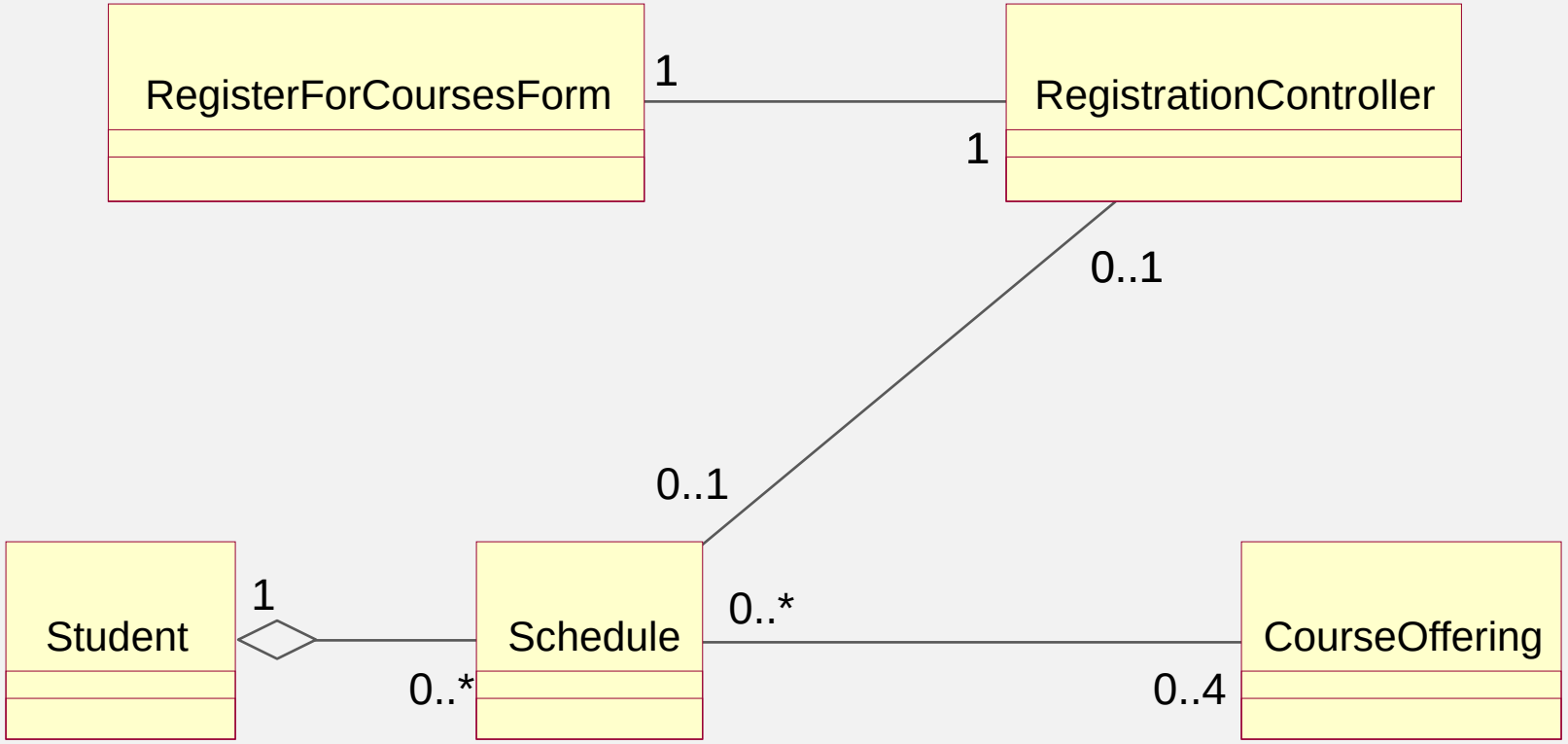
Ví dụ


RegisterForCoursesForm
1
RegistrationController
1
0..1
0..1
Student
1
Schedule
0..*
CourseOffering
0..*
0..4

Kết tập

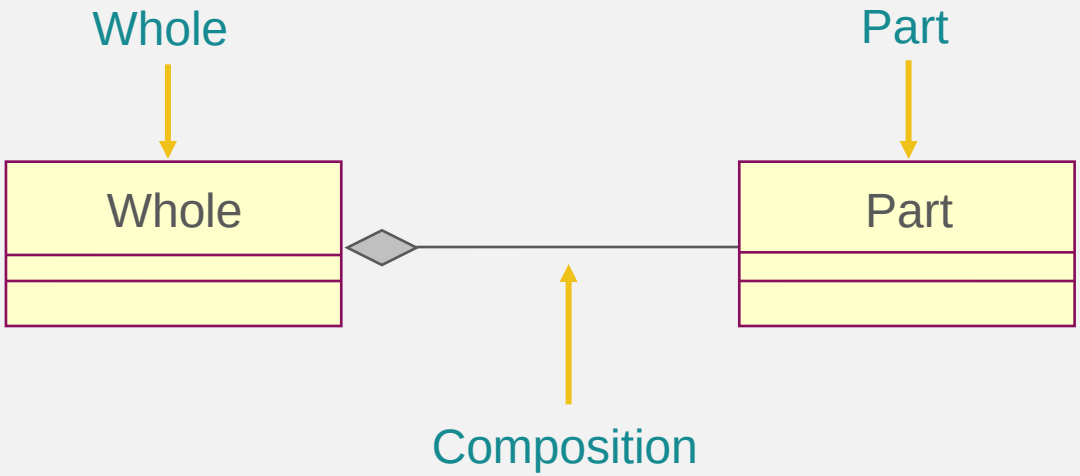
- Là một dạng đặc biệt của liên kết mô hình hóa mối quan hệ toàn thể-bộ phận (whole-part) giữa đối tượng toàn thể và các bộ phận của nó.
  - Kết tập là mối quan hệ "là một phần" ("is a part-of").
- Bội số quan hệ được biểu diễn giống như các liên kết khác

