# MÔ HÌNH HOÁ PHẦN MỀM

## TUẦN 2:
## USE CASE DIAGRAM

GVLT: NGUYỄN THỊ MINH TUYỀN

# NỘI DUNG

1. **Giới thiệu**
2. **Các thành phần**
3. **Mô tả use case**
4. **Best practices**

# NỘI DUNG

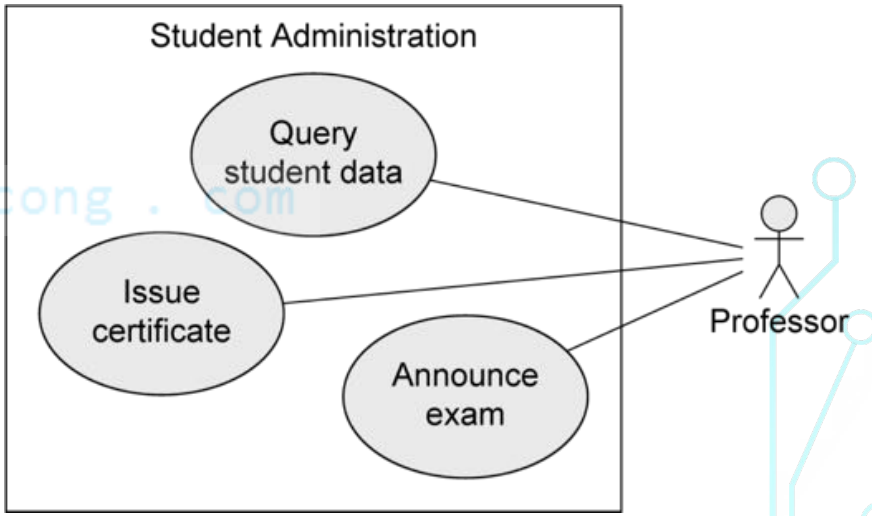
1. **Giới thiệu**
2. **Các thành phần**
3. **Mô tả use case**
4. **Best practices**

# GIỚI THIỆU

- Use case là một khái niệm nền tảng của nhiều phương pháp phát triển hướng đối tượng.
- Các biểu đồ use case biểu diễn mong muốn của khách hàng/stakeholders
  - Cần thiết cho một thiết kết chi tiết
- Biểu đồ use case được dùng trong suốt quá trình phân tích và thiết kế.
- Ta có thể sử dụng biểu đồ use case để trả lời các câu hỏi sau:
  - Cái gì đang được mô tả ? (hệ thống)
  - Ai tương tác với hệ thống? (các actor)
  - Các actor có thể làm gì? (các use case)

# VÍ DỤ: STUDENT ADMINISTRATION

- Hệ thống (Cái gì đang được mô tả?)
  - Student Administration
- Các actor (Ai tương tác với hệ thống?)
  - Professor
- Các use case (Các actor có thể làm gì?)
  - Query student data
  - Issue certificate
  - Announce exam

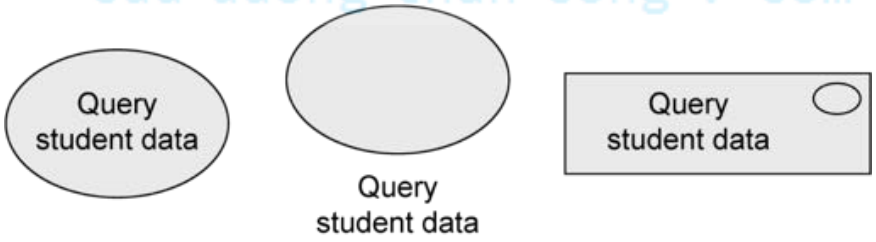


# NỘI DUNG

1. Giới thiệu
2. **Các thành phần**
3. Mô tả use case
4. Best practices

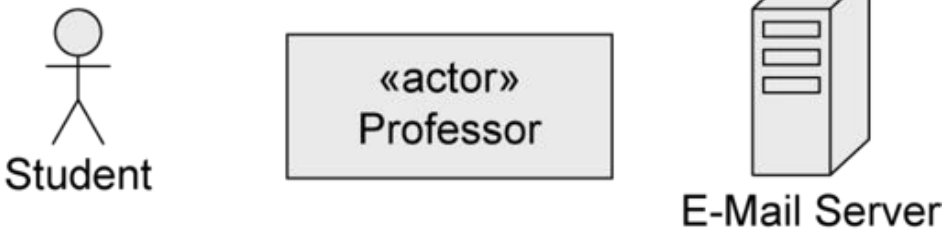
# USE CASE


- Mô tả chức năng mong đợi từ hệ thống đang phát triển.
- Cung cấp một lợi ích hữu hình cho một hoặc nhiều actor tương tác với use case này.
- Xuất phát từ các mong muốn thu thập được từ khách hàng
- Tập hợp tất cả các use case mô tả chức năng mà hệ thống sẽ cung cấp → có thể được sử dụng làm tài liệu về chức năng mà hệ thống cung cấp
- Các ký hiệu thay thế:



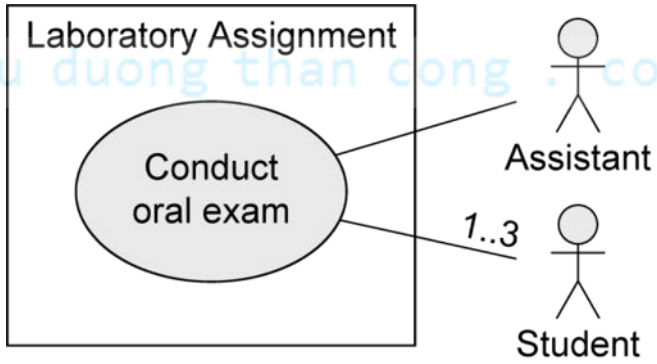
# ACTOR [1]


- Các actor tương tác với hệ thống ...
  - bằng cách sử dụng các use case, nghĩa là các actor bắt đầu thực hiện các use case
  - bằng cách được sử dụng bởi các use case, nghĩa là các actor cung cấp chức năng cho việc thực thi của các use case
- Các actor biểu diễn vai trò mà user (người dùng) chấp nhận.
  - Các user có thể chấp nhận và thiết lập nhiều vai trò cùng lúc.
- Các actor <u>không phải là một phần của hệ thống</u>, nghĩa là họ nằm ngoài ranh giới hệ thống.
- Ký hiệu thay thế:



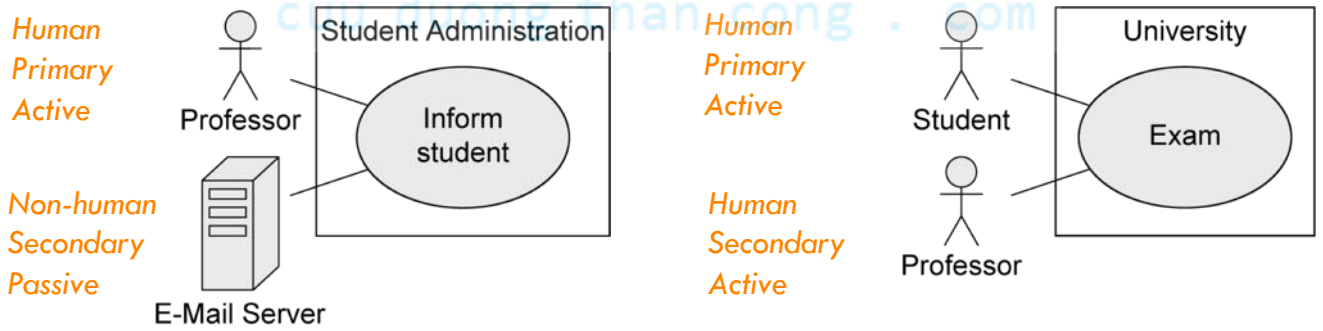
# ACTOR [2]


- Thông thường, dữ liệu người dùng cũng được quản lý trong hệ thống. Dữ liệu này được mô hình hoá trong hệ thống dưới dạng các đối tượng và các lớp.
- Ví dụ: Actor Assistant
  - Actor **Assistant** tương tác với hệ thống **Laboratory Assignment** bằng cách sử dụng nó.
  - Lớp **Assistant** mô tả các đối tượng biểu diễn dữ liệu người dùng (ví dụ: name, ssNr, ...)

