
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# Systems analysis activities
## Structural analysis

Viet-Trung Tran
trungtv.github.io
School of Information and Communication Technology

# Phân tích cấu trúc

- Mục đích của phân tích cấu trúc
- Đối tượng và lớp
- Phát hiện các lớp lĩnh vực
- Phát hiện các lớp tham gia các ca sử dụng
- Bài tập tổng hợp

# Mục đích của phân tích cấu trúc

- Sơ bộ phát hiện các lớp chính tạo nên hệ thống
  - Xây dựng thuật ngữ chung cho người sử dụng và người phân tích hệ thống
  - Biểu diễn sự vật, ý tưởng và khái niệm quan trọng trong hệ thống

# Đối tượng và lớp

- Định nghĩa và biểu diễn đối tượng và lớp
- Các thuộc tính
- Các thao tác
- Các mối liên quan
  - Phụ thuộc
  - Khái quát hóa
  - Liên kết
- Biểu đồ lớp và biểu đồ đối tượng

# Định nghĩa và biểu diễn đối tượng và lớp (1)

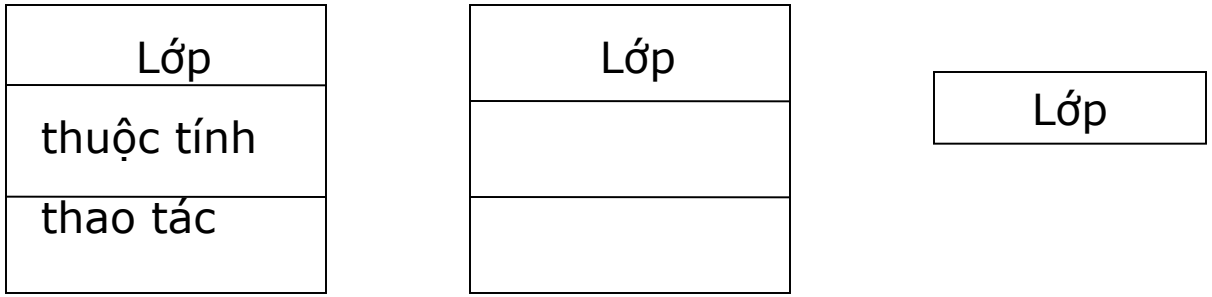
- **Đối tượng (tin học)** là một biểu diễn trừu tượng của một thực thể (vật lý hay khái niệm) có định danh và ranh giới rõ ràng trong thế giới thực, bao gồm cả trạng thái và hành vi của thực thể đó
  - **Trạng thái** của đối tượng thể hiện bởi một tập hợp các thuộc tính. Ở mỗi thời điểm, mỗi thuộc tính của đối tượng có một giá trị nhất định.
  - **Hành vi** của đối tượng thể hiện bằng một tập hợp các thao tác, đó là các dịch vụ mà nó có thể thực hiện khi được một đối tượng khác yêu cầu.
  - **Định danh** của đối tượng là cái để phân biệt nó với đối tượng khác

# Định nghĩa và biểu diễn đối tượng và lớp (2)

- **Lớp** là một mô tả của một tập hợp các đối tượng cùng có chung các thuộc tính, các thao tác, các mối liên quan, các ràng buộc và ngữ nghĩa
- Lớp là một kiểu, và mỗi đối tượng thuộc lớp là một cá thể (instance)