Ví dụ


RegisterForCoursesForm
1
RegistrationController
1
0..1
0..1
Student
1
Schedule
0..*
CourseOffering
0..*
0..4

Cấu thành

- Một dạng của kết tập với quyền sở hữu mạnh và các vòng đời trùng khớp giữa hai lớp
  - Whole sở hữu Part, tạo và hủy Part.
  - Part bị bỏ đi khi Whole bị bỏ, Part không thể tồn tại nếu Whole không tồn tại.



So sánh kết tập và cấu thành

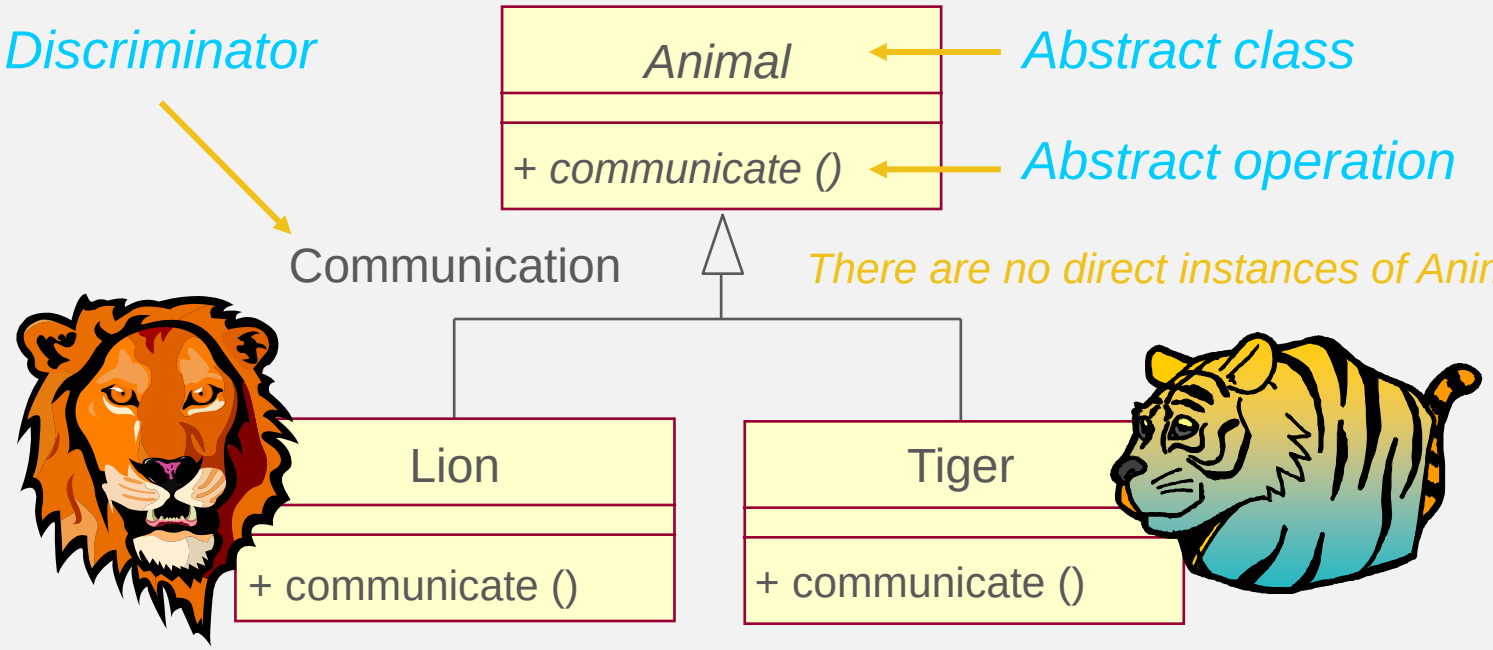
- Aggregation – University and Chancellor
  - Nếu không có trường Đại học (University), hiệu trưởng (Chancellor) không thể tồn tại.
  - Nếu không có Chancellor, University vẫn có thể tồn tại
- Composition – University and Faculty
  - University không thể tồn tại nếu không có các khoa (Faculty) và ngược lại (share time-life)
    + Thời gian sống của University gắn chặt với thời gian sống của Faculty
    + Nếu Faculties được giải phóng thì University không thể tồn tại và ngược lại

Tổng quát hóa (Generalization)

- Mối quan hệ giữa các lớp trong đó một lớp chia sẻ cấu trúc và/hoặc hành vi với một hoặc nhiều lớp khác
- Xác định sự phân cấp về mức độ trừu tượng hóa trong đó lớp con kế thừa từ một hoặc nhiều lớp cha
  - Đơn kế thừa (Single inheritance)
  - Đa kế thừa (Multiple inheritance)
- Là mối liên hệ "là một loại" ("is a kind of")

Lớp trừu tượng

- Lớp trừu tượng không thể có đối tượng
  - Chứa phương thức trừu tượng
  - Chữ nghiêng



Thank you!

Any questions?