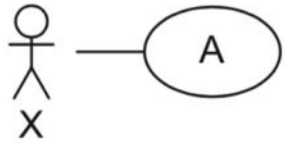


# CÁC LOẠI ACTOR

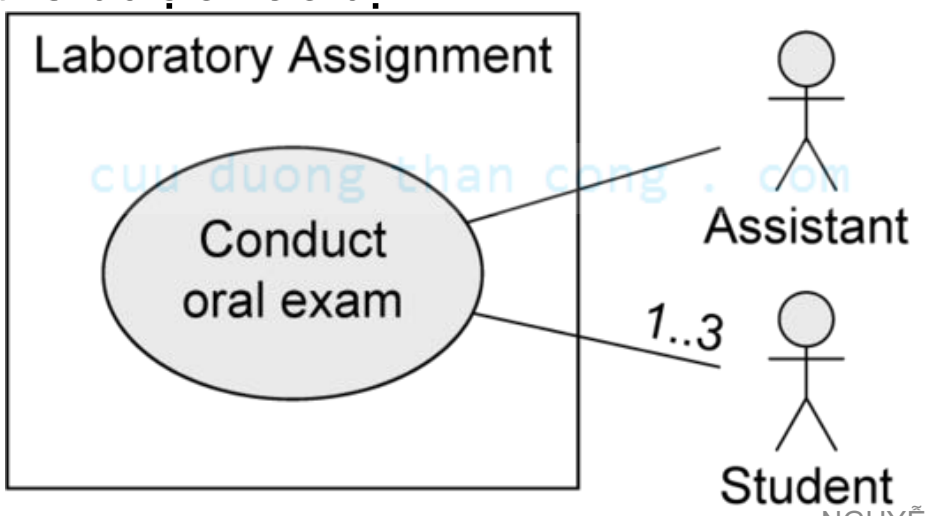

- **Human**
  - Ví dụ: **`Student`**, **`Professor`**
- **Non-human**
  - Ví dụ: **`E-Mail Server`**
- **Primary**: nhận lợi ích trực tiếp từ việc thực hiện use case
- **Secondary**: không nhận lợi ích trực tiếp
- **Active**: bắt đầu thực hiện use case
- **Passive**: cung cấp chức năng để thực hiện use case



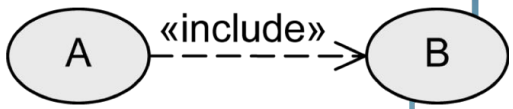
# QUAN HỆ GIỮA USE CASE VÀ ACTOR



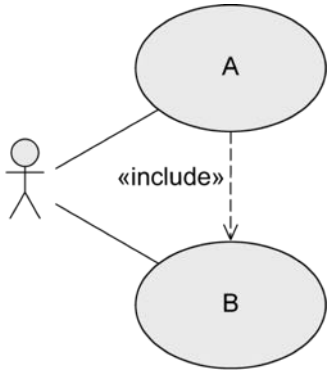

- Các actor được kết nối với use case bằng các đường thẳng (association – liên kết): actor giao tiếp với hệ thống và sử dụng một chức năng nào đó.
- Mỗi actor phải giao tiếp với ít nhất một use case.
- Một association luôn là nhị phân: luôn đặc tả một use case và một actor
- Multiplicity có thể được xác định



# QUAN HỆ GIỮA CÁC USE CASE




- Hành vi của một use case (included use case) được tích hợp vào hành vi của một use case khác (base use case).



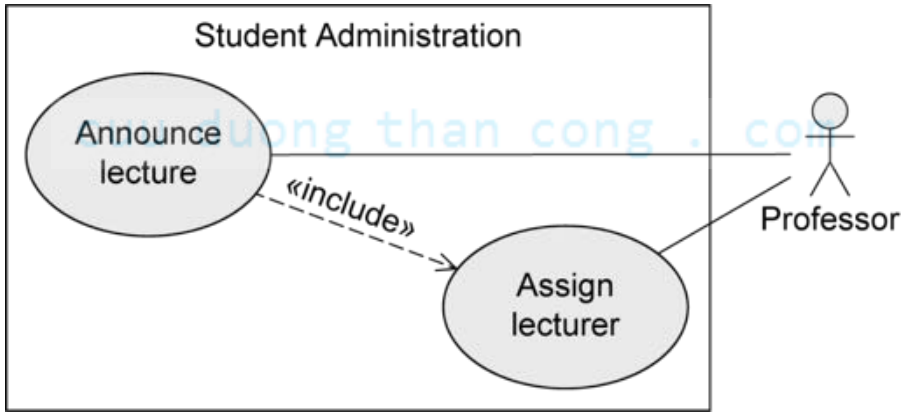

*Base use case*

yêu cầu hành vi của included use case
có khả năng cung cấp chức năng của nó

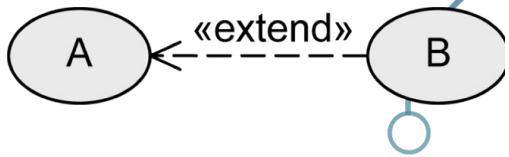
*Included use case*

có thể tự thực thi

- Ví dụ:

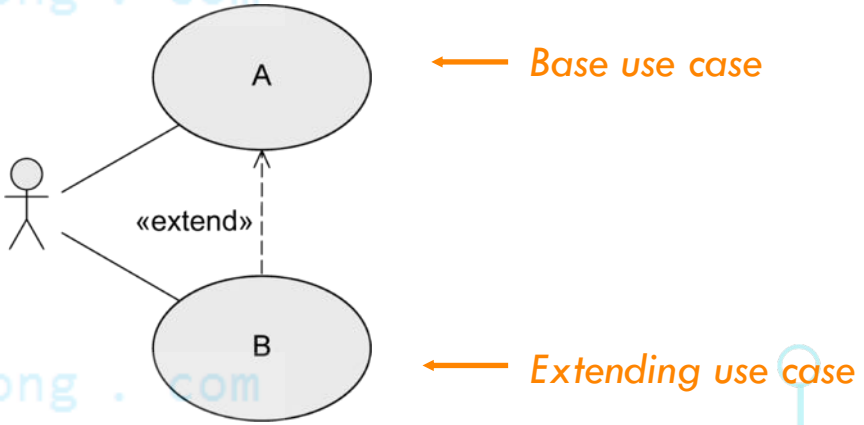


# QUAN HỆ GIỮA CÁC USE CASE

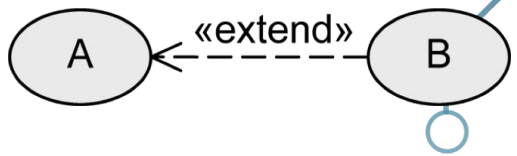



- Hành vi của một use case (extending use case) có thể được tích hợp vào hành vi của một use case khác (base use case) nhưng **không bắt buộc**.
- Cả hai use case có thể được thực thi độc lập với nhau.
- B có thể được kích hoạt bởi A