# Định nghĩa và biểu diễn đối tượng và lớp (3)

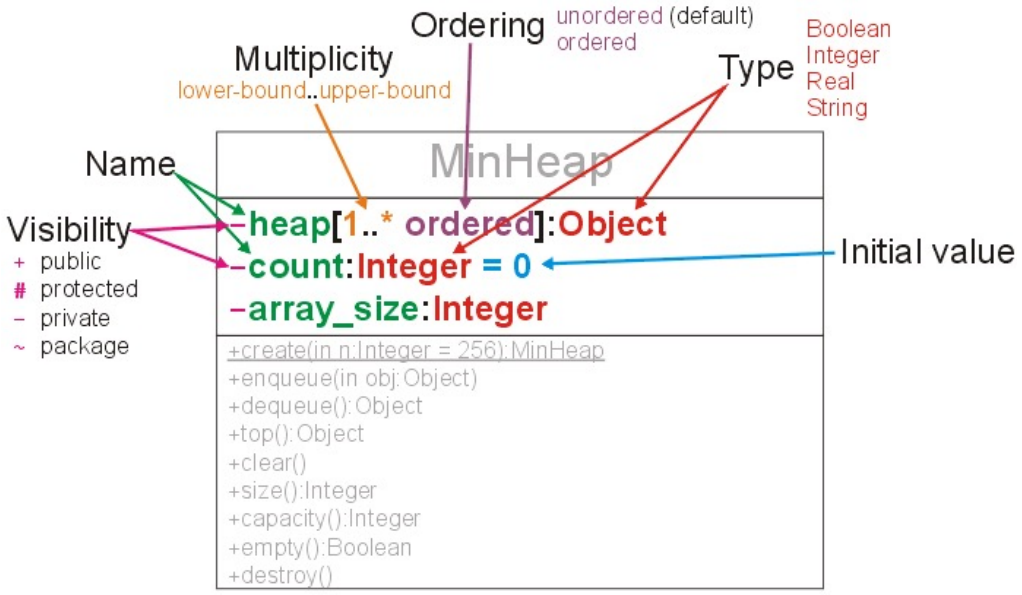
- Biểu diễn lớp




- Biểu diễn đối tượng

# Các thuộc tính (1)

- Thuộc tính là một tính chất có đặt tên của một lớp và nó nhận một giá trị cho mỗi đối tượng thuộc lớp đó tại mỗi thời điểm
- Cú pháp của thuộc tính:
- [tầm nhìn] [/] tên [: Kiểu] [cơ số] [= giá trị đầu] [{xâu tính chất}]
- https://www.uml-diagrams.org/property.html?context=class-diagrams



# Các thuộc tính (2)

- [tầm nhìn] [/] tên [: Kiểu] [cơ sở] [= giá trị đầu] [{xâu tính chất}]
- Tầm nhìn (visibility) cho biết thuộc tính đó được thấy và dùng từ các lớp khác như thế nào
  - Riêng tư (private), ký hiệu bởi dấu '-', nếu thuộc tính đó không thể truy cập được từ bất kỳ lớp khác
  - Bảo vệ (protected), ký hiệu bởi dấu '#', nếu thuộc tính đó chỉ có thể truy cập được từ các lớp kế thừa lớp hiện tại
  - Gói (package), ký hiệu bởi dấu '~', nếu thuộc tính đó có thể truy cập được từ các phần tử thuộc cùng một gói (hẹp nhất) với lớp hiện tại
  - Công cộng (public), ký hiệu bởi dấu '+', nếu thuộc tính đó có thể truy cập được từ bất kỳ lớp khác

# Các thuộc tính (3)

- [tầm nhìn] [/] tên [: Kiểu] [cơ số] [= giá trị đầu] [{xâu tính chất}]
- Kiểu (type): là kiểu của các giá trị của thuộc tính
  - Các kiểu cơ bản như Integer, Real, Boolean
  - Các kiểu có cấu trúc như Point, Area, Enumeration
  - Kiểu là một lớp khác
- Cơ số (multiplicity): là số các giá trị có thể nhận của thuộc tính
  - Ví dụ: [0..1] để chỉ thuộc tính này là tuỳ chọn (không nhận giá trị nào, hoặc nhận một giá trị)

# Các thuộc tính (4)

- [tầm nhìn] [/] tên [: Kiểu] [cơ số] [= giá trị đầu] [{xâu tính chất}]
- Giá trị đầu (initial value) là giá trị ngầm định gán cho thuộc tính khi một đối tượng được tạo lập (instantiated) từ lớp đó
- Xâu tính chất (property-string) để chỉ các giá trị có thể gán cho thuộc tính, thường dùng đối với một kiểu liệt kê
- Ví dụ:  tìnhtrạng: Tìnhtrạng = chưatrả {chưatrả, đãtrả}

# Các thuộc tính (4)

- Ngoài ra, mỗi thuộc tính lại có thể có phạm vi lớp (class-scope) nếu nó phản ánh đặc điểm chung của lớp chứ không phải của riêng đối tượng nào
  - Thuộc tính thuộc phạm vi lớp phải được gạch dưới
  - Ví dụ: Thuộc tính **số lượng các hoá đơn** trong lớp **Hoá đơn**
- Một thuộc tính là dẫn xuất, nếu giá trị của nó được tính từ giá trị của những thuộc tính khác của lớp
  - Thuộc tính dẫn xuất phải mang thêm dấu gạch chéo '/' ở đầu
  - Ví dụ: /tuổi (khi đã có ngày sinh)

# Các thao tác (1)

- Thao tác là một dịch vụ mà đối tượng có thể đáp ứng được khi được yêu cầu (thông qua một thông điệp)
- Các thao tác được cài đặt thành các phương thức (methods)
- Cú pháp đầy đủ của một thao tác là như sau:
- `[tầm nhìn] tên [(danh sách tham số)] [: Kiểu trả lại] [{xâu tính chất}]`
  - Tầm nhìn hoàn toàn giống tầm nhìn của thuộc tính

| CStudent |
| --- |
| - name<br>- address<br>- studentID<br>- nextAvailID : int |
| + addSchedule(theSchedule : Schedule, forSemester : Semester)<br>+ getSchedule(forSemester : Semester) : Schedule<br>+ hasPrerequisites(forCourseOffering : CourseOffering) : boolean<br># passed(theCourseOffering : CourseOffering) : boolean<br>+ getNextAvailID() : int |

