MÔ HÌNH HÓA YÊU CẦU

Các phương pháp dựa trên kịch bản

NỘI DUNG

- Các tiếp cận thiết kế
- Quy luật phân tích Thumb
- Mô hình hóa chức năng
  - Sơ đồ usecase
  - Sơ đồ activity

Mục tiêu của mô hình hóa yêu cầu

- Mô hình hóa những cái gì được biết và được sử dụng - "Cái gì?" - chứ không phải "Như thế nào"
  - Những đối tượng nào mà hệ thống xử lý?
  - Những chức năng nào mà hệ thống phải thực hiện?
  - Những hành vi nào mà hệ thống thể hiện?
  - Giao diện nào được xác định?
  - Những hạn chế nào cần áp dụng?

Các tiếp cận thiết kế

- BDUF (Big Design Up Front)
- NDUF (No Design Up Front)
- EDUF (Enough Design Up Front)

Big Design Upfront

- Thiết kế đầy đủ ở thời điểm bắt đầu

Ưu điểm:

- Hiệu quả cao khi yêu cầu ban đầu rõ ràng
- Có cái nhìn tổng quan về hệ thống ngay từ đầu
- Chi phí được tính toán chính xác → an toàn khi lập kế hoạch

Big Design Upfront

Hạn chế:

- Nếu có thay đổi → thiết kế lại toàn bộ → tốn rất nhiều thời gian và chi phí
- Mất thời gian và chi phí để thiết kế các vấn đề/yêu cầu không cần thiết
- Làm cho hệ thống phức tạp hơn

NDUF (No Design Up Front)

- Bắt đầu phát triển phần mềm mà không cần mất thời gian cho thiết kế.
- Trong quá trình phát triển, các đầu vào và ý tưởng mới liên tục được đưa ra và thực hiện.
- Ưu điểm: linh hoạt khi phát triển
- Hạn chế:
  - Đòi hỏi đội ngũ phát triển phải có nhiều kinh nghiệm
  - Nguy cơ kiến trúc không ổn định
  - Đầu tư vào một số nội dung dư thừa hoặc không sử dụng được

NDUF (No Design Up Front)

"Big design up front is **dumb**, but doing no design up front is even dumber." (BDUF is stupid, but NDUF is **even more stupid.**)

(**Dave Thomas**, one of the authors of the “Agile Manifesto”)

https://www.agileagreement.com/2020/07/21/enough-design-upfront-vor-big-design-upfront/

EDUF (Enough Design Up Front)

- Thiết kế các nội dung quan trọng/đủ cần thiết cho quá trình phát triển.
- Các nội dung chi tiết được bổ sung trong quá trình phát triển mà không làm ảnh hưởng đến tiến độ/chi phí xây dựng hệ thống.

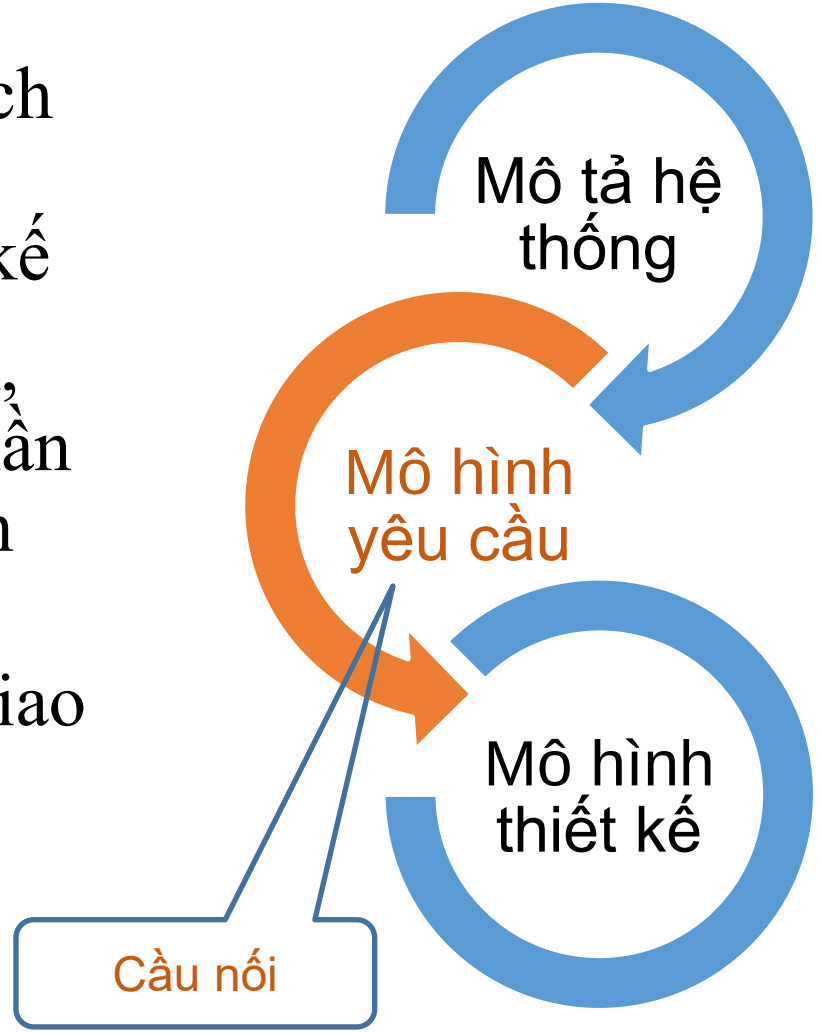
EDUF (Enough Design Up Front)

Ưu điểm:

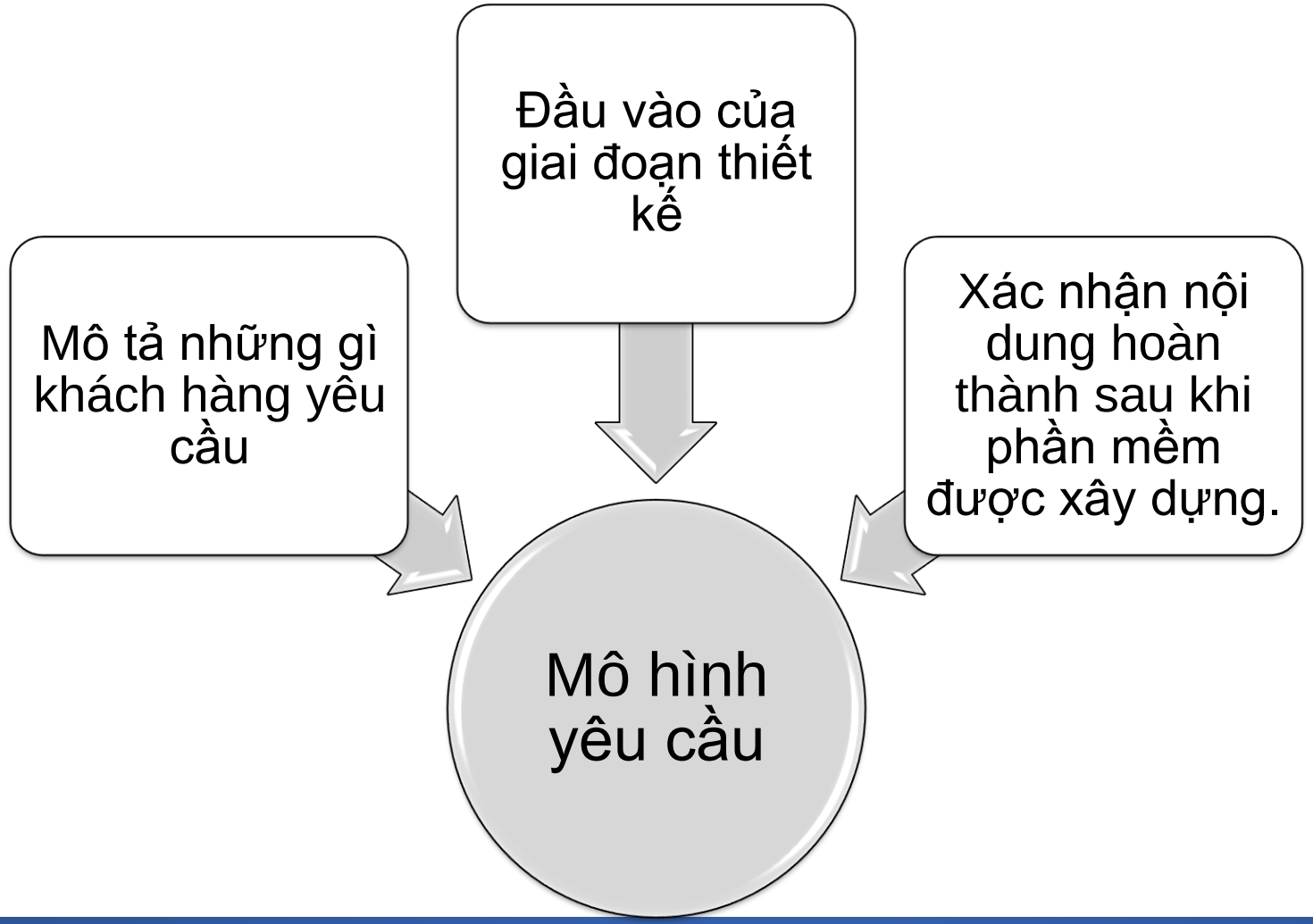
- Kết hợp các khía cạnh của BDUF và NDUF
- Thông qua việc xác định sơ bộ các nhu cầu, phạm vi của dự án có thể được xác định sơ bộ và ngoại suy từ các nhu cầu quan trọng nhất đến các nhu cầu khác
- Do đó, có thể điều chỉnh hợp đồng về phạm vi dịch vụ, ngày giao hàng và chi phí, …

Mục tiêu chính của mô hình yêu cầu

- Mô hình yêu cầu còn được gọi là mô hình phân tích
- Là cầu nối giữa mô tả hệ thống và mô hình thiết kế
  - Mô tả hệ thống: mô tả tổng quan về phần mềm, những chức năng nghiệp vụ của phần mềm, phần cứng, dữ liệu, người dùng, những hệ thống liên quan khác
  - Mô hình thiết kế: mô tả kiến trúc phần mềm, giao diện, dữ liệu, xử lý



Mục tiêu chính của mô hình yêu cầu



Quy luật phân tích Thumb

Được Arlow and Neustadt [Arl02] đề nghị các quy luật đáng giá trong lĩnh vực phân tích yêu cầu phần mềm.

1. Mô hình nên tập trung vào các yêu cầu có thể nhìn thấy trong hệ thống phần mềm hoặc nghiệp vụ.
2. Mỗi yếu tố của mô hình yêu cầu phải mô tả tổng thể phần mềm hoặc các miền thông tin, chức năng và hành vi của phần mềm
3. Là nội dung xác định cơ sở hạ tầng và các mô hình phi chức năng khác cho đến khi có thiết kế.

Quy luật phân tích Thumb

Được Arlow and Neustadt [Arl02] đề nghị các quy luật đáng giá trong lĩnh vực phân tích yêu cầu phần mềm.

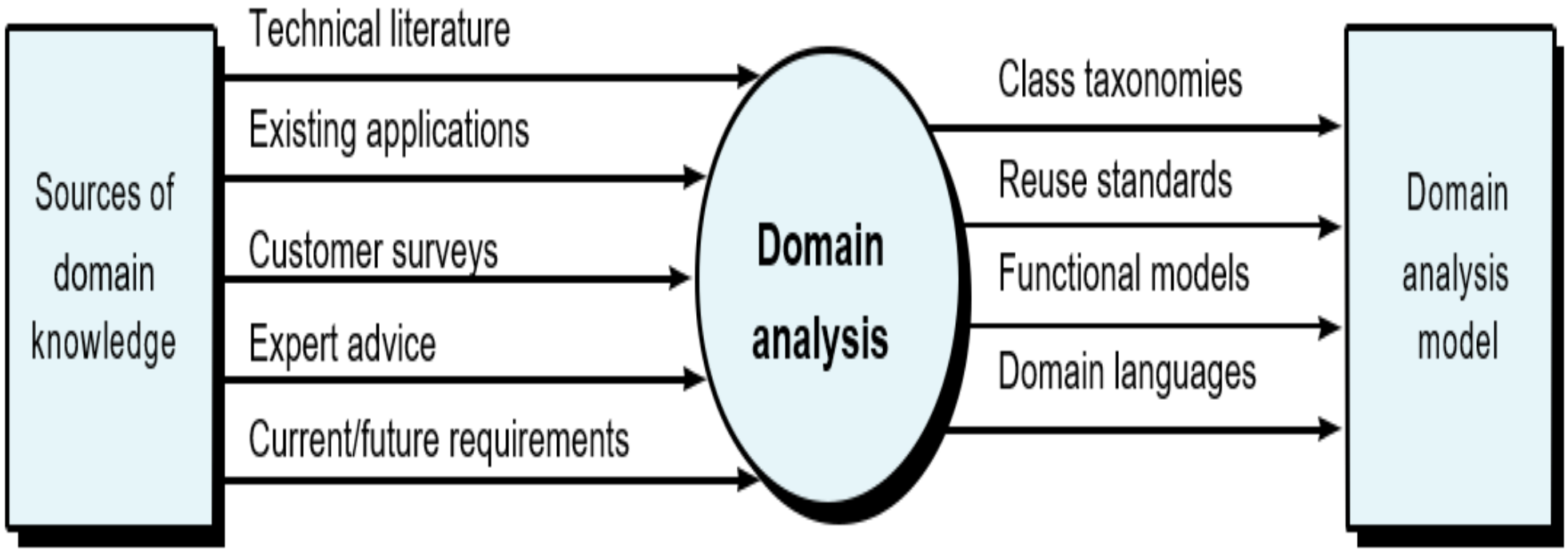
4. Giảm thiểu khớp nối trong toàn hệ thống (biểu diễn các mối quan hệ giữa các lớp và các hàm)
5. Mô hình yêu cầu phải có giá trị cho tất cả các bên liên quan.
6. Mô hình càng đơn giản càng tốt.