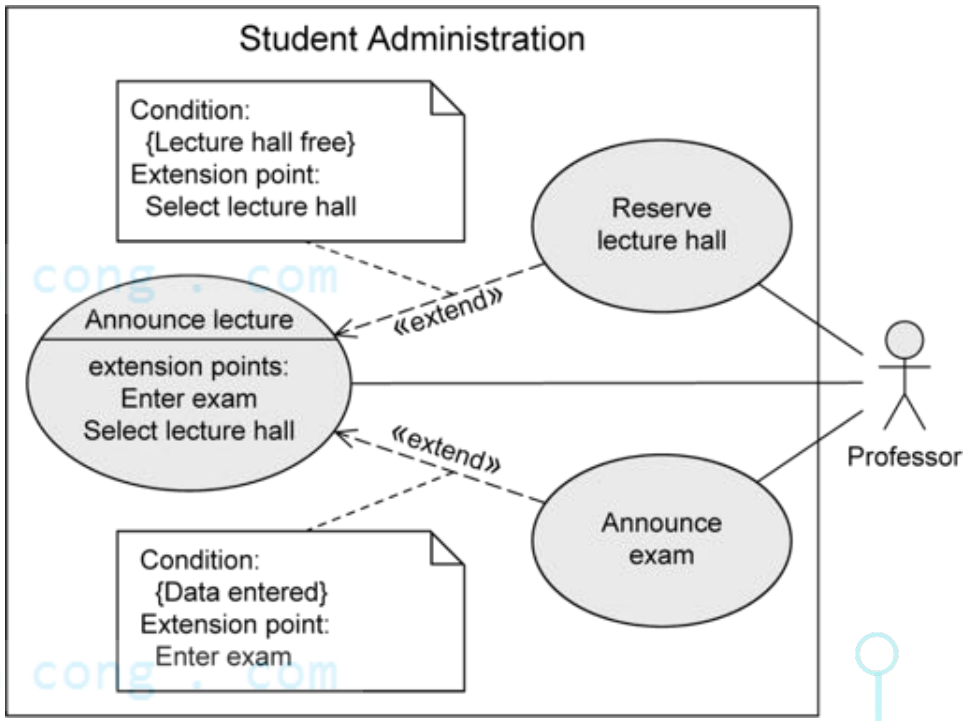
để thêm hành vi của B vào A.



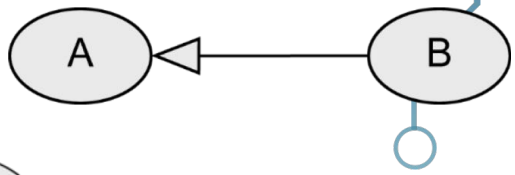
# QUAN HỆ GIỮA HAI USE CASE



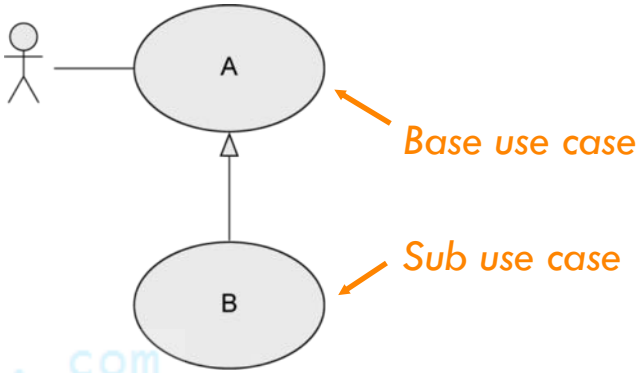

- Một condition là điều kiện phải thoả mãn để base use case tích hợp hành vi của extending use case.
- Các điểm mở rộng (Extension point) định nghĩa điểm mà tại đó hành vi của extending use case phải được tích hợp vào base use case.
- Các điểm mở rộng được viết trực tiếp trong use case.
- Có thể đặc tả nhiều điểm mở rộng.

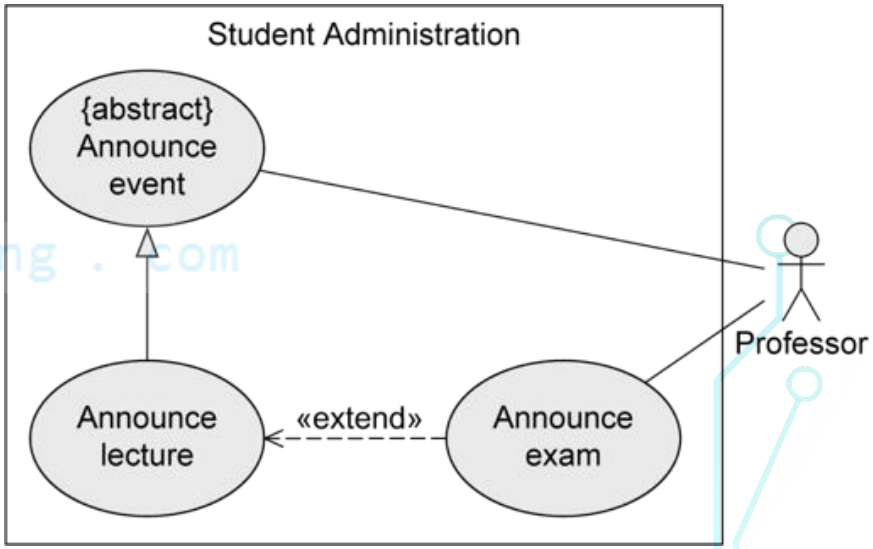


# QUAN HỆ GIỮA CÁC USE CASE

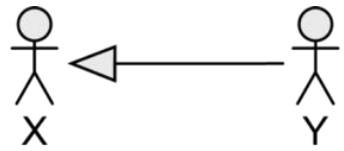



- Use case A tổng quát hơn use case B.
- B kế thừa hành vi của A và có thể hoặc mở rộng hoặc ghi đè lên nó.
- B cũng có thể kế thừa tất cả các quan hệ từ A.
- B sử dụng chức năng cơ bản của A nhưng tự quyết định phần nào của A được thực thi hoặc thay đổi.
- A có thể được gán nhãn {abstract}
  - Không thể được thực thi trực tiếp
  - Chỉ B có thể thực thi được.