# Các thao tác (2)

- Danh sách tham số là một danh sách gồm một số các tham số hình thức, cách nhau bằng dấu phẩy, mỗi tham số có dạng
- `[hướng] tên : Kiểu [= giá trị ngầm định]`
  - Hướng có thể lấy các giá trị in, out, inout và return tuỳ thuộc tham số là: không thể điều chỉnh (in), có thể điều chỉnh để đưa thông tin cho bên gọi (out), có thể điều chỉnh được (inout), hay là để trả lại kết quả cho bên gọi (return)
  - Giá trị ngầm định là giá trị được sử dụng khi trong lời gọi khuyết (thiếu) tham số tương ứng đó
- Xâu tính chất bao gồm các tiền điều kiện, hậu điều kiện, các tác động lên trạng thái đối tượng...

# Hai dạng lớp: phân tích và thiết kế

**Analysis**

| Order |
| --- |
| Placement Date<br>Delivery Date<br>Order Number |
| Calculate Total<br>Calculate Taxes |

**Bỏ qua các chi tiết không cần thiết**

**Design**

| Order |
| --- |
| - deliveryDate: Date<br>- orderNumber: int<br>- placementDate: Date<br>- taxes: Currency<br>- total: Currency |
| # calculateTaxes(Country, State): Currency<br># calculateTotal(): Currency<br>getTaxEngine() {visibility=implementation} |

**Phải đầy đủ & chi tiết các thành phần**

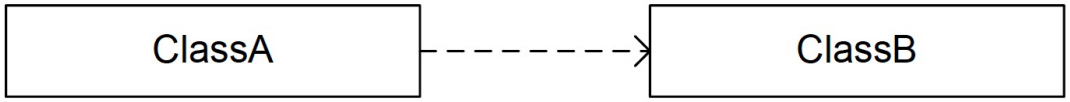
# Mối liên quan giữa các lớp

# Mối liên quan

- Giữa các lớp có thể có ba mối liên quan
  - Mối liên quan phụ thuộc
  - Mối liên quan khái quát hóa
  - Mối liên quan liên kết

# Mối liên quan phụ thuộc (1)

- Mối liên quan phụ thuộc (dependency relationship) được dùng để diễn đạt một lớp (bên phụ thuộc) chịu ảnh hưởng của mọi thay đổi trong một lớp khác (bên độc lập)
  - Chiều ngược lại thì không nhất thiết
  - Thường thì bên phụ thuộc cần dùng bên độc lập để đặc tả hay cài đặt cho mình
- **UML biểu diễn mối liên quan phụ thuộc bằng một mũi tên đứt nét từ bên phụ thuộc sang bên độc lập**




Tham số truyền vào

```
class A
{
   void   F(B   x)
   {
    …
   }
};
```

Kết quả trả ra

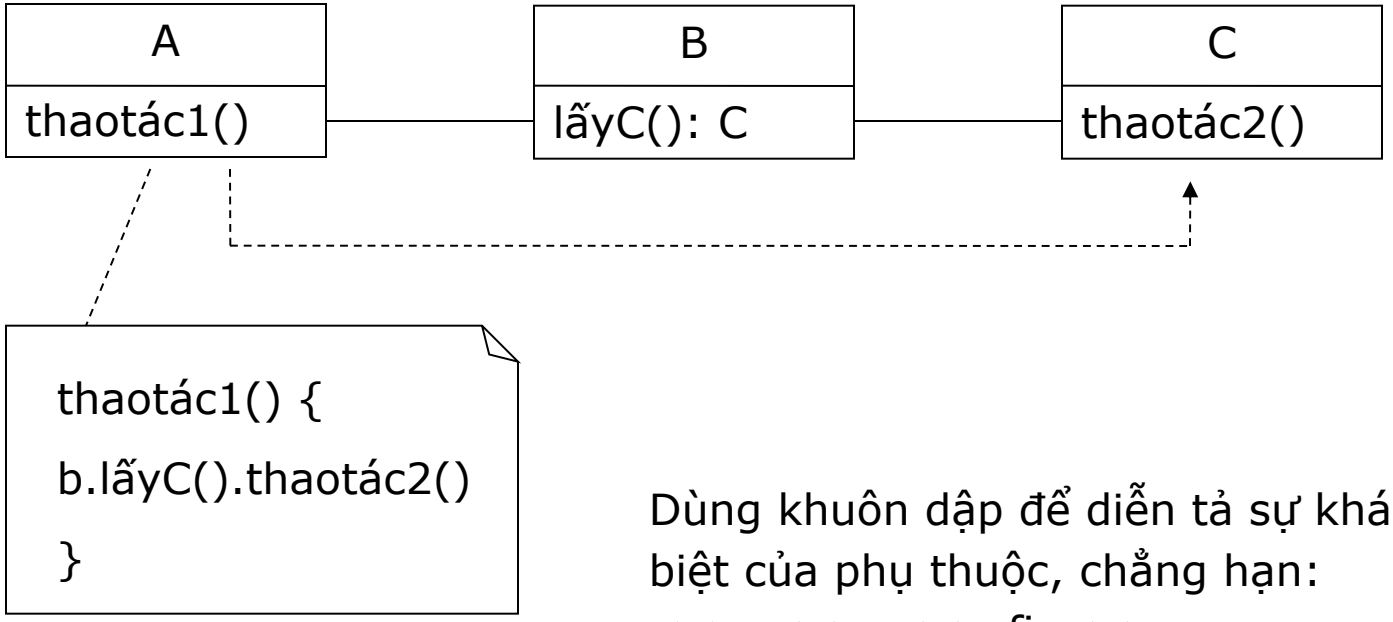
```
class A
{
   B   F()
   {
    …
   }
};
```