Kỹ thuật thu thập yêu cầu

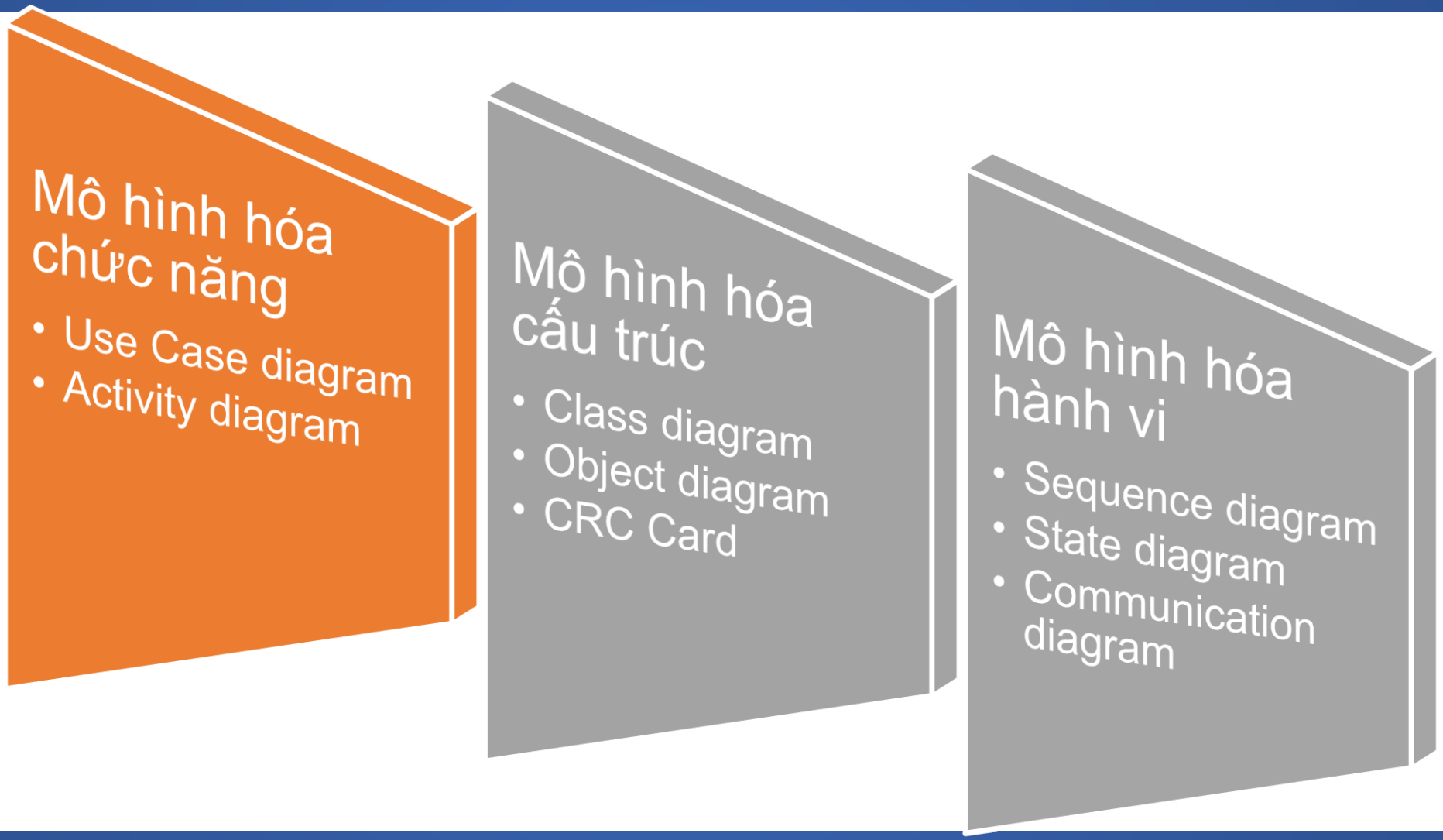
- Interviews
- Questionnaires or Surveys
- User Observation
- Document Analysis
- Interface analysis
- Workshops
- Brainstorming
- Role-play
- Use Cases and Scenarios
- Focus Groups
- Prototyping

Phân tích miền (Domain Analysic)

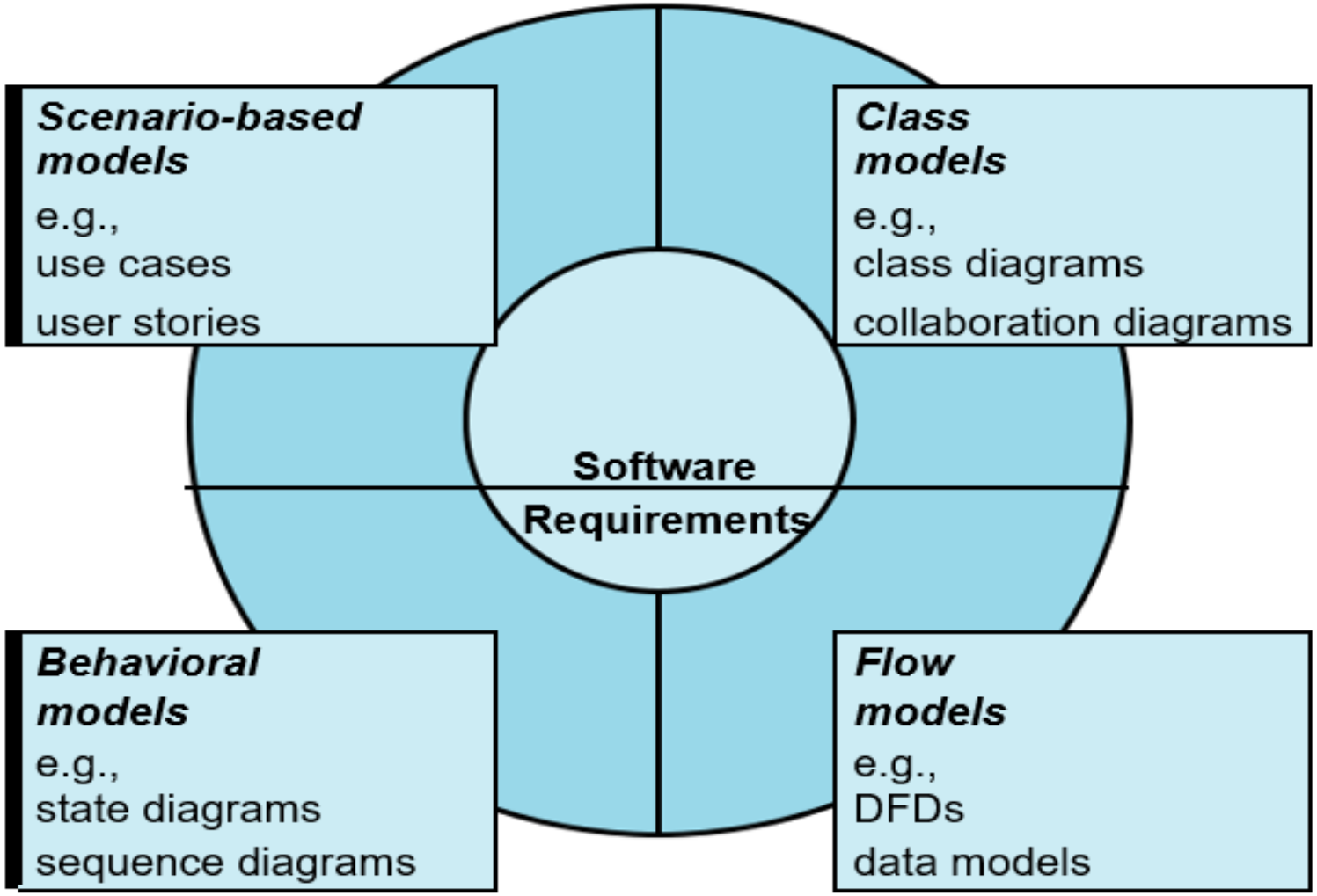
- Là xác định, phân tích và đặc tả các yêu cầu phần mềm từ một miền ứng dụng cụ thể → tái sử dụng những dự án trong cùng miền.



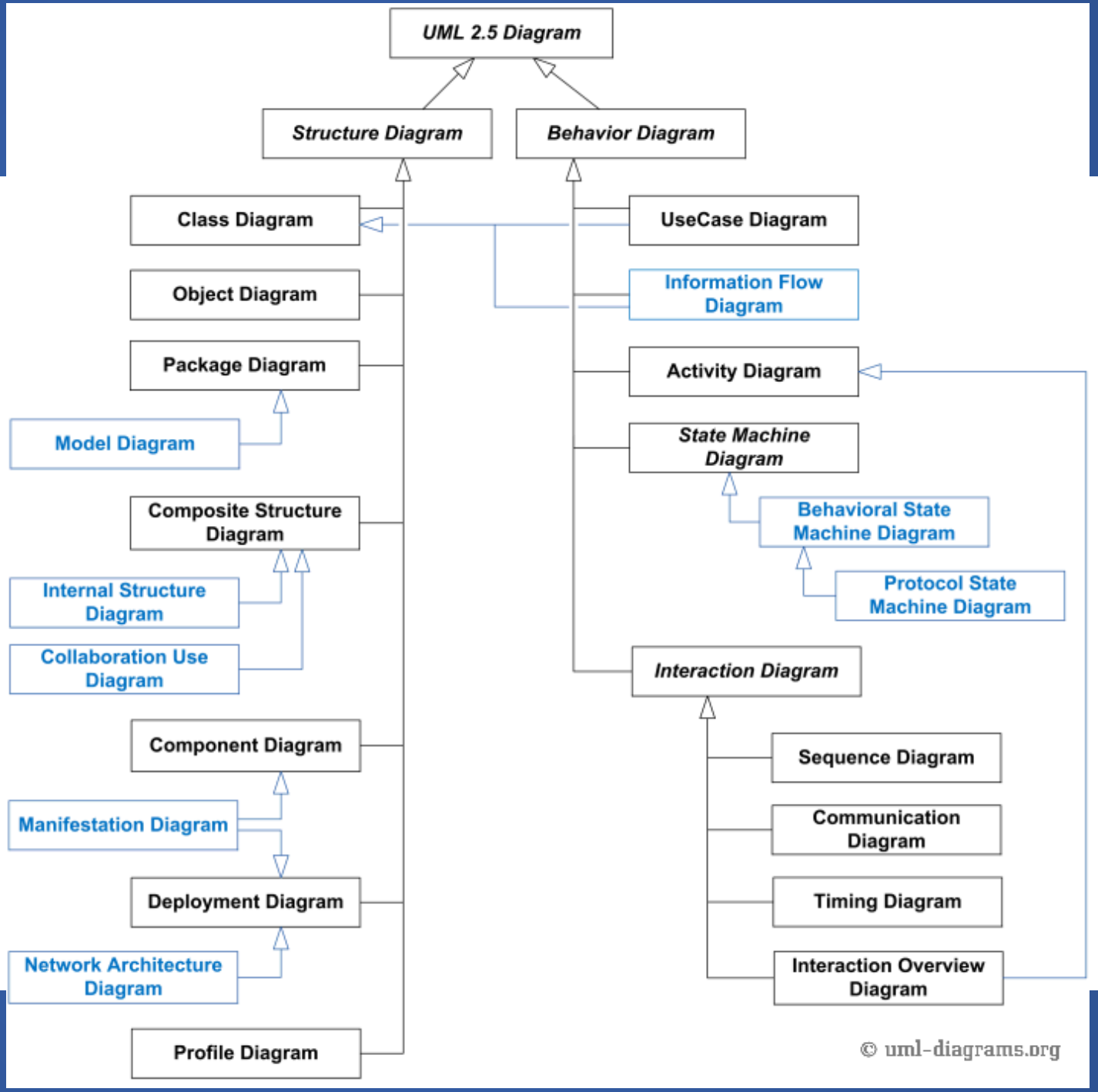
MÔ HÌNH HÓA YÊU CẦU



Phân loại mô hình yêu cầu



Phân loại mô hình yêu cầu



Mô hình dựa trên kịch bản (Scenario-base model)

- Sự hài lòng của người dùng là tiêu chí đầu tiên để xác định sự thành công của một dự án phần mềm.
- Hiểu đúng cách khách hàng muốn tương tác với phần mềm → Mô tả đúng các yêu cầu của khách hàng → Xây dựng mô hình thiết kế phù hợp.
- Mô hình hóa yêu cầu với UML bắt đầu bằng việc tạo ra các kịch bản người dùng dưới dạng:
  - Sơ đồ Use cases
  - Sơ đồ Activity (sơ đồ swimlane)

Sơ đồ USE CASE

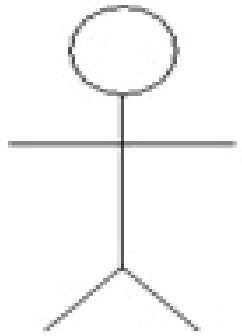
Trường hợp sử dụng (Use case) là gì?