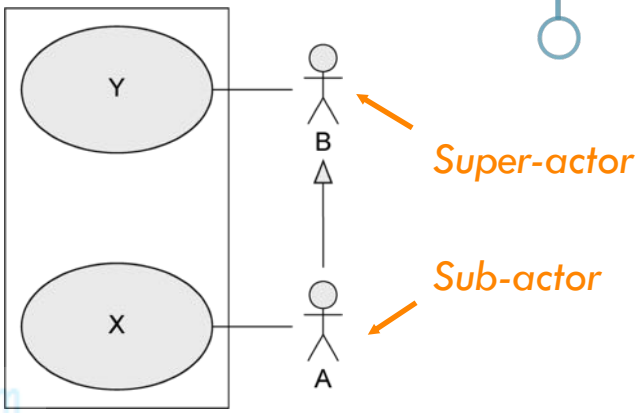


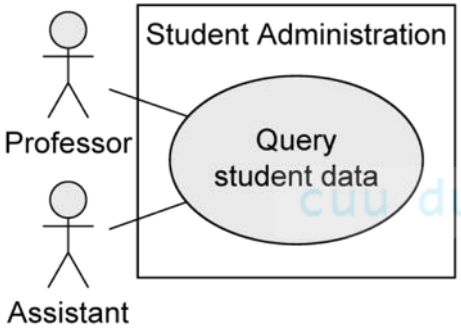



# QUAN HỆ GIỮA CÁC ACTOR

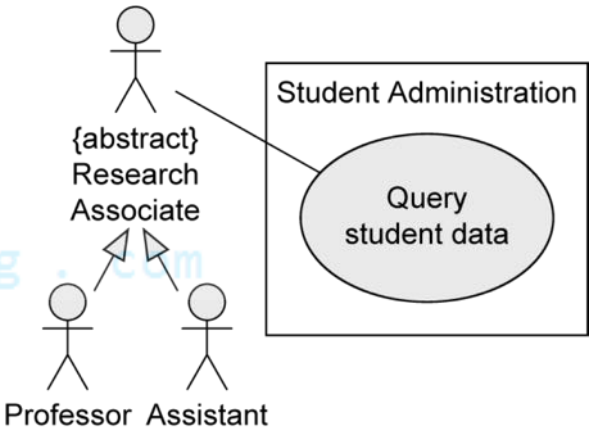



- Actor A kế thừa từ actor B.
- A có thể giao tiếp với X và Y.
- B chỉ có thể giao tiếp với Y.
- Cho phép đa kế thừa.
- Có thể có abstract actor.
- Ví dụ:



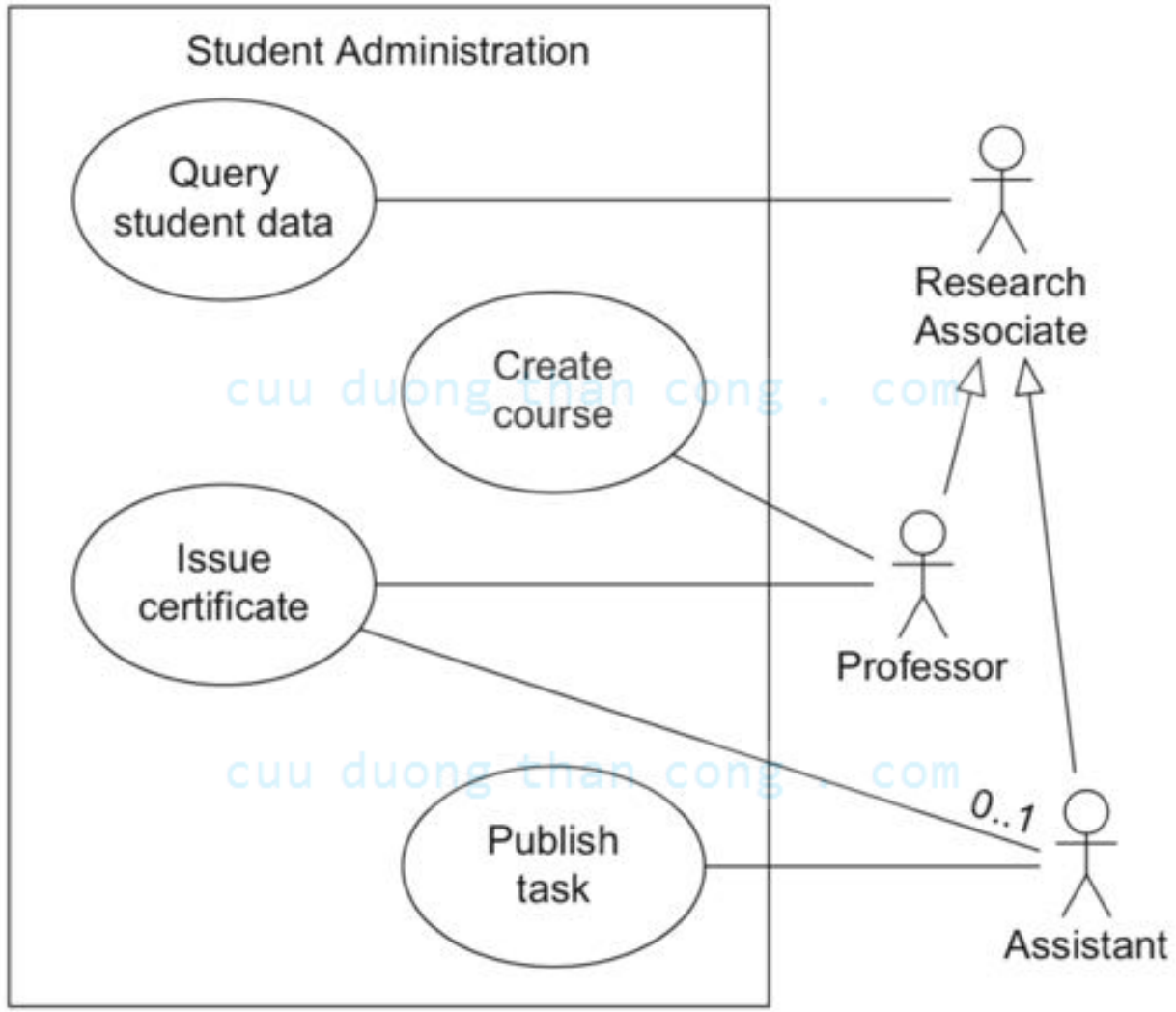

≠




**Professor** AND **Assistant** cần thực hiện **Query student data**

**Professor** OR **Assistant** cần thực hiện **Query student data**

# VÍ DỤ



# NỘI DUNG

1. Giới thiệu
2. Các thành phần
3. **Mô tả use case**
4. Best practices

# MÔ TẢ USE CASE

Phương pháp có cấu trúc

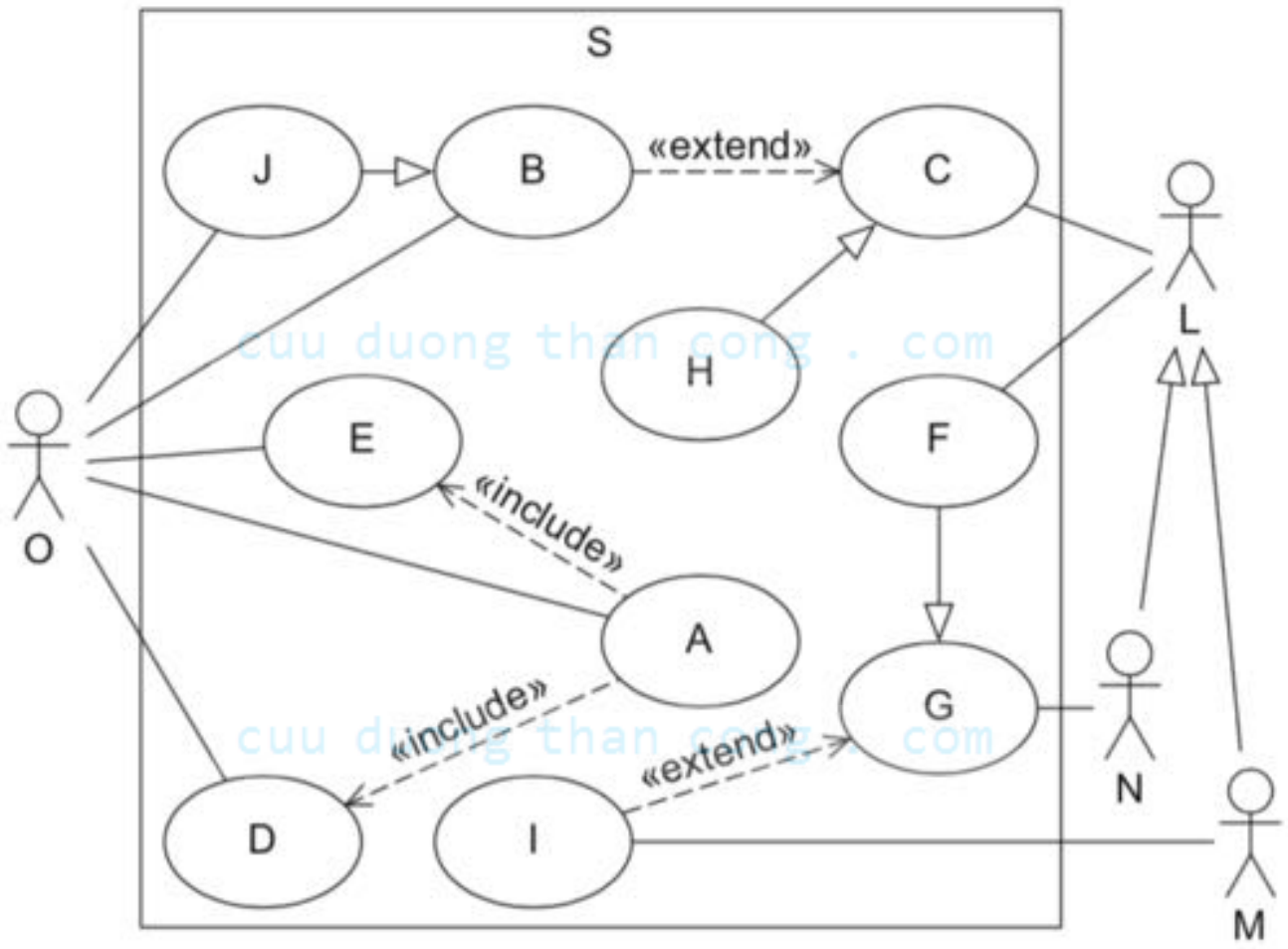
- Name
- Short description
- Precondition: prerequisite for successful execution
- Postcondition: system state after successful execution
- Error situations: errors relevant to the problem domain
- System state on the occurrence of an error
- Actors that communicate with the use case
- Trigger: events which initiate/start the use case
- Standard process: individual steps to be taken
- Alternative processes: deviations from the standard process

[A. Cockburn: Writing Effective Use Cases, Addison Wesley, 2000]

# VÍ DỤ

| | |
|---|---|
| **Name** | **Reserve lecture hall** |
| **Short description** | An employee reserves a lecture hall at the university for an event. |
| **Precondition** | The employee is authorized to reserve lecture halls. |
| **Postcondition** | A lecture hall is reserved. |
| **Error situations** | There is no free lecture hall. |
| **System state in the event of an error** | The employee has not reserved a lecture hall. |
| **Actors** | Employee |
| **Trigger** | Employee requires a lecture hall. |
| **Standard process** | (1) Employee logs in to the system.<br>(2) Employee selects the lecture hall.<br>(3) Employee selects the date.<br>(4) System confirms that the lecture hall is free.<br>(5) Employee confirms the reservation. |
| **Alternative processes** | (4') Lecture hall is not free.<br>(5') System proposes an alternative lecture hall.<br>(6') Employee selects alternative lecture hall and confirms the reservation. |

# VÍ DỤ VỀ CÁC MỐI QUAN HỆ

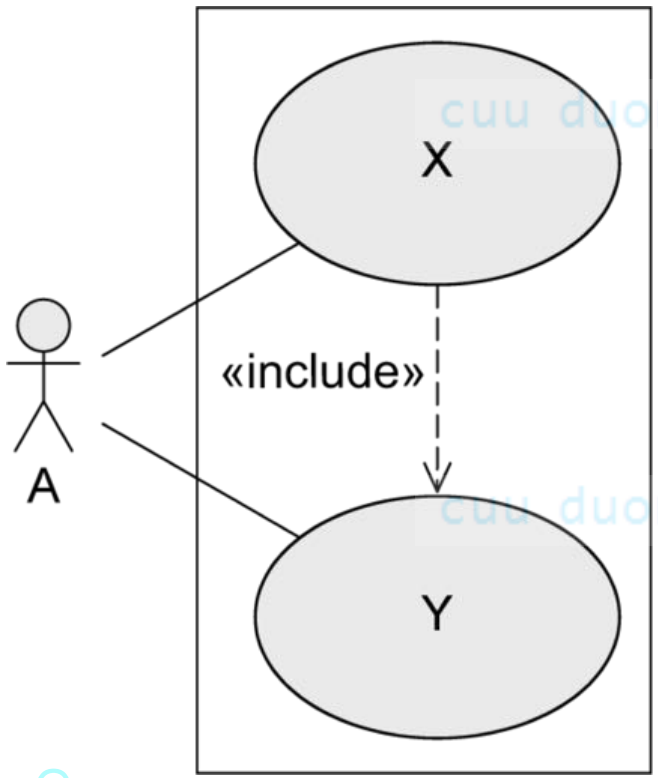


# NỘI DUNG

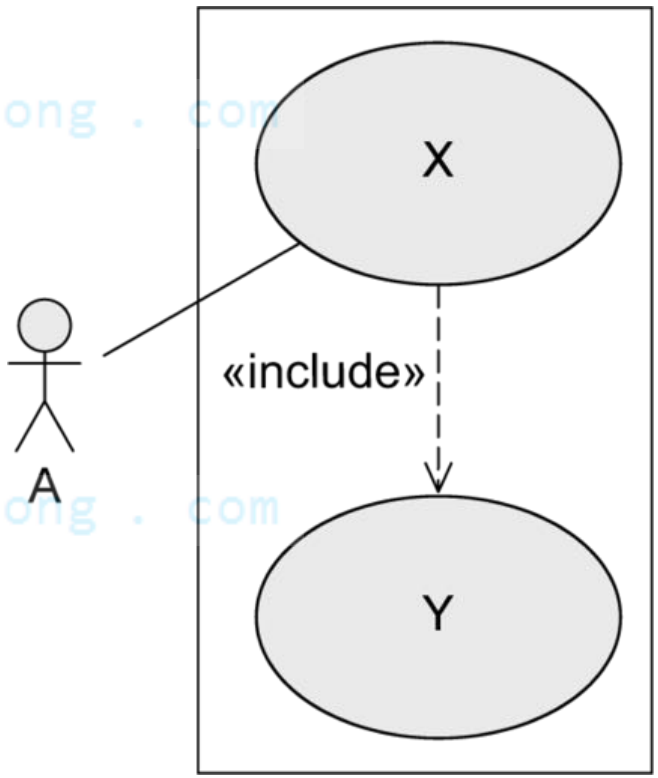
1. Giới thiệu
2. Các thành phần
3. Mô tả use case
4. **Best practices**