Biến cục bộ

```
class A
{
   void   F()
   {
      B   x;
   }
};
```

Trong ClassA có sử dụng biến toàn cục (kiểu B), hoặc sử dụng phương thức/thuộc tính static của ClassB

# Mối liên quan phụ thuộc (2)



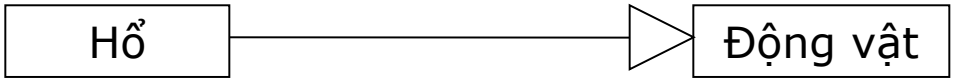

Dùng khuôn dập để diễn tả sự khác biệt của phụ thuộc, chẳng hạn: <<use>>, <<refine>>

# Mối liên quan khái quát hóa (1)

- Khái quát hoá (generalization) là sự rút ra các đặc điểm chung của nhiều lớp để tạo thành một lớp giản lược hơn
  - Được gọi là lớp cha (superclass)
- Ngược lại, cụ thể hoá (specialization) là sự tăng cường (bổ sung) thêm một số đặc điểm mới từ một lớp đã cho, tạo thành một lớp cụ thể hơn
  - Được gọi là lớp con (subclass)
- Một lớp có thể không có lớp cha, có một hay nhiều lớp cha
  - Một lớp chỉ có một lớp cha gọi là lớp thừa kế đơn (simple inheritance)
  - Một lớp có nhiều lớp cha được gọi là lớp thừa kế bội (multiple inheritance)
- Một lớp không có lớp cha và có lớp con, thì được gọi là **lớp gốc (lớp cơ sở)**

# Mối liên quan khái quát hóa (2)

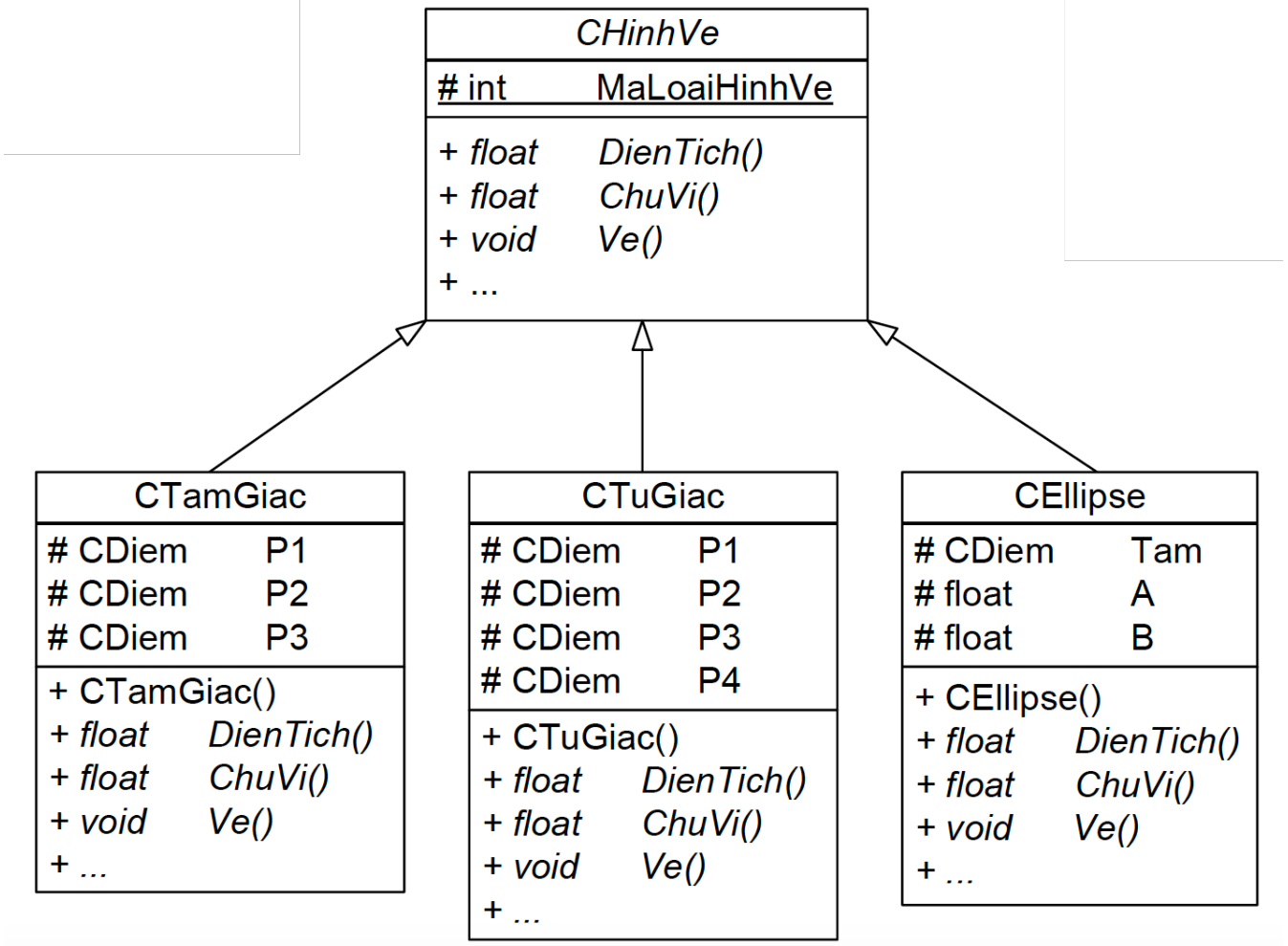
- Thuật ngữ thừa kế (inherit) được sử dụng phổ biến trong các ngôn ngữ lập trình, nhằm diễn tả một lớp con có mọi thuộc tính, thao tác và liên kết được mô tả ở một lớp cha
  - Lớp con có thể có thêm (riêng) các thuộc tính, các thao tác và các liên kết mới
  - Lớp con có thể định nghĩa lại (overwrite) một thao tác của lớp cha: Sự đa hình (polymorphism)
- Biểu diễn của liên quan khái quát hoá:



# Lớp trừu tượng

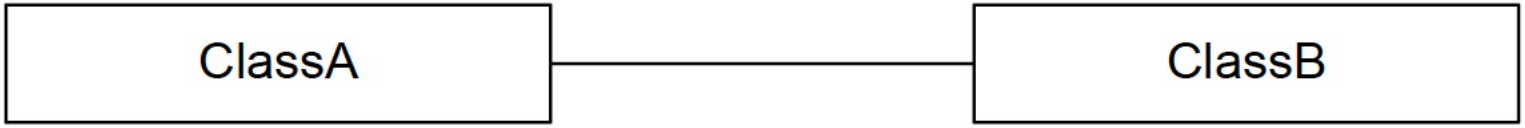
- Thông qua sự khái quát hoá, ta có thể làm xuất hiện các **lớp trừu tượng (abstract class) là các lớp không có đối tượng cá thể** (no instances), mà chỉ dùng để mô tả các đặc điểm chung của những lớp con (lớp dưới) mà thôi
- Một lớp trừu tượng thường chứa các thao tác trừu tượng (abstract method) là các thao tác chỉ có tiêu đề (tên) mà không có cài đặt
  - Thao tác trừu tượng phải được định nghĩa lại và kèm cài đặt ở các lớp con
  - Tên của lớp trừu tượng và tiêu đề của thao tác trừu tượng phải được viết nghiêng và có thể kèm thêm xâu tính chất {abstract}
- Ví dụ: Động vật là lớp trừu tượng được khái quát hoá từ các lớp Ngựa, Hổ, Dơi. Nó có một thao tác trừu tượng là ngủ(). Thao tác trừu tượng này sẽ được cài đặt cụ thể hoá trong các lớp con là Ngựa, Hổ, Dơi theo các cách thức khác nhau, nhưng vẫn giữ nguyên tên là ngủ()

# Ví dụ


CHinhVe
# int MaLoaiHinhVe
+ float DienTich()
+ float ChuVi()
+ void Ve()
+ ...
CTamGiac
# CDiem P1
# CDiem P2
# CDiem P3
+ CTamGiac()
+ float DienTich()
+ float ChuVi()
+ void Ve()
+ ...
CTuGiac
# CDiem P1
# CDiem P2
# CDiem P3
# CDiem P4
+ CTuGiac()
+ float DienTich()
+ float ChuVi()
+ void Ve()
+ ...
CEllipse
# CDiem Tam
# float A
# float B
+ CEllipse()
+ float DienTich()
+ float ChuVi()
+ void Ve()
+ ...