- Là một kỹ thuật được dùng trong kỹ thuật phần mềm để xác định yêu cầu chức năng của hệ thống.
- Mục đích:
  - Mô tả sự tương tác đặc trưng giữa tác nhân (người dùng) và hệ thống để đạt được mục tiêu nào đó.
  - Mô tả ứng xử của hệ thống.
- *Mô tả các yêu cầu đối với hệ thống (những gì hệ thống phải làm chứ không phải mô tả hệ thống làm như thế nào.*
- Sử dụng ngôn ngữ của người dùng cuối, tránh dùng thuật ngữ kỹ thuật.

Sơ đồ Use Case

- Là một loại sơ đồ trong UML được dùng để mô tả các chức năng của một hệ thống phần mềm.
- Các thành phần tron sơ đồ:
  - Tác nhân – Actor: người dùng/nhóm người dùng, các hệ thống bên ngoài.
  - Các trường hợp sử dụng – use-case: các chức năng/thao tác/xử lý
  - Mối quan hệ giữa actors và use-cases.

Các ký hiệu trong sơ đồ Use case

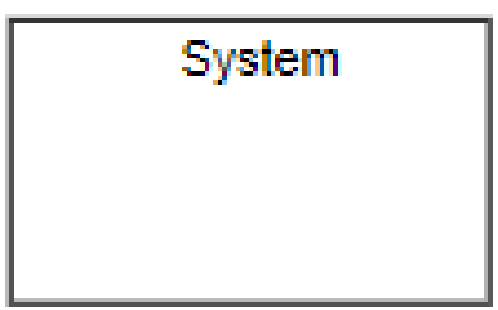
- 

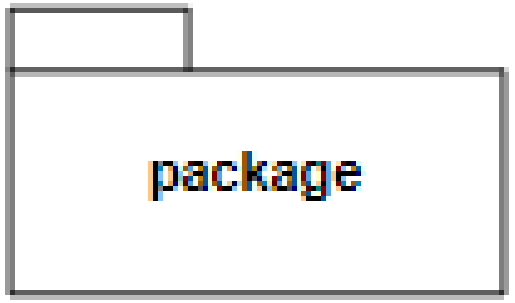
Association


Generalization

Include

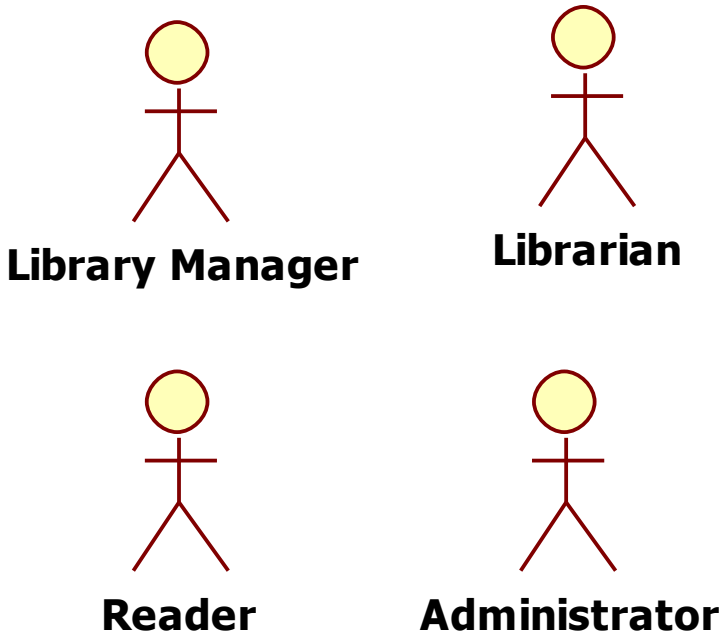
Extend





Actors

- Bao gồm:
  - Các nhóm người dùng sử dụng phần mềm
  - Các hệ thống bên ngoài có sự tương tác với phần mềm



Xác định actors

Danh từ

Ai (nhóm người) là người sử dụng phần mềm?

Những hệ thống nào khác nào có sự tương tác với phần mềm

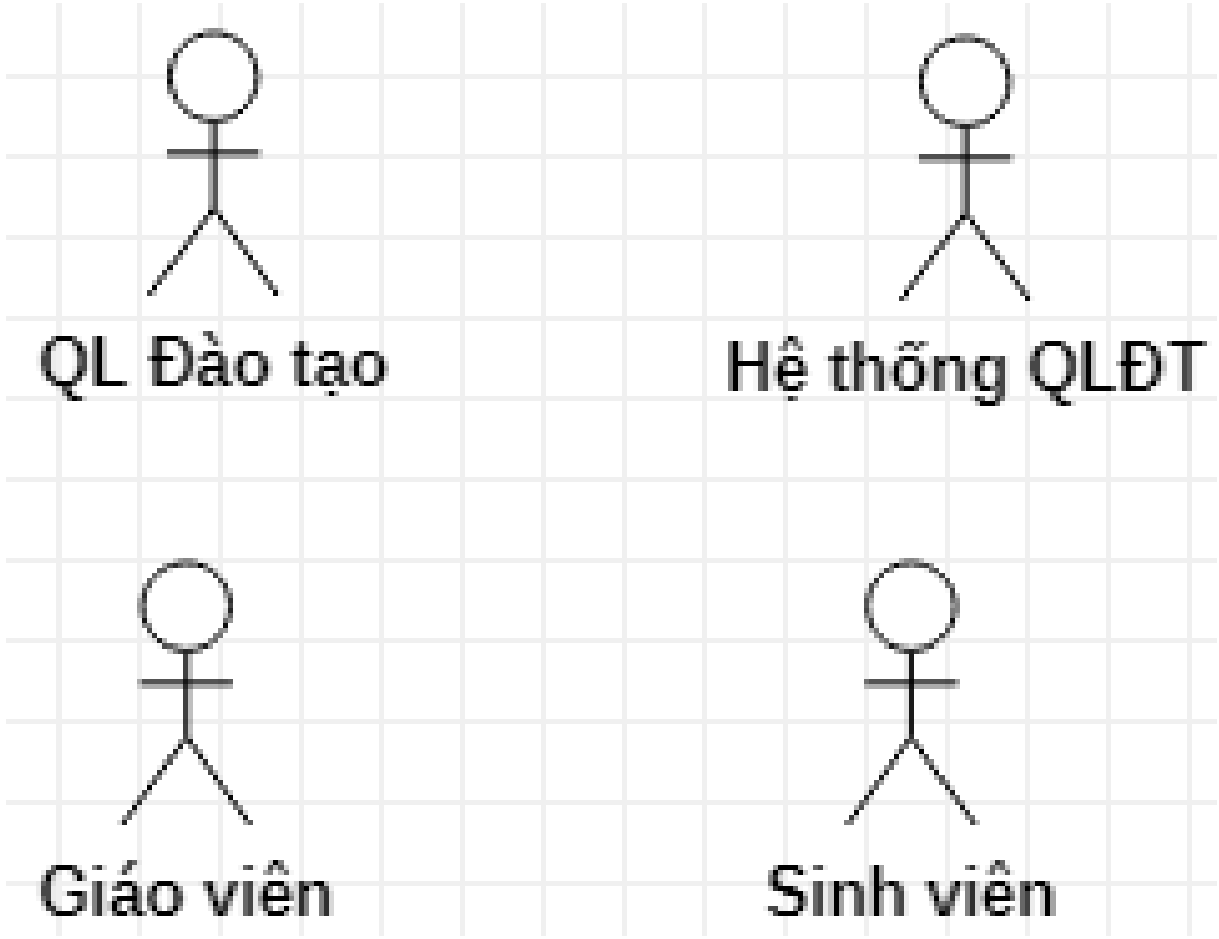
Ví dụ: Tìm actors?

- Đầu mỗi học kì, người quản lí đào tạo lên danh sách môn học và giáo viên giảng dạy để thông báo trên hệ thống đăng ký môn học. Danh sách môn học được lấy từ hệ thống quản lý đào tạo của trường. Dựa vào danh sách trên, giáo viên sẽ đăng ký lịch dạy cho các môn mình được phân công.
- Mỗi sinh viên khi đăng ký môn học, hệ thống sẽ hiển thị danh sách các môn học hiện có để sinh viên lựa chọn. Sinh viên muốn đăng ký môn nào thì chọn vào môn tương ứng, mỗi môn học có thể có nhiều lớp, do nhiều giáo viên khác nhau giảng dạy, sinh viên có thể chọn lớp và giáo viên tùy thích. Hệ thống sẽ thông báo đăng kí thành công nếu lớp đó vẫn còn chỗ trống, ngược lại sẽ yêu cầu sinh viên chọn lớp khác.

Ví dụ: Tìm actors?

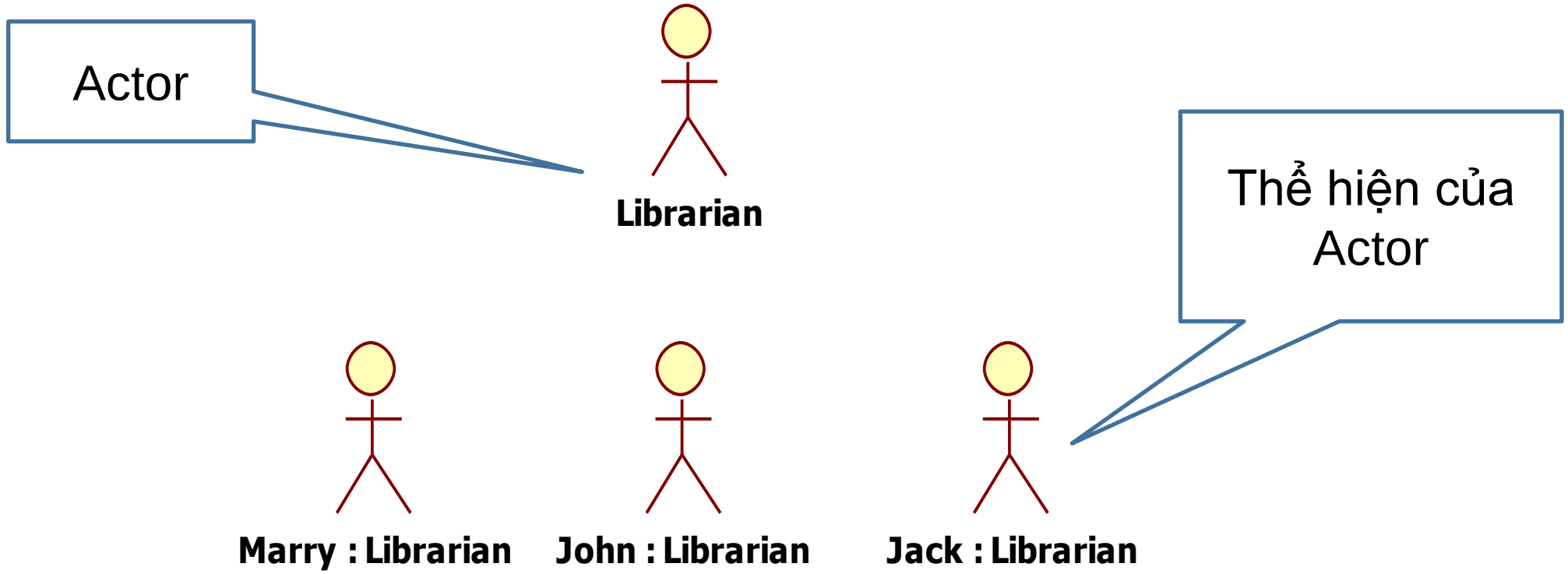
- Đầu mỗi học kì, người quản lí đào tạo lên danh sách môn học và giáo viên giảng dạy để thông báo trên hệ thống đăng ký môn học. Danh sách môn học được lấy từ hệ thống quản lý đào tạo của trường. Dựa vào danh sách trên, giáo viên sẽ đăng ký lịch dạy cho các môn mình được phân công.
- Mỗi sinh viên khi đăng ký môn học, hệ thống sẽ hiển thị danh sách các môn học hiện có để sinh viên lựa chọn. Sinh viên muốn đăng ký môn nào thì chọn vào môn tương ứng, mỗi môn học có thể có nhiều lớp, do nhiều giáo viên khác nhau giảng dạy, sinh viên có thể chọn lớp và giáo viên tùy thích. Hệ thống sẽ thông báo đăng kí thành công nếu lớp đó vẫn còn chỗ trống, ngược lại sẽ yêu cầu sinh viên chọn lớp khác.