# <<include>>

*UML standard*

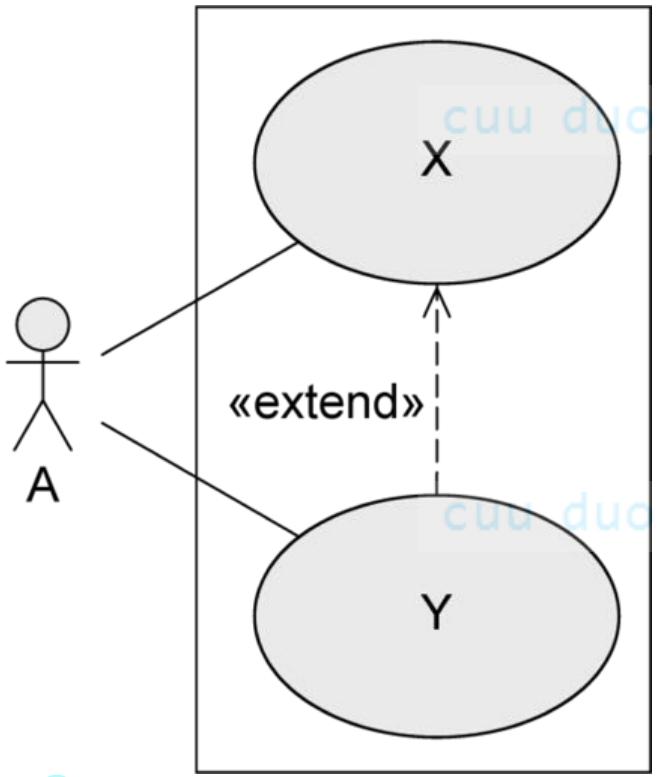



*Best practice*

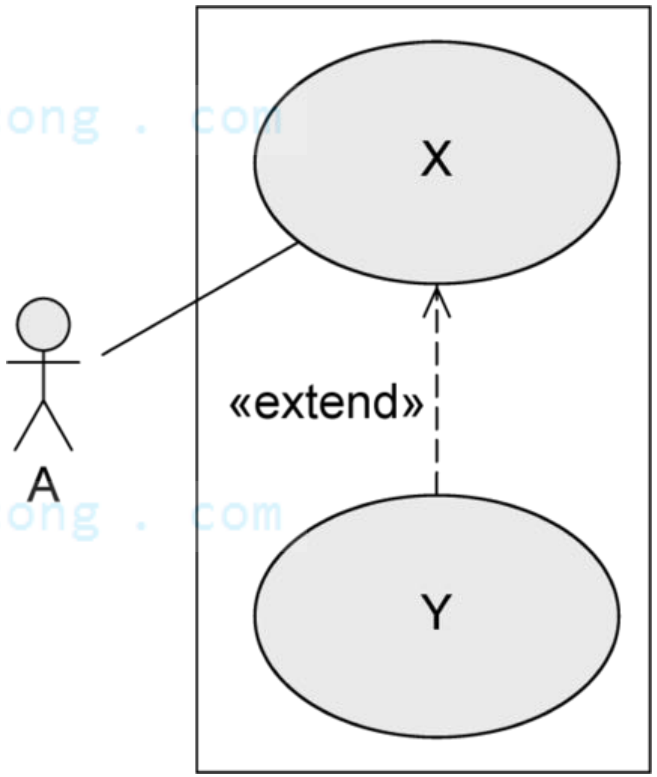


# <<extend>>

*UML standard*




*Best practice*



# NHẬN DIỆN ACTOR

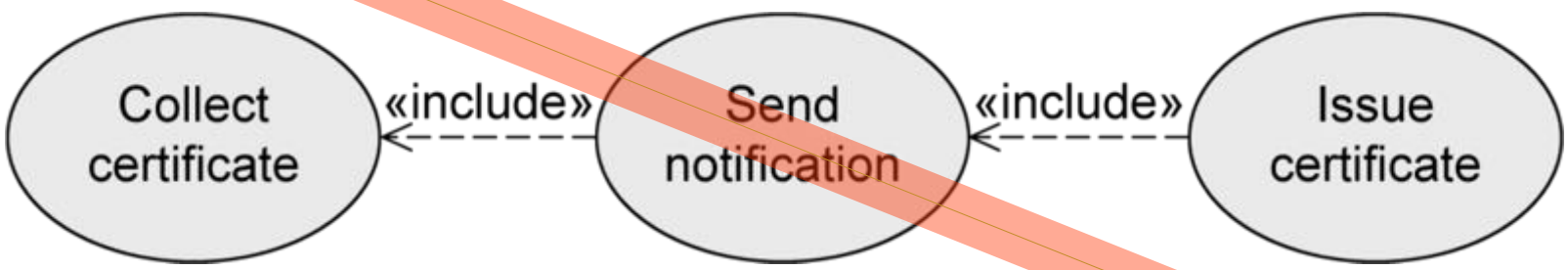
- Ai sử dụng các use case chính?
- Ai cần hỗ trợ cho công việc hàng ngày của họ?
- Ai chịu trách nhiệm quản trị hệ thống?
- Các thiết bị/hệ thống (phần mềm) mà hệ thống cần phải giao tiếp là gì?
- Ai quan tâm đến kết quả của hệ thống?

# NHẬN DIỆN USE CASE

- Các tác vụ chính mà một actor phải thực hiện là gì?
- Actor có muốn truy vấn hay thay đổi thông tin có trong hệ thống không?
- Actor có muốn thông báo cho hệ thống về những thay đổi trong các hệ thống khác không?
- Actor có nên được thông báo về các sự kiện bất ngờ trong hệ thống không?

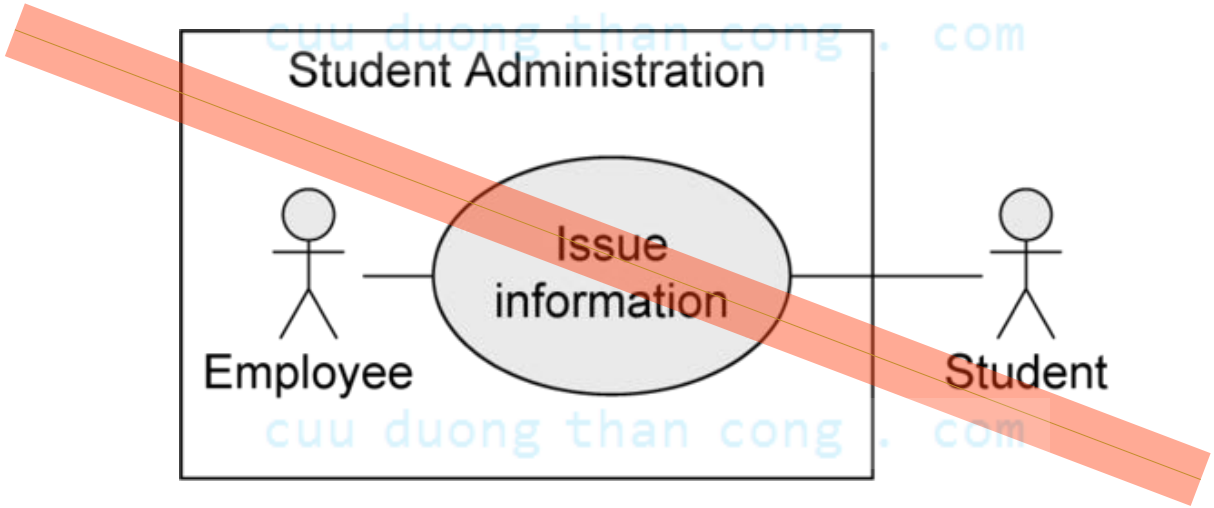
# LỖI THÔNG DỤNG CẦN TRÁNH (1/5)