# Mối liên quan liên kết (1)

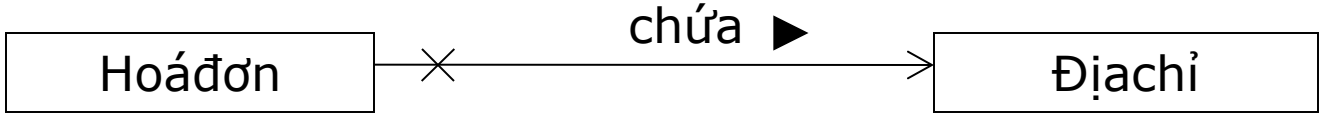
- **Kết nối (link):** thể hiện một mối liên hệ nào đó trên thực tế **giữa các cá thể (đối tượng)** của hai lớp
  - Ví dụ: kết nối vợ - chồng, kết nối thầy - trò, kết nối xe máy - chủ xe, kết nối khách hàng - hóa đơn, ...
- **Liên kết (association):** là mối **liên quan giữa hai lớp**, bao gồm tập hợp những kết nối cùng loại (cùng ý nghĩa) giữa các cá thể của hai lớp đó
  - Đây chính là một quan hệ (hiểu theo nghĩa toán học) giữa hai tập hợp (là hai lớp)




- Về lập trình
  - Trong ClassA có thuộc tính có kiểu là ClassB
  - Hoặc trong ClassB có thuộc tính có kiểu là ClassA

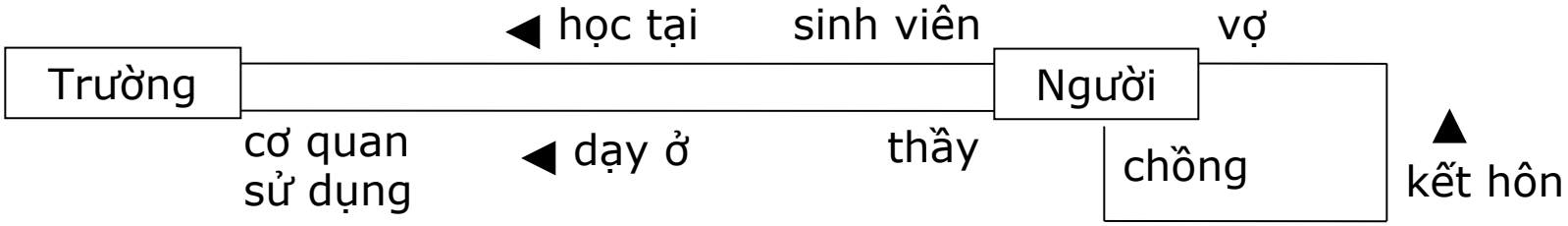
# Mối liên quan liên kết (2)

- Sự lưu hành (navigation) trên một liên kết
  - Sự tồn tại một kết nối giữa hai đối tượng (của hai lớp trong một liên kết) có nghĩa là các đối tượng đó "biết nhau".
    - Nhờ có kết nối, mà từ một đối tượng này, ta có thể tìm đến được đối tượng kia
  - Sự lưu hành có thể thực hiện theo cả hai chiều (tức là cả ở hai đầu) trên một liên kết
  - Sự lưu hành có thể bị hạn chế theo một chiều
    - Khi đó, ta thêm một mũi tên vào đầu được lưu hành và dấu chéo vào đầu không được lưu hành



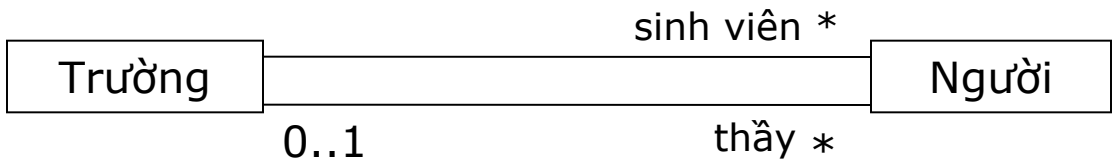
# Mối liên quan liên kết (3)

- Vai trò (role)
  - Trong một liên kết giữa hai lớp, thì mỗi lớp đóng một vai trò khác nhau
  - Tên vai trò (với chữ cái đầu tiên viết thường) có thể được viết thêm vào mỗi đầu của liên kết (vì vậy mà vai trò cũng được gọi là tên của một đầu liên kết)
  - Về ý nghĩa, thì một vai trò biểu diễn cho một tập con các đối tượng của lớp tương ứng



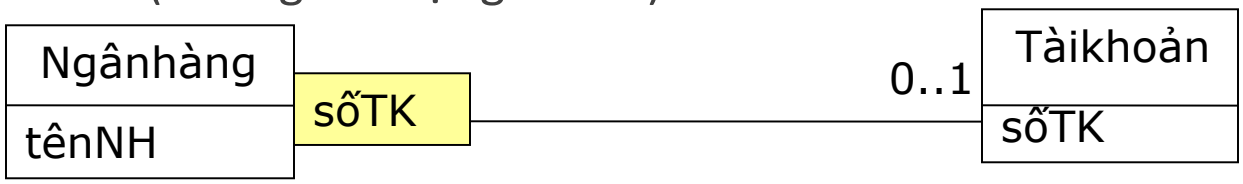
# Mối liên quan liên kết (4)