Ví dụ: Tìm actors?



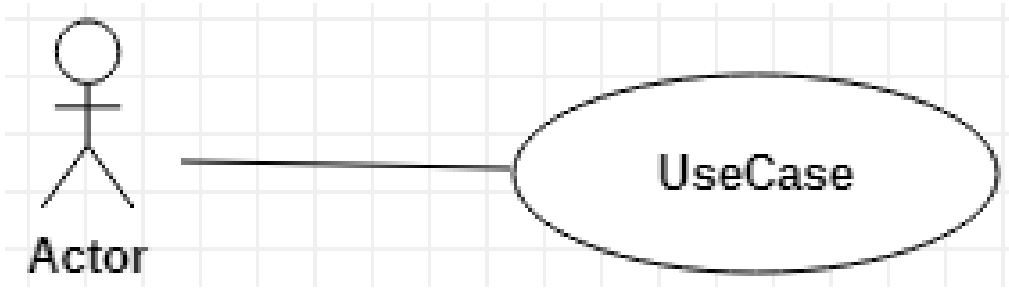
Thể hiện của actor (Actor Instance)

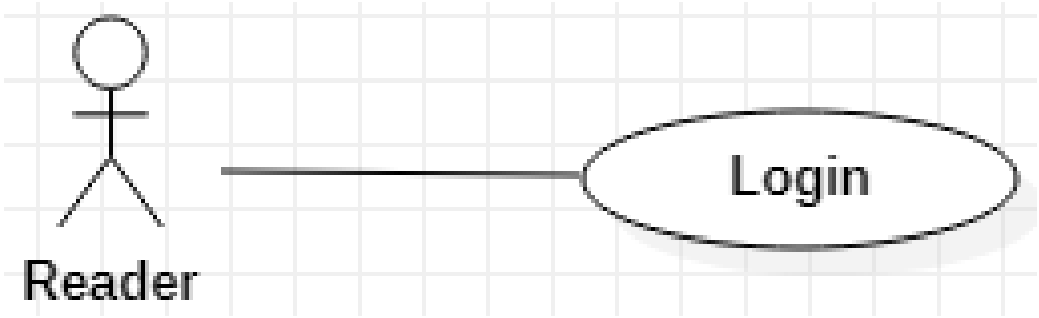
- Là một người dùng/hệ thống cụ thể



Usecase (Trường hợp sử dụng)

- Là một yêu cầu chức năng mong muốn của người dùng trên hệ thống.
- Trả lời câu hỏi:
  - Actor làm gì trên hệ thống?
  - Hệ thống có thể làm gì?





Tìm usecase của các actor?

- Tìm các **động từ** chỉ hành động, thao tác, xử lý của các actor?

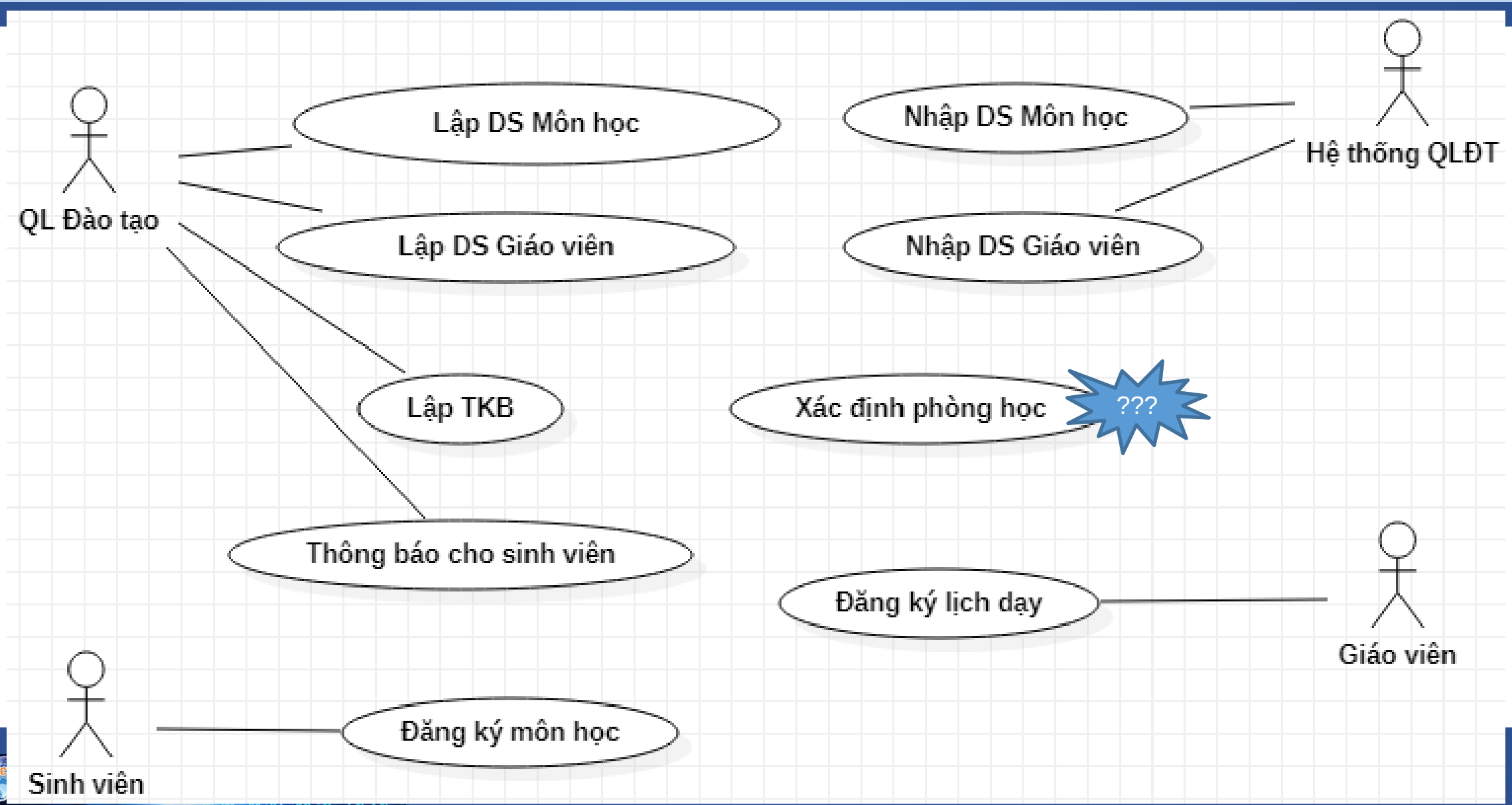
Tìm usecase của các actor?

- Đầu mỗi học kì, người quản lí đào tạo lên danh sách môn học và giáo viên giảng dạy để thông báo trên hệ thống đăng ký môn học. Danh sách môn học và giáo viên được lấy từ hệ thống quản lý đào tạo của trường. Dựa vào danh sách trên, giáo viên sẽ đăng ký lịch dạy cho các môn mình được phân công.
- Mỗi sinh viên khi đăng ký môn học, hệ thống sẽ hiển thị danh sách các môn học hiện có để sinh viên lựa chọn. Sinh viên muốn đăng ký môn nào thì chọn vào môn tương ứng, mỗi môn học có thể có nhiều lớp, do nhiều giáo viên khác nhau giảng dạy, sinh viên có thể chọn lớp và giáo viên tùy thích. Hệ thống sẽ thông báo đăng kí thành công nếu lớp đó vẫn còn chỗ trống, ngược lại sẽ yêu cầu sinh viên chọn lớp khác.
- Sau khi kết thúc thời gian đăng kí môn học, dựa vào số sinh viên đăng kí, người quản lí đào tạo lập thời khóa biểu, sắp xếp địa điểm học và thông báo cho sinh viên.
- Để thực hiện được các chức năng trên hệ thống, tất cả người dùng đều phải đăng nhập thành công với một tài khoản đã được cấp trước đó.

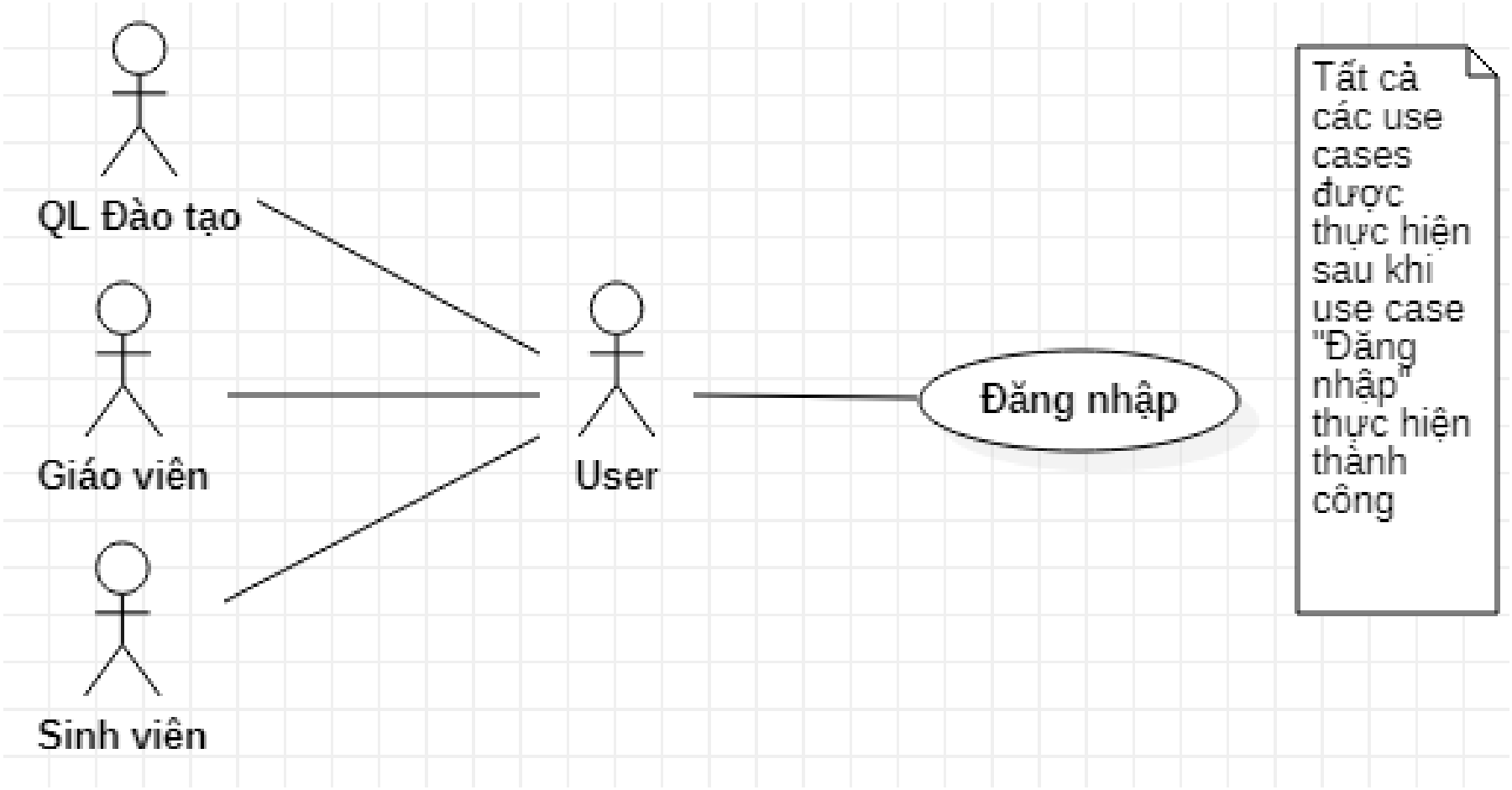
Tìm usecase của các actor?