- Biểu đồ use case không mô hình hoá các tiến trình/workflows



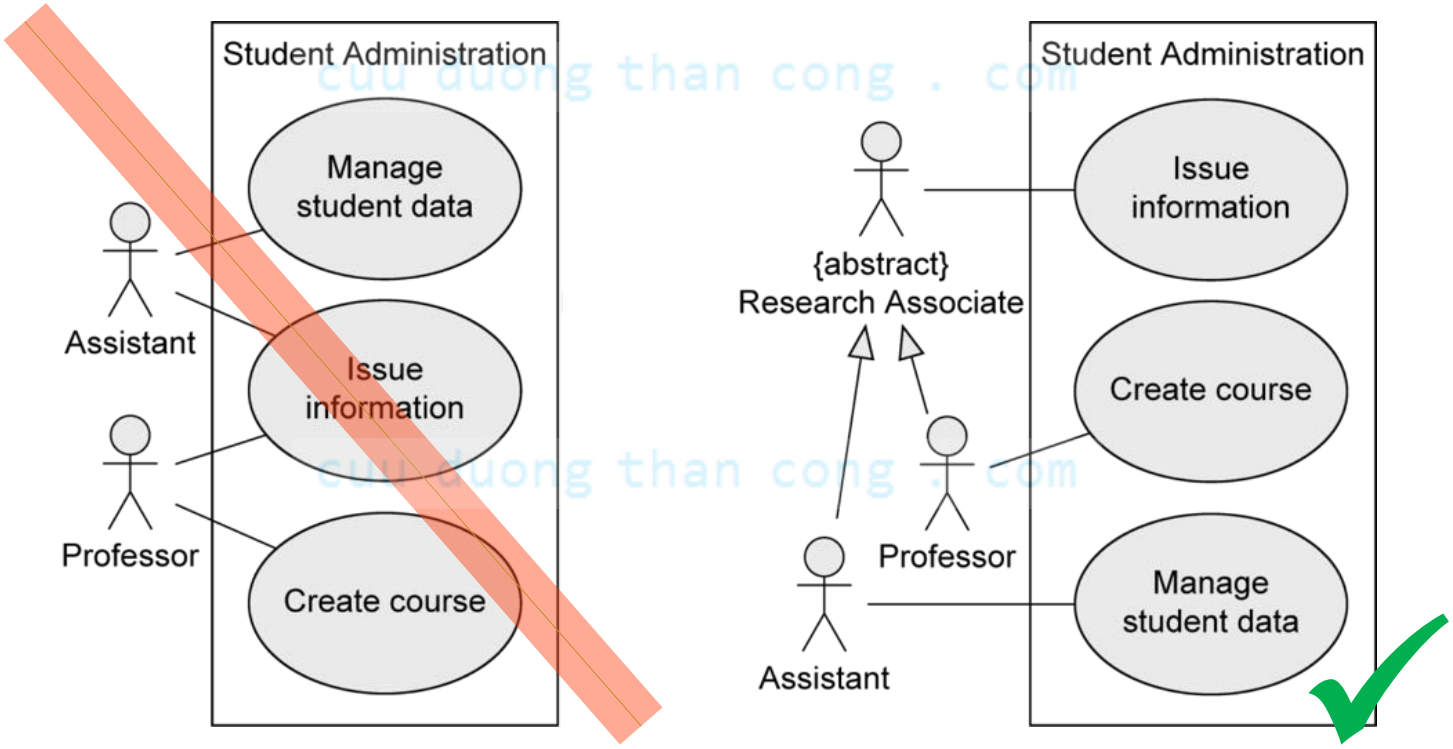
# LỖI THÔNG DỤNG CẦN TRÁNH (2/5)

- Các actor không phải là một phần của hệ thống, vì vậy chúng được đặt ngoài các boundary.



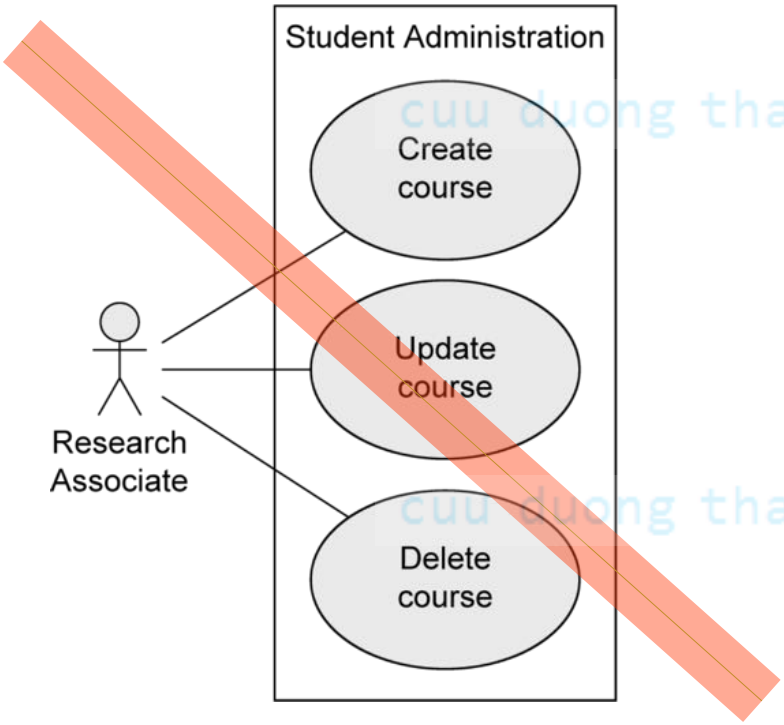
# LỖI THÔNG DỤNG CẦN TRÁNH (3/5)

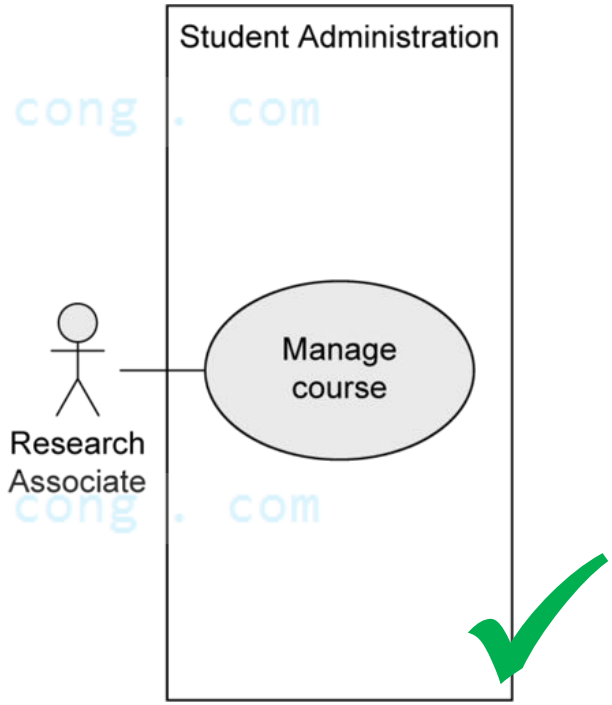
- Use case `Issue information` cần **hoặc** một actor `Assistant` **hoặc** một `Professor` để thực thi.