- Cơ số (multiplicity)
  - Mỗi đầu của liên kết còn có thể chứa thêm một cơ số để cho biết số (tối thiểu và tối đa) các cá thể của đầu đó tham gia liên kết với một cá thể ở đầu kia
  - Các giá trị của cơ số thường dùng là:
    - 1 1 và chỉ một
    - 0..1 0 hoặc 1
    - m..n Từ m tới n (m và n là các số tự nhiên)
    - 0..* hoặc * Từ 0 tới nhiều
    - 1..* Từ 1 tới nhiều



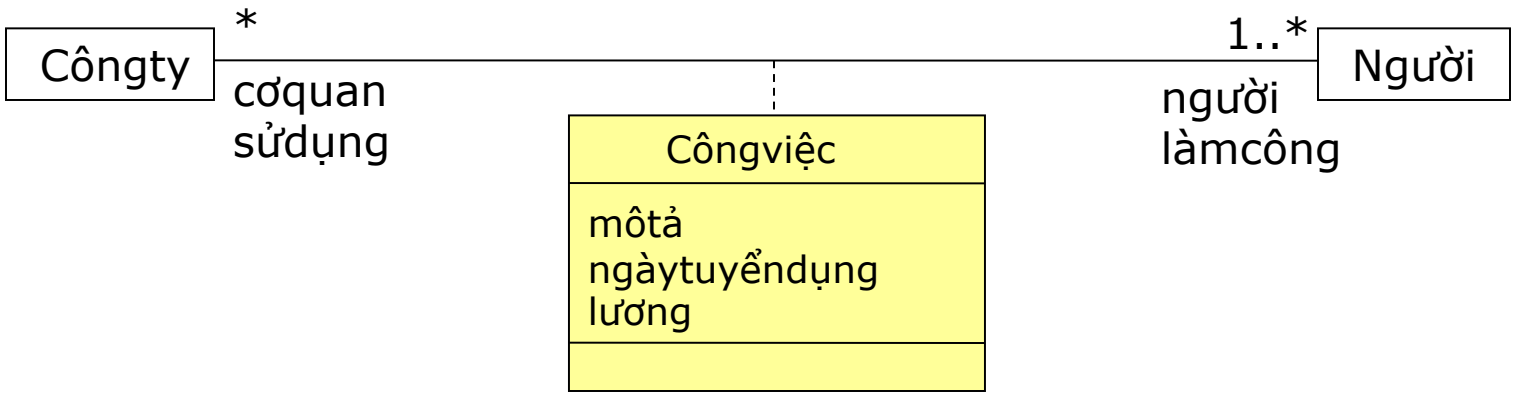
# Mối liên quan liên kết (5)

- Hạn định (qualifier)
  - Vấn đề luôn luôn cần giải quyết khi mô hình hóa các liên kết, đó là vấn đề tìm kiếm: cho trước một đối tượng ở một đầu của liên kết, hãy tìm một đối tượng hay tập hợp các đối tượng kết nối với nó ở đầu kia
  - Để giảm bớt số lượng các đối tượng tìm được, ta có thể giới hạn khu vực tìm kiếm theo (giá trị của) một số thuộc tính nào đó
  - Các thuộc tính này được gọi là các hạn định (qualifier)
    - Mỗi thuộc tính như vậy được ghi trong một hộp nhỏ gắn vào đầu mút của liên kết, **phía lớp xuất phát của sự lưu hành**
  - **Như vậy hạn định được áp dụng cho các liên kết 1-nhiều hay nhiều-nhiều**, để giảm từ nhiều xuống 1 (hay 0..1), hoặc để giảm từ nhiều xuống nhiều (nhưng số lượng ít hơn)



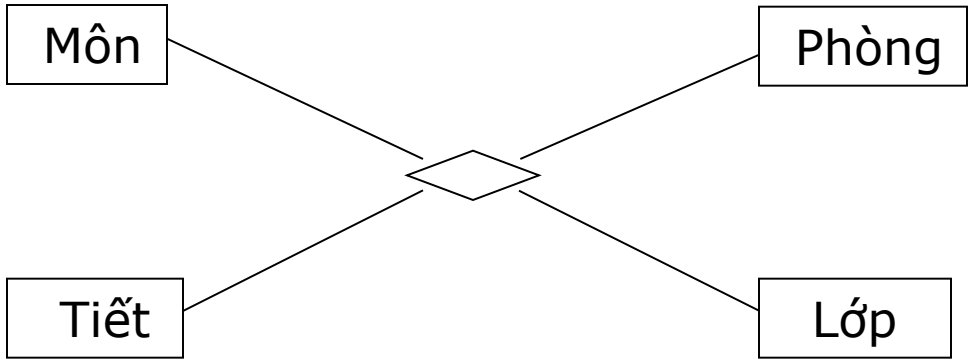
# Mối liên quan liên kết (6)

- Lớp liên kết (association class)
  - Bản thân các liên kết (association) cũng có thể cần có các tính chất đặc trưng cho nó
  - UML thể hiện điều này bằng các lớp liên kết (association class)
  - Lớp liên kết là một lớp như bình thường (có các thuộc tính, các thao tác và tham gia liên kết với những lớp khác), nhưng ngăn tên có thể có tên hay để trống tùy ý, và nó được gắn với liên kết mô tả bởi một đường đứt nét



# Mối liên quan liên kết (7)

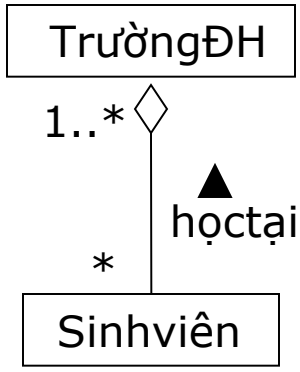
- Liên kết nhiều bên
  - Có thể có liên kết giữa nhiều hơn hai lớp, theo nghĩa của quan hệ nhiều ngôi (một kết nối ở đây là một bộ -n). Lúc đó liên kết được diễn tả bởi một hình thoi nhỏ, nối bằng các đường liền nét với các lớp tham gia và cũng có thể có một lớp liên kết cho nó.



# Mối liên quan liên kết: Kết nhập (aggregation)

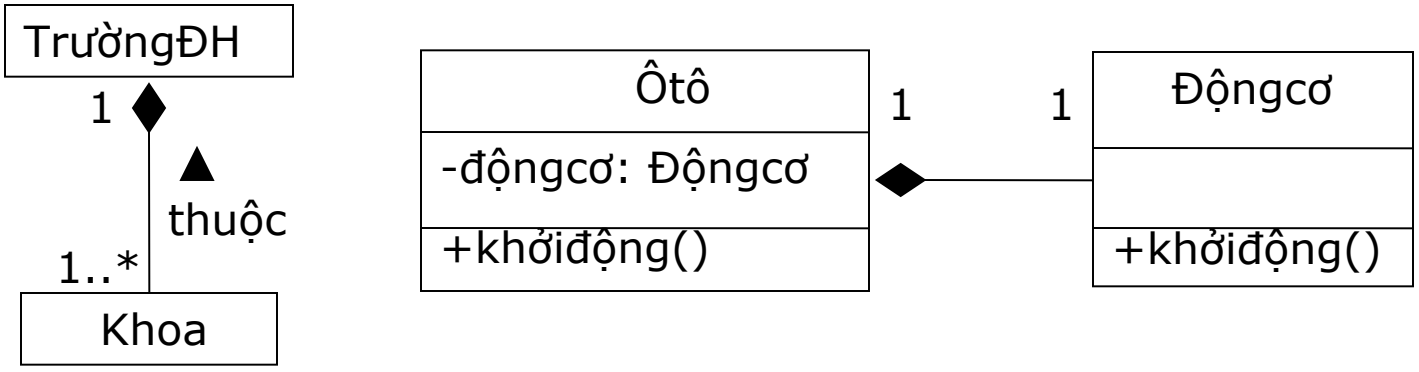
- **Kết nhập (aggregation)**
  - Trong một liên kết, hai bên tham gia được xem là bình đẳng, không bên nào được nhấn mạnh hơn bên nào
  - Tuy nhiên, cũng có lúc ta muốn mô hình hóa **mối quan hệ "toàn thể/bộ phận"** giữa một lớp các vật thể lớn (cái "toàn thể") với một lớp các vật thể bé (các "bộ phận") bao gồm trong chúng. Đó là loại liên kết kết nhập (aggregation) được biểu diễn bằng cách gắn thêm một hình thoi rỗng vào một đầu của liên kết, phía cái toàn thể

- Ví dụ: Lớp Sinhviên (lớp bộ phận) có mối quan hệ kết nhập với lớp TrườngĐH (lớp toàn thể); và 1 đối tượng thuộc lớp Sinhviên có quan hệ “học tại” với nhiều đối tượng thuộc lớp TrườngĐH



# Mối liên quan liên kết: Hợp thành (composition)

- **Hợp thành (composition)**
  - Một hợp thành (composition) là một loại kết nhập (aggregation) đặc biệt với quan hệ sở hữu mạnh hơn, trong đó **một bộ phận chỉ thuộc vào một cái toàn thể duy nhất (không thể tồn tại độc lập) và cái toàn thể có trách nhiệm tạo lập và hủy bỏ cái bộ phận.**
  - **Khi cái toàn thể bị hủy bỏ, thì cái bộ phận cũng bị hủy bỏ theo**
  - Hợp thành