- Đầu mỗi học kì, người quản lí đào tạo lên danh sách môn học và giáo viên giảng dạy để thông báo trên hệ thống đăng ký môn học. Danh sách môn học và giáo viên được lấy từ hệ thống quản lý đào tạo của trường. Dựa vào danh sách trên, giáo viên sẽ đăng ký lịch dạy cho các môn mình được phân công.
- Mỗi sinh viên khi đăng ký môn học, hệ thống sẽ hiển thị danh sách các môn học hiện có để sinh viên lựa chọn. Sinh viên muốn đăng ký môn nào thì chọn vào môn tương ứng, mỗi môn học có thể có nhiều lớp, do nhiều giáo viên khác nhau giảng dạy, sinh viên có thể chọn lớp và giáo viên tùy thích. Hệ thống sẽ thông báo đăng kí thành công nếu lớp đó vẫn còn chỗ trống, ngược lại sẽ yêu cầu sinh viên chọn lớp khác.
- Sau khi kết thúc thời gian đăng kí môn học, dựa vào số sinh viên đăng kí, người quản lí đào tạo lập thời khóa biểu, sắp xếp địa điểm học và thông báo cho sinh viên.
- Để thực hiện được các chức năng trên hệ thống, tất cả người dùng đều phải đăng nhập thành công với một tài khoản đã được cấp trước đó.

Tìm usecase của các actor?



Tìm usecase của các actor?

- 

Mô tả Usecase

Mô tả tất cả các chi tiết liên quan đến Usecase mà chưa được thể hiện trong sơ đồ Usecase

Bao gồm các mô tả sau:

- Mã, tên
- Mục tiêu
- Mô tả
- Actor thực hiện
- Actor thực hiện
- Ràng buộc
- Điều kiện trước khi thực hiện (tiên quyết)
- Điều kiện sau khi thực hiện
- Kịch bản:
  - Các bước thực hiện Usecase
  - Các trường hợp có thể xảy ra khi thực hiện Usecase

Mô tả Usecase: Lập DS Môn học

- Mã: UC11
- Tên: Lập DS Môn học
- Mô tả: Lập danh sách các môn học sẽ dạy trong học kỳ
- Mục tiêu: Lập danh sách các môn học sẽ được giảng dạy trong học kỳ để các giáo viên có thể đăng ký giảng dạy

Mô tả Usecase: Lập DS Môn học

- Actor: Nhân viên QL Đào tạo.
- Ràng buộc:
  - Hệ thống QLĐT phải tồn tại dữ liệu về danh sách các môn học theo khung chương trình đào tạo
  - Phần mềm phải lấy được danh sách các môn học theo khung chương trình đào tạo từ hệ thống QLĐT.
- Điều kiện trước khi thực hiện:
  - Nhân viên QL Đào tạo phải đăng nhập thành công vào hệ thống với quyền "QL Đào tạo"
- Điều kiện sau khi thực hiện: danh sách môn học được lưu trữ trong hệ thống

Mô tả Usecase: Lập DS Môn học

Kịch bản:

- Nhân viên QLĐT chọn năm học, học kỳ và chọn chức năng "Nhập DS môn học"
- Hệ thống kết nối với hệ thống QLĐT
  - Nếu hệ thống kết nối thành công với hệ thống QLĐT: hệ thống sẽ yêu cầu lấy danh sách các môn học theo học kỳ của tất cả các lớp mà nhân viên QLĐT đã chọn
    - ✓ Nếu có danh sách các môn học:
      - Nhân viên QLĐT chọn các môn học sẽ mở và chọn chức năng "Mở học phần"

Mô tả Usecase: Lập DS Môn học

Kịch bản:

- Hệ thống sẽ lưu danh sách các học phần sẽ mở trong học kỳ và năm học đã được chọn trước đó.
  - Nếu danh sách học phần được lưu thành công: thông báo nhập danh sách môn học thành công
  - Ngược lại: thông báo lỗi đã xảy

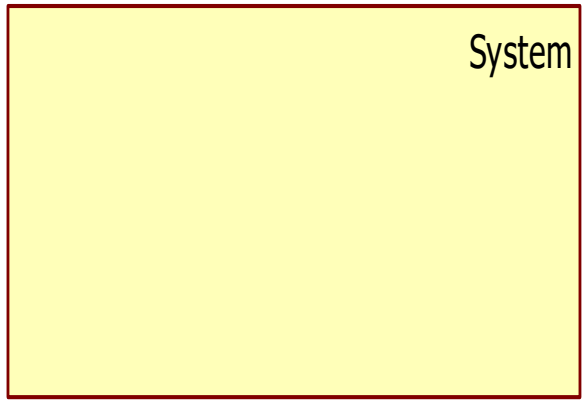
✓ Ngược lại nêu không có danh sách các môn học: hệ thống thông báo lỗi không lấy được danh sách môn học.

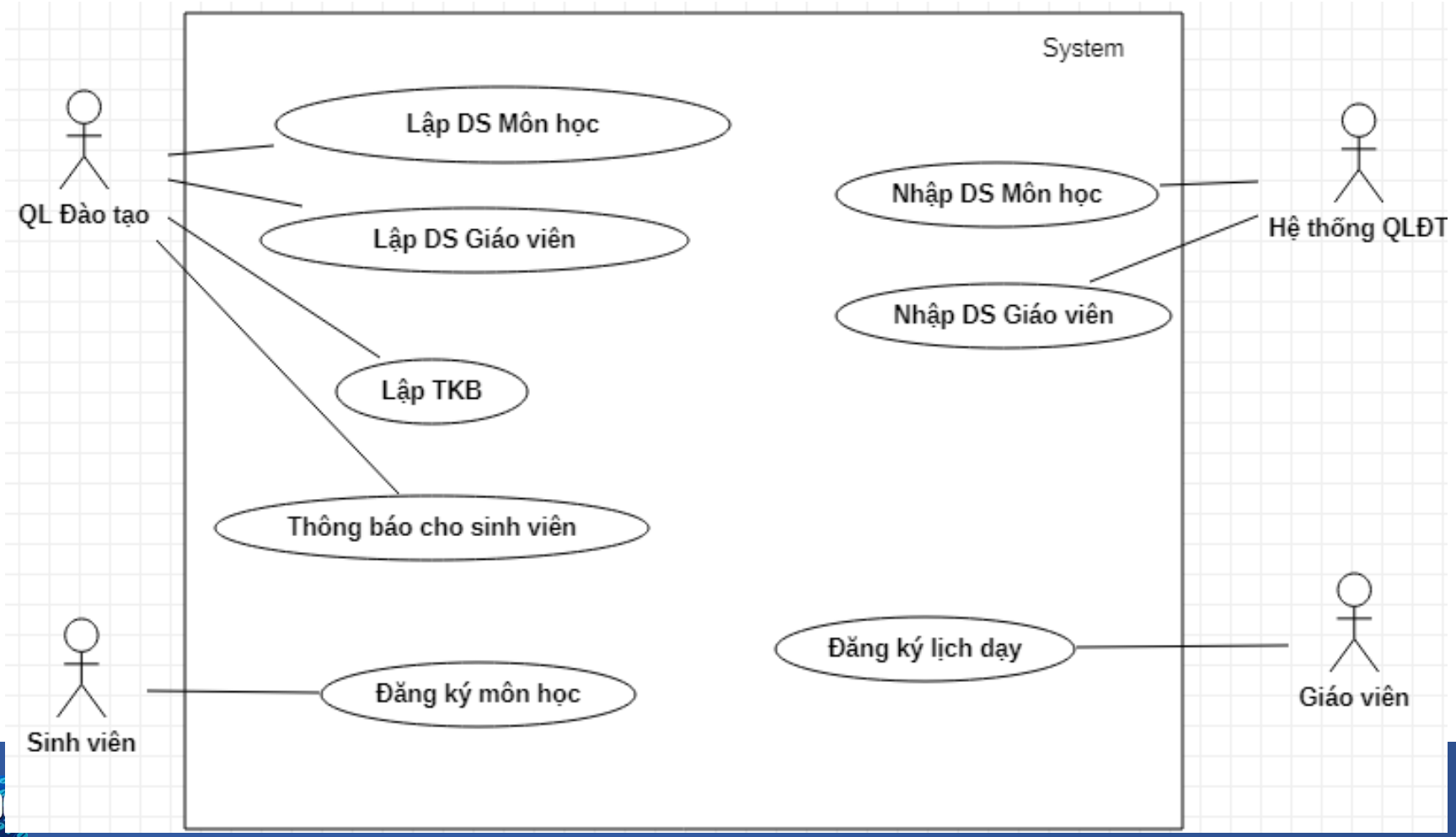
➢ Nếu hệ thống kết nối không thành công với hệ thống QLĐT: thông báo lỗi không kết nối được với hệ thống QLĐT sẽ được hiển thị.

Mô tả Usecase: Đăng ký môn học

System boundary (Biên của hệ thống)

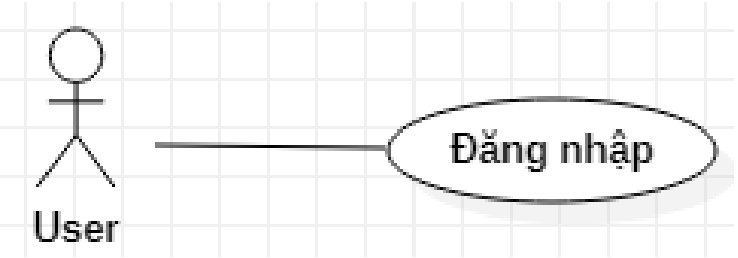
- Các usecase có thể được đặt bên trong một hình chữ nhật đại diện cho ranh giới của hệ thống