# LỖI THÔNG DỤNG CẦN TRÁNH (4/5)

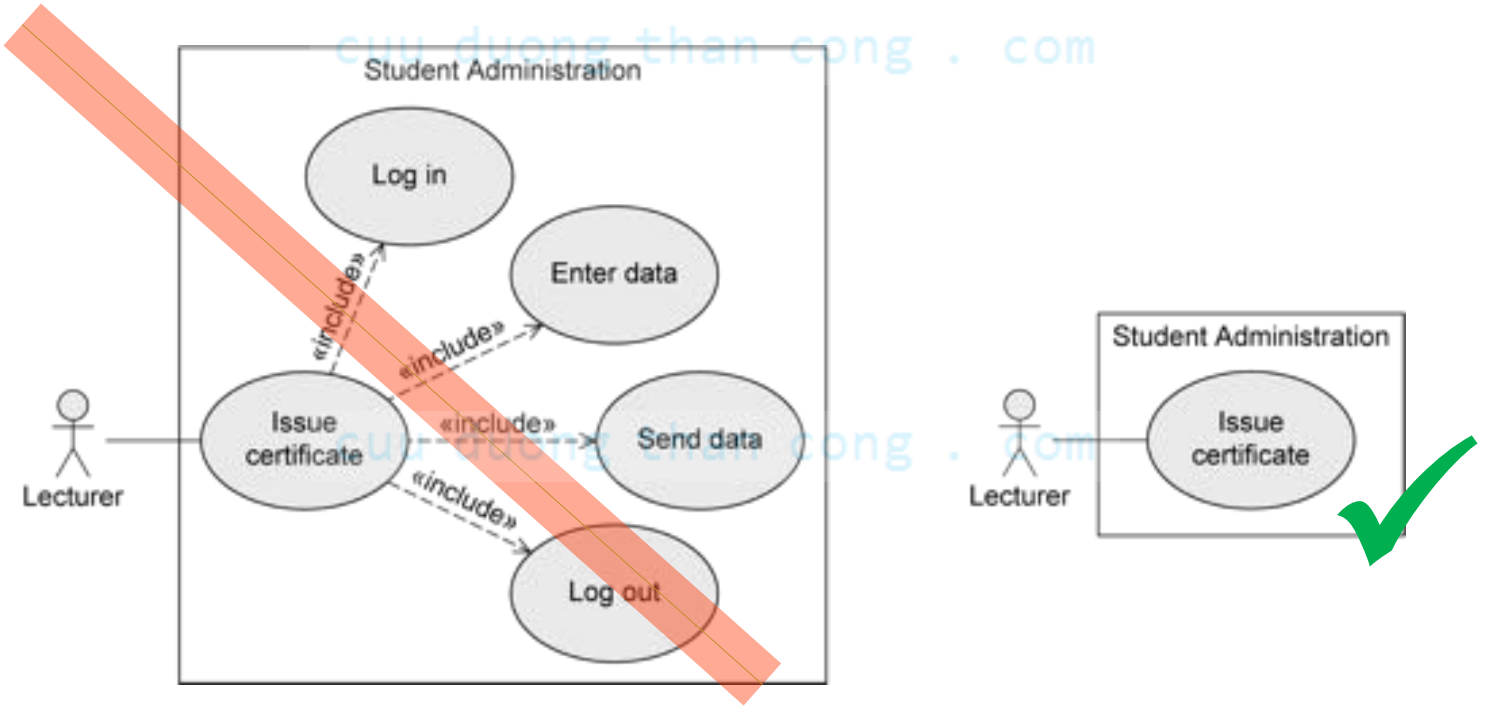
- Nhiều use case nhỏ có cùng mục tiêu có thể gom nhóm thành dạng một use case.



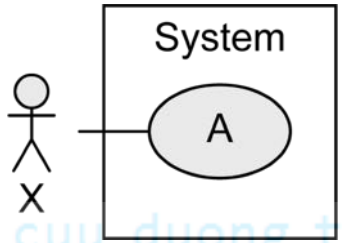



# LỖI THÔNG DỤNG CẦN TRÁNH (5/5)

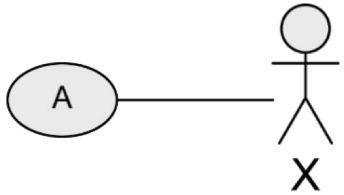
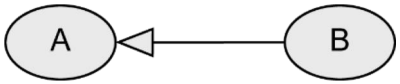
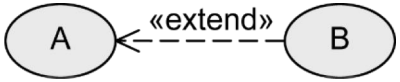
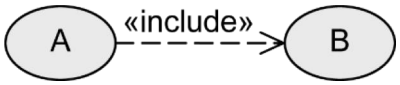
- Các bước khác nhau là một phần của use case, không tách rời thành nhiều use case -> KHÔNG phân rã chức năng.



# CÁC THÀNH PHẦN KÝ HIỆU (1/2)

| Tên | Ký hiệu | Mô tả |
|---|---|---|
| System |  | Ranh giới (Boundary) giữa hệ thống và người sử dụng hệ thống |
| Use case |  | Đơn vị chức năng của hệ thống |
| Actor |  | Vai trò của người dùng hệ thống |

# CÁC THÀNH PHẦN KÝ HIỆU (2/2)

| Tên | Ký hiệu | Mô tả |
|---|---|---|
| Association |  | Quan hệ giữa use case và actor |
| Generalization |  | Quan hệ kế thừa giữa các actor và giữa các use case. |
| Extend relationship |  | B extends A: tuỳ chọn sử dụng use case B bởi use case A |
| Include relationship |  | A includes B: bắt buộc sử dụng use case B bởi use case A |

# THAM KHẢO

- **UML @ Classroom, An Introduction to Object-Oriented Modeling**, *Martina Seidl, Marion Scholz, Christian Huemer, Gerti Kappel*, Springer International Publishing, 2015
- **Slide của sách UML @ Classroom, An Introduction to Object-Oriented Modeling**, http://www.uml.ac.at/wp-content/uploads/2012/05/01_UseCaseDiagram_slides_2015.pptx

# VÍ DỤ: INFORMATION SYSTEM OF THE STUDENT OFFICE OF A UNIVERSITY

- Many important administrative activities of a university are processed by the student office. Students can register for studies (matriculation), enroll, and withdraw from studies here. Matriculation involves enrolling, that is, registering for studies.
- Students receive their certificates from the student office. The certificates are printed out by an employee. Lecturers send grading information to the student office. The notification system then informs the students automatically that a certificate has been issued.
- There is a differentiation between two types of employees in the student office: a) those that are exclusively occupied with the administration of student data (service employee, or ServEmp), and b) those that fulfill the remaining tasks (administration employee, or AdminEmp), whereas all employees (ServEmp and AdminEmp) can issue information.
- Administration employees issue certificates when the students come to collect them. Administration employees also create courses. When creating courses, they can reserve lecture halls.