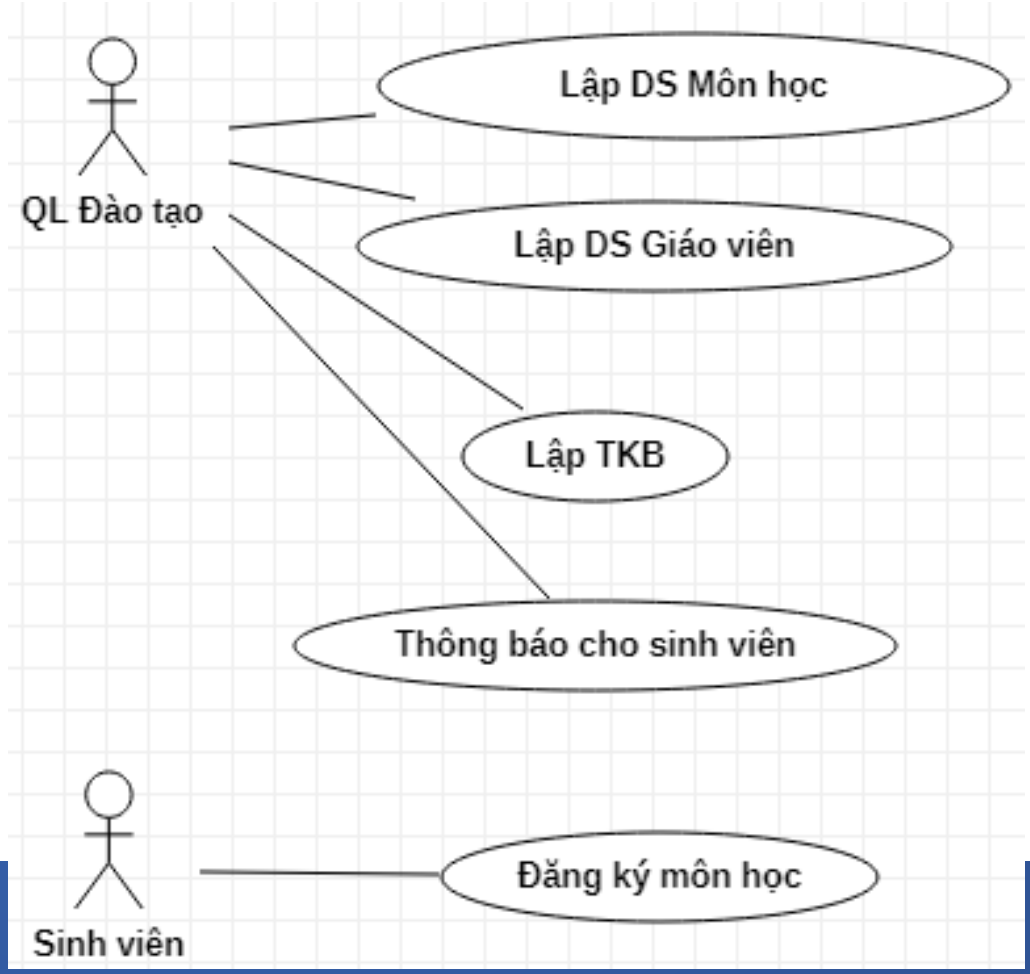




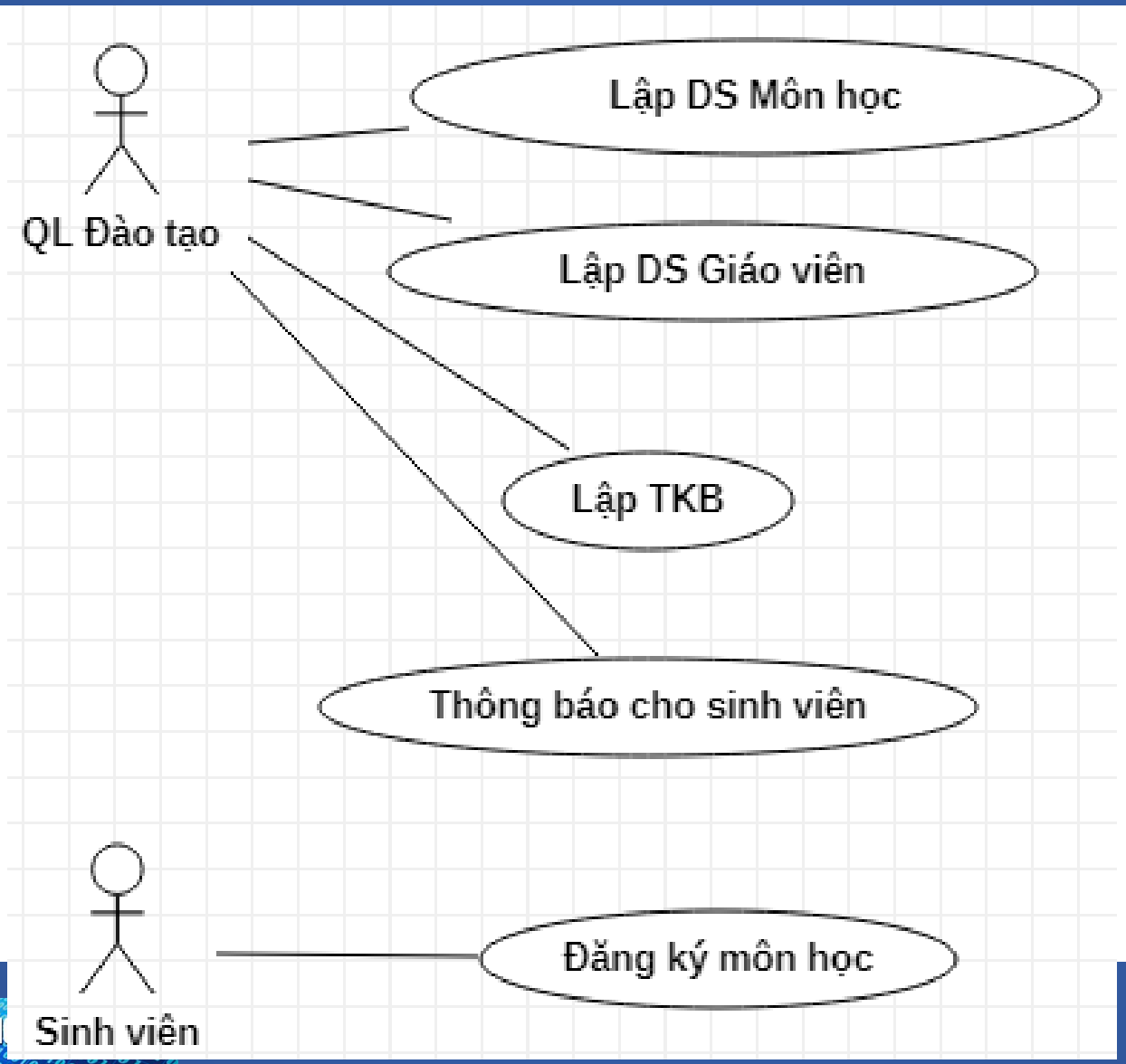
Quan hệ kết hợp (Association/Communication)

- Mô tả mối quan hệ giữa Actor và Use Case: **Actor thực hiện Use Case**

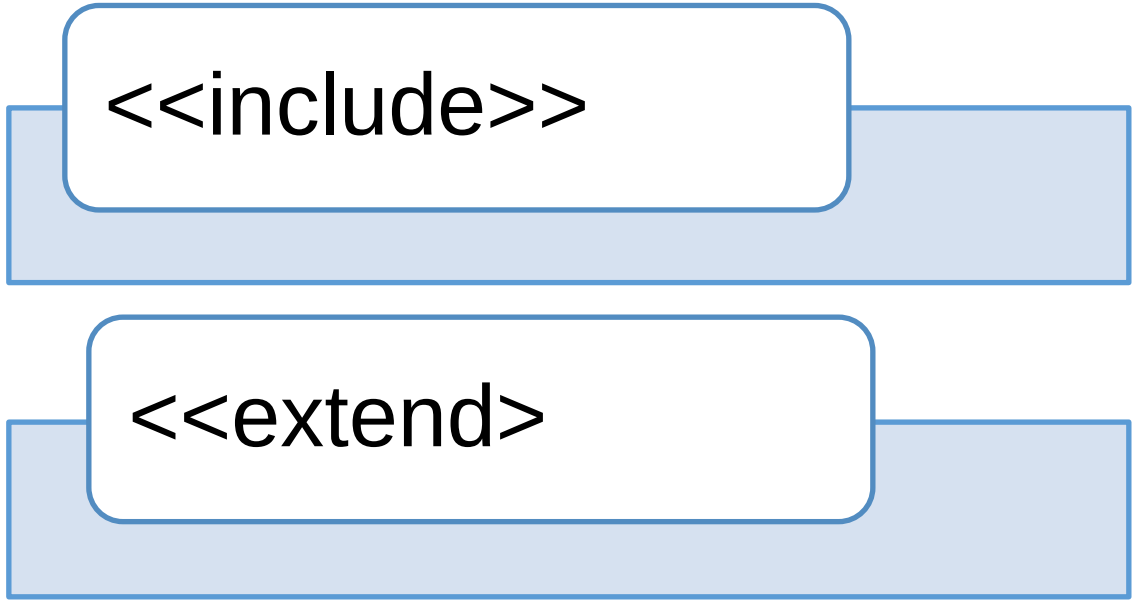




Quan hệ kết hợp (Association/Communication)

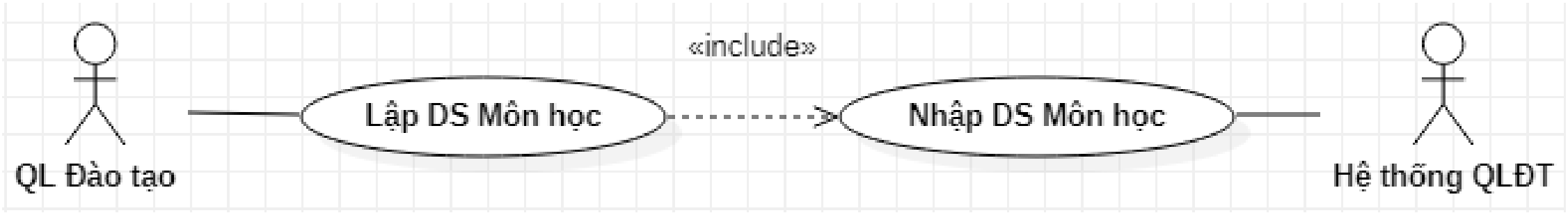


Quan hệ phụ thuộc (Dependencies)



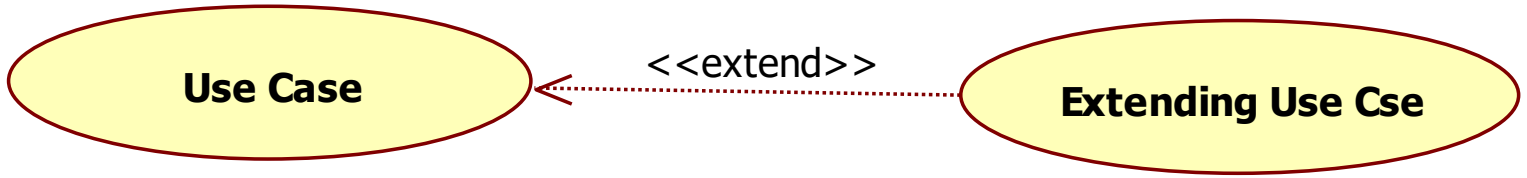
Quan hệ bao gồm - “include”

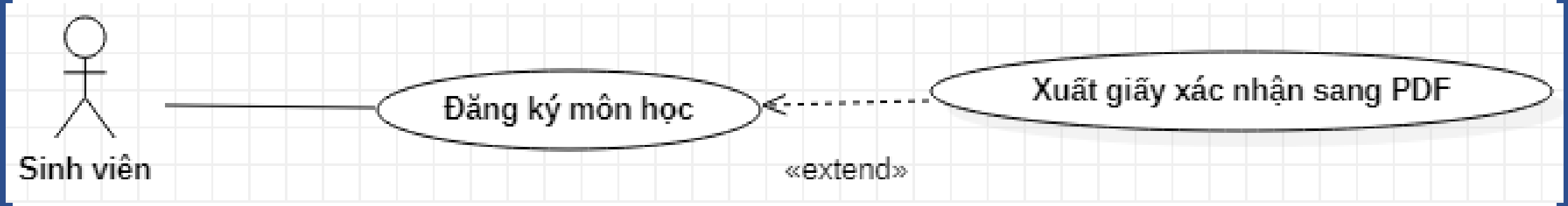
- Quan hệ bao gồm giữa các usecase với nhau
- Usecase 2 được thực hiện trong khi thực hiện usecase 1 (Usecase 2 bắt buộc là một trong các bước thực hiện trong usecase 1)



Quan hệ mở rộng- "extend"

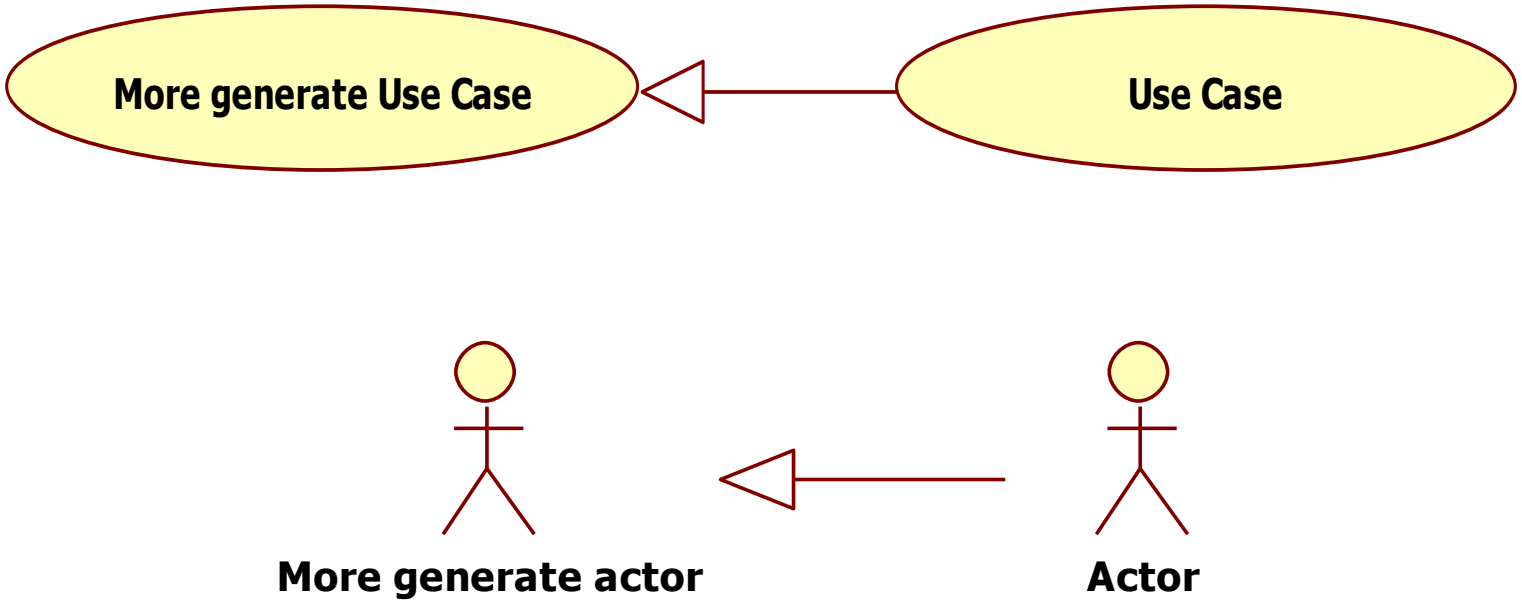
- Quan hệ mở rộng giữa các usecase với nhau
- Usecase 2 được thực hiện sau khi (đôi lúc trong khi) thực hiện usecase 1.
  - Có những trường hợp thực hiện usecase 1 mà không thực hiện usecase 2.





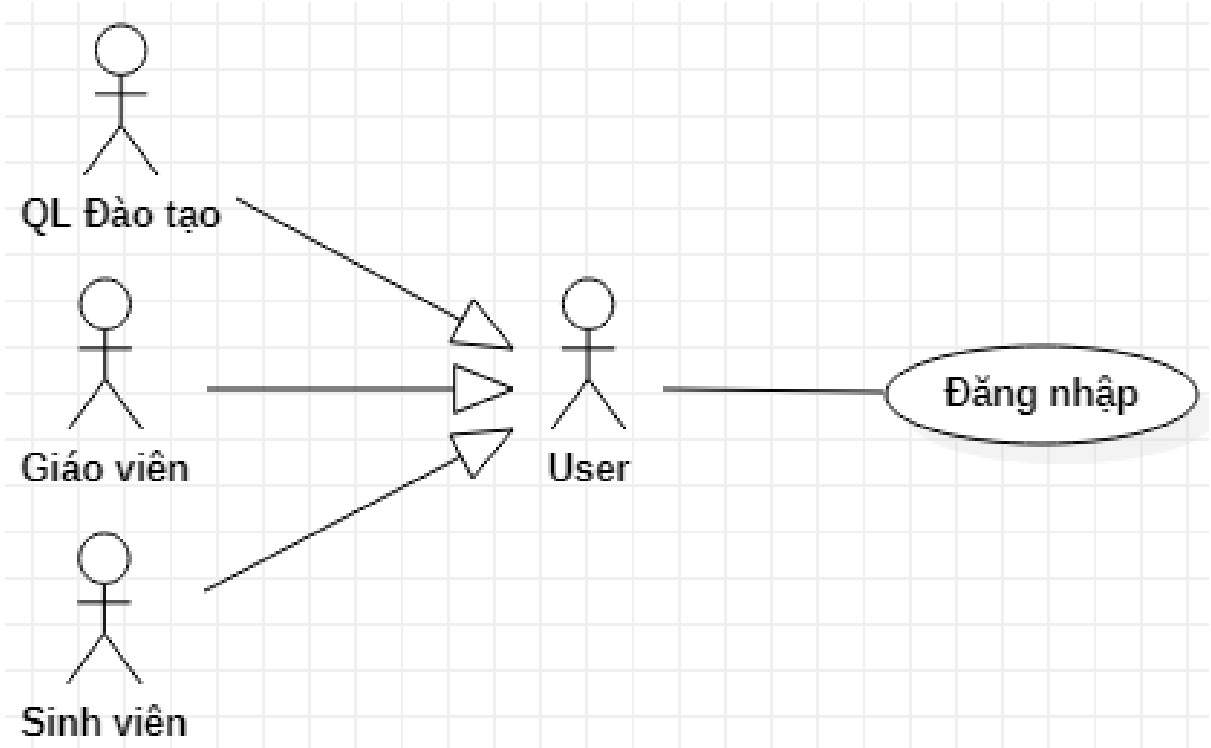
Quan hệ tổng quát hóa - Generalization

- Mô tả mối quan hệ tổng quát/phân loại/kế thừa giữa các usecase với nhau hoặc giữa các actor với nhau.



Quan hệ tổng quát hóa - Generalization

- Mô tả mối quan hệ tổng quát/phân loại/kế thừa giữa các usecase với nhau hoặc giữa các actor với nhau.



Quan hệ tổng quát hóa - Generalization

- Mô tả mối quan hệ tổng quát/phân loại/kế thừa giữa các usecase với nhau hoặc giữa các actor với nhau.

BT: Hệ thống Quản lý thư viện

- Tại thư viện của thành phố ABC, việc quản lý vẫn được thực hiện thủ công. Trong thư viện, thủ thư chịu trách nhiệm quản lý sách và các tài nguyên, quản lý đọc giả và việc mượn trả sách của đọc giả.
- Sách được quản lý theo các danh mục sách. Mỗi danh mục là một loại sách, như sách văn học, hóa học, ngoại ngữ, tin học ….Mỗi danh mục sách được đánh một mã để dễ quản lý. Mỗi quyển sách được xếp vào một loại danh mục sách. Mỗi quyển sách được đánh một mã sách dùng để phân biệt với các quyển sách khác (giả sử nếu một tác phẩm có nhiều bản giống nhau hoặc có nhiều tập thì cũng xem là có mã sách khác nhau). Ngoài ra, mỗi quyển sách còn lưu trữ các thông tin khác như : tên sách, năm xuất bản. Mỗi quyển sách cũng cần được biết do tác giả nào viết. Thông tin tác giả bao gồm mã tác giả, tên tác giả, năm sinh, quốc tịch. Một tác giả có thể viết nhiều sách. Một quyển sách có thể do nhiều tác giả viết.
- Thủ thư chịu trách nhiệm quản lý danh mục sách và các quyển sách. Khi có một quyển sách được nhập về, thủ thư sẽ xem xét quyển sách đó thuộc danh mục nào. Nếu quyển sách thuộc một danh mục mới, thủ thư có thể tạo mới danh mục đó. Việc nhập thông tin quyển sách phải được thực hiện trước khi quyển sách đó được đưa vào sử dụng.
- Đọc giả muốn thực hiện việc mượn sách phải là đọc giả của thư viện. Khi đăng ký làm đọc giả của thư viện, mỗi đọc giả được thư viện cấp cho một mã đọc giả. Ngoài ra, thư viện còn lưu trữ họ tên, ngày sinh, địa chỉ, giới tính, nghề nghiệp, ảnh cá nhân …. Khi không sử dụng các dịch vụ của thư viện nữa, đọc giả có thể yêu cầu hủy tư cách đọc giả của mình.

BT: Hệ thống Quản lý thư viện

- Cứ mỗi lượt mượn sách, đọc giả phải đăng ký các quyển sách cần mượn vào một phiếu mượn, mỗi phiếu mượn có một số phiếu mượn khác nhau, mỗi phiếu mượn xác định các thông tin như: ngày mượn, đọc giả mượn và các quyển sách được mượn. Các các quyển sách trong cùng một phiếu mượn không nhất thiết phải trả trong một lần. Mỗi quyển sách có thể thuộc nhiều phiếu mượn khác nhau (tất nhiên là tại các thời điểm khác nhau). Mỗi phiếu mượn chỉ được mượn tối đa là 5 quyển sách và khoảng thời gian mượn tối đa là 1 tuần. Sau khoảng thời gian quy định, nếu đọc giả có nhu cầu mượn tiếp tục sẽ phải làm phiếu gia hạn cho quyển sách mình mượn. Thời gian gia hạn cũng không quá 1 tuần. Một quyển sách được mượn chỉ được gia hạn tối đa 3 lần. Đọc giả không được mượn một quyển sách ngay sau khi trả quyển sách đó. Thời gian có thể mượn lại quyển sách đó tối thiểu phải là 24 giờ.
- Khi nhận được phiếu mượn, thủ thư sẽ quyết định xem phiếu mượn có được chấp nhận hay không. Việc chấp nhận phụ thuộc vào việc quyển sách đó có thể cho mượn hay không. Khi quyển sách đọc giả cần mượn đã cho người khác mượn rồi hoặc nó đang bị hư hỏng gì đó … thì không thể cho đọc giả đó mượn.
- Khi trả sách, thủ thư phải ghi nhận lại tình trạng của sách là bình thường hay có hư hỏng hoặc mất mát gì hay không. Nếu quyển sách bị hư hoặc mất mát, tùy theo trường hợp mà thủ thư sẽ đưa ra các hình thức xử lý khác nhau. Việc trả sách quá thời hạn quy định cũng phải chịu phạt. Mức phạt được áp dụng tùy theo thời gian quá hạn.
- Cuối mỗi tháng, thủ thư phải lập và gởi báo cáo về tình trạng cho mượn sách lên cấp quản lý của thư viện. Quản lý của thư viện cũng có thể yêu cầu thủ thư lập các bảng thống kê tình hình quản lý sách ở thư viện như: thống kê các quyển sách theo danh mục, thống kê các quyển sách được mượn nhiều nhất hoặc ít nhất, thống kê các quyển sách hư hỏng không sử dụng được …

Activity diagram

Sơ đồ Activity (hoạt động)

- Mô tả các hoạt động, các luồng xử lý xảy ra trong hệ thống.
- Là chi tiết hóa mỗi một usecase
  - Usecase: “Hệ thống làm cái gì - what?”
  - Activity: “Hệ thống làm như thế nào - how?”
- Được sử dụng để mô tả các qui trình nghiệp vụ trong hệ thống, các luồng của một chức năng hoặc các hoạt động của một đối tượng.

Swimlance

- Mô tả đối tượng tham gia vào hoạt động

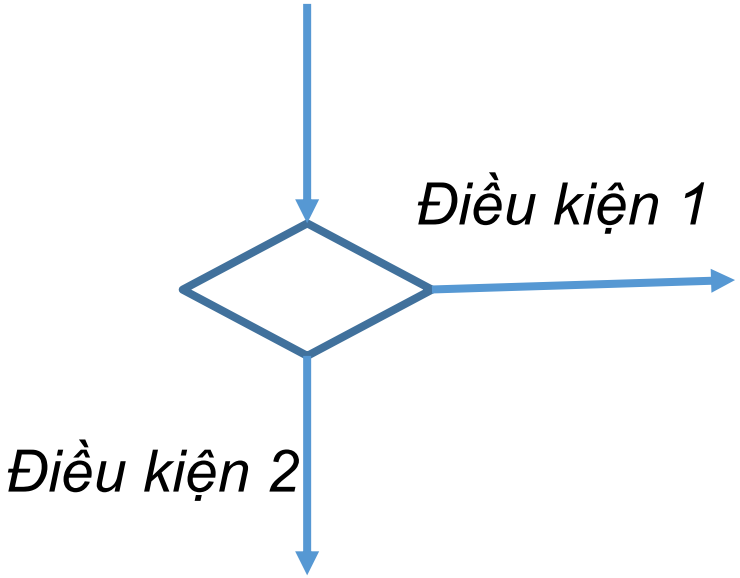
Action

- Mô tả hoạt động trong hệ thống

Bắt đầu Xử lý Kết thúc

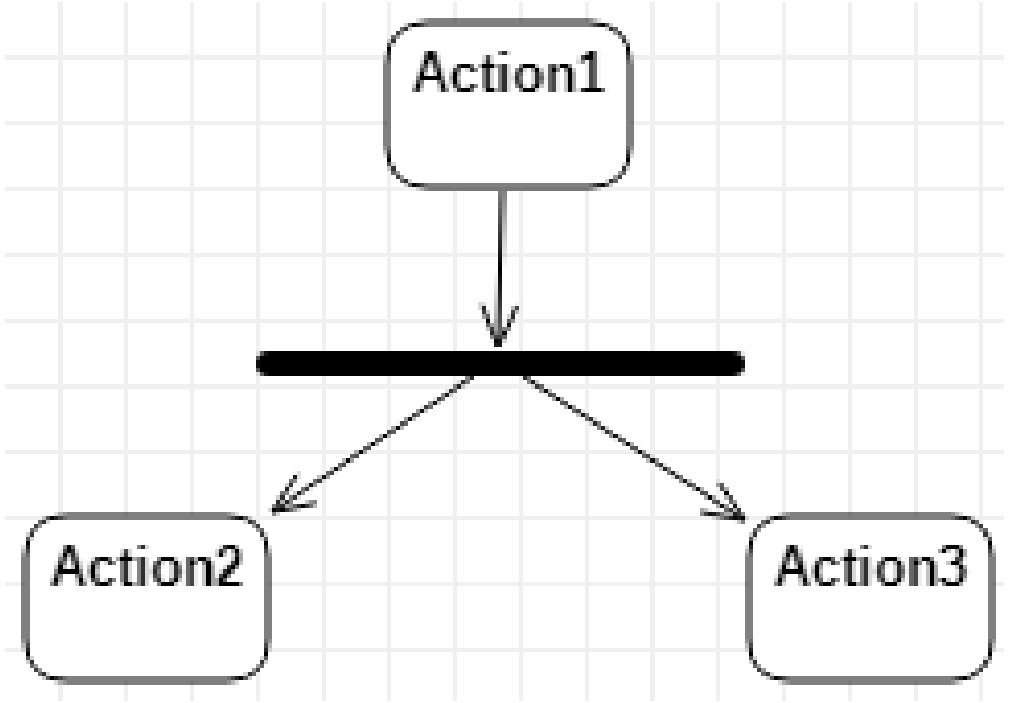
Điều kiện

- Xác định điều kiện thực hiện các hoạt động



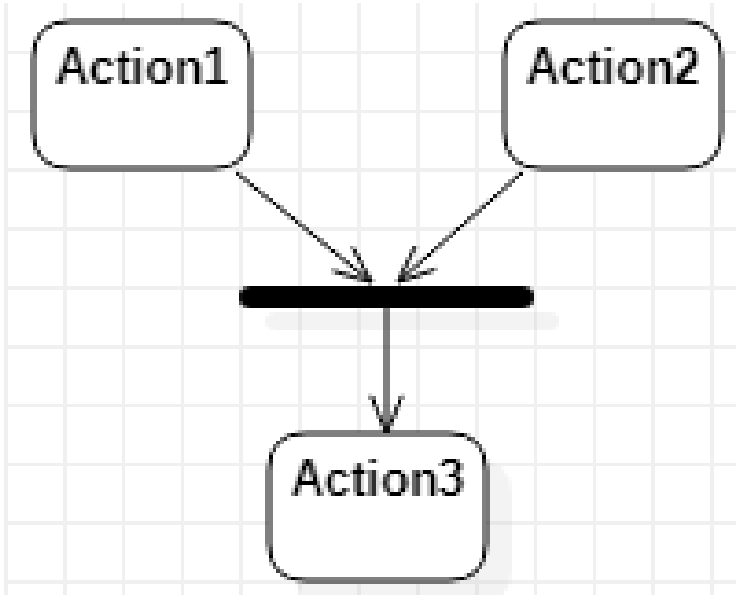
Phân nhánh - Fork

- Thực hiện xong một hoạt động rồi rẽ nhánh thực hiện nhiều hoạt động tiếp theo một cách song song

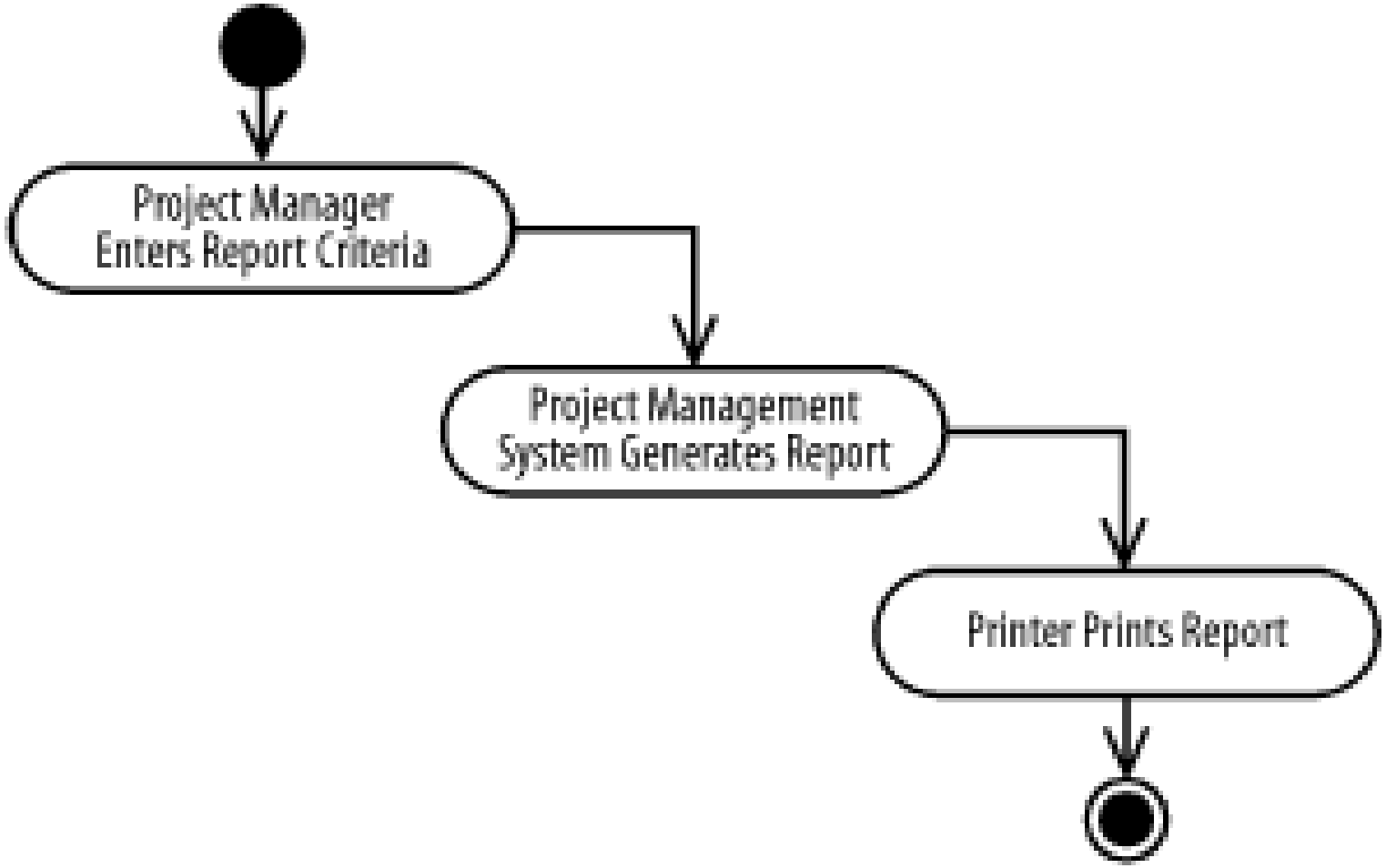


Nhập nhánh - Join

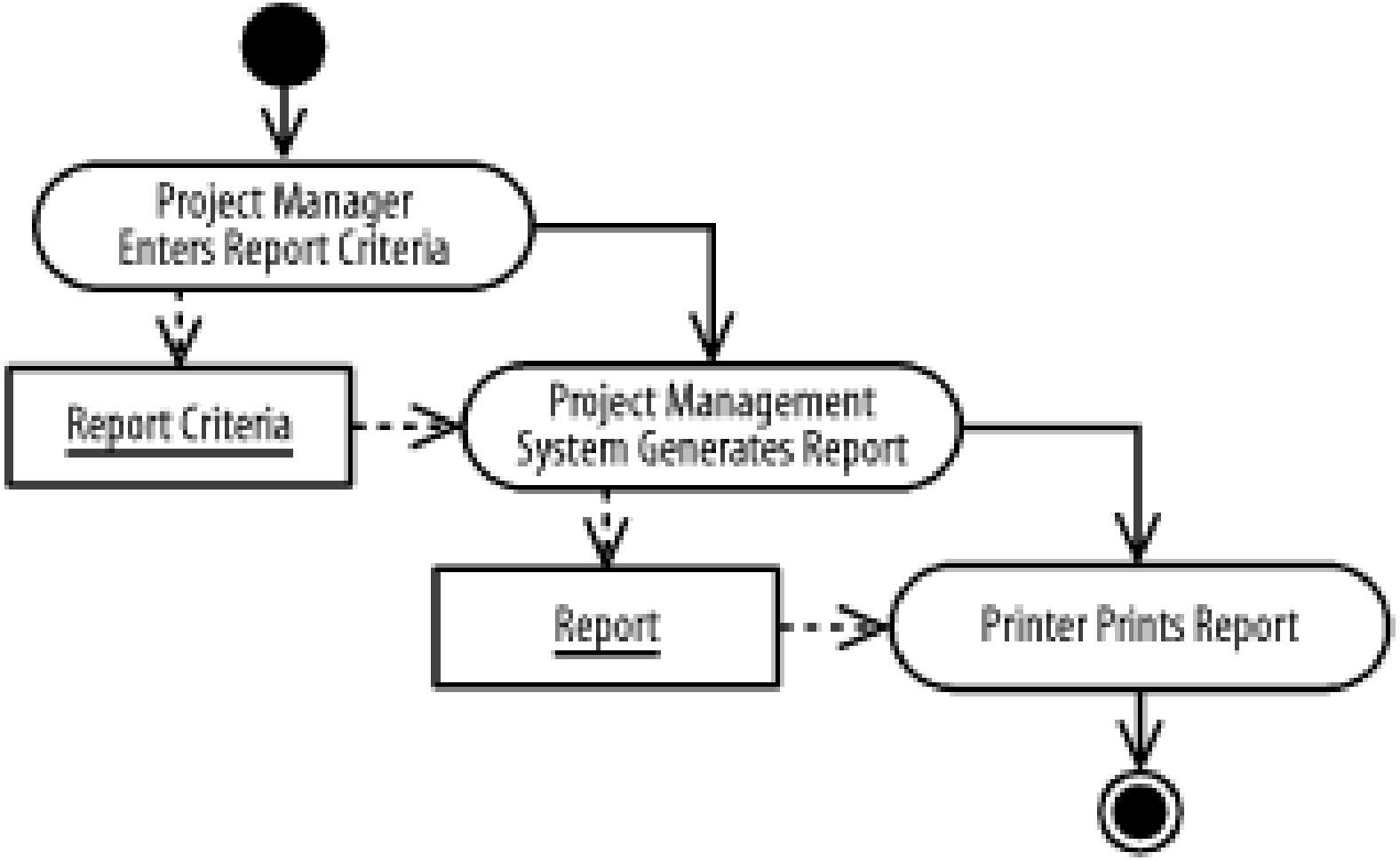
- Thực hiện xong hết tất cả các hoạt động trước đó, sau đó mới thực hiện hoạt động tiếp theo



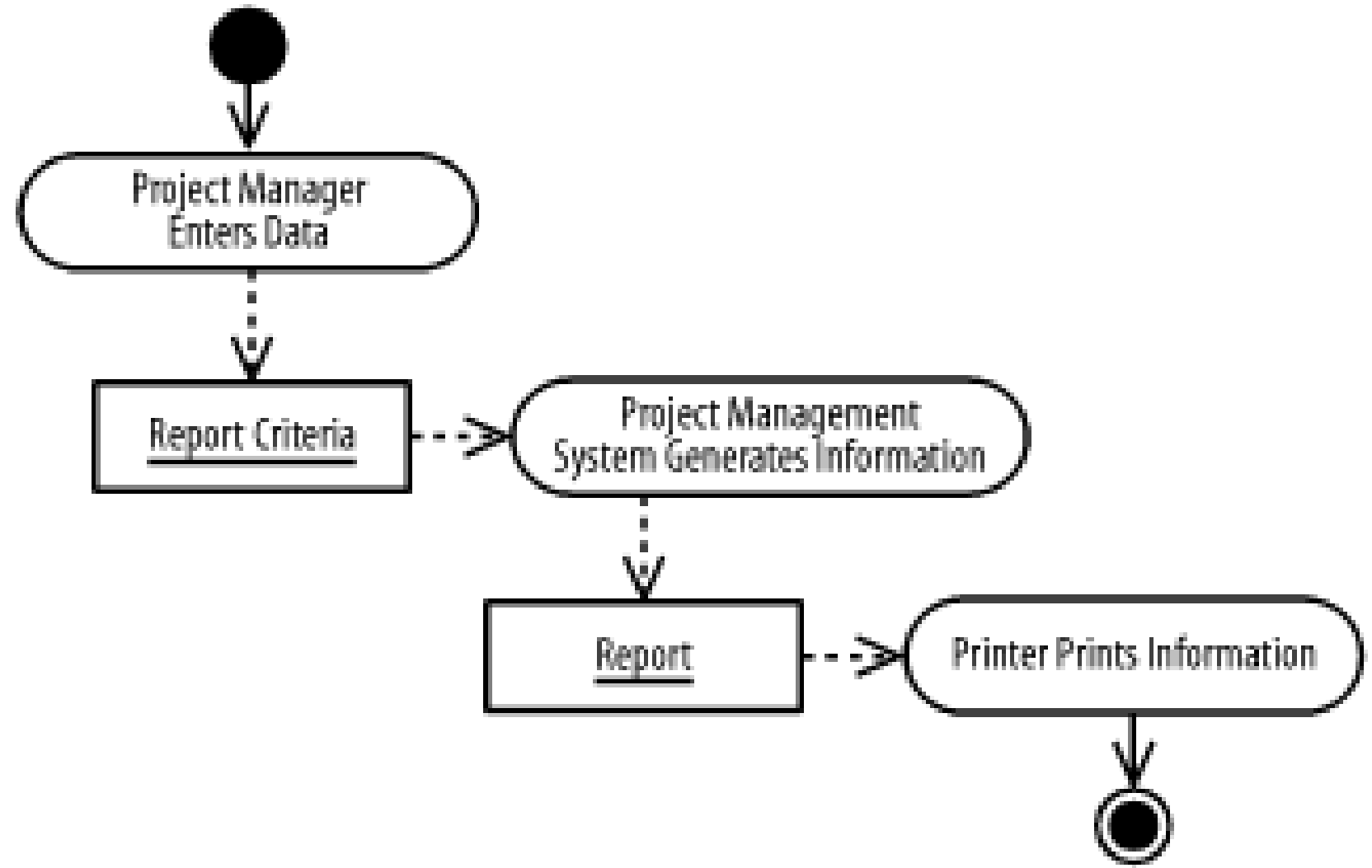
Control-Flow

- 

Object-Flow


Project Manager
Enters Report Criteria
Report Criteria
Project Management
System Generates Report
Report
Printer Prints Report

Bỏ qua control-flow



Bài tập

Thiết kế sơ đồ Activity cho các usecase:

- Đăng ký lịch dạy
- Lập thời khóa biểu
- Đăng ký môn học

Câu hỏi

- Phân tích các tiếp cận thiết kế: BDUF (Big Design Up Front), NDUF (No Design Up Front) và EDUF (Enough Design Up Front)?
- Mục tiêu chính của các mô hình yêu cầu là gì?
- Mục tiêu của mô hình yêu cầu dựa trên kịch bản?
- Kể tên các mô hình yêu cầu dựa trên kịch bản?
- Mô tả ý nghĩa của sơ đồ Use Case? Các ký hiệu trong sơ đồ Use Case?
- Swimlane diagram là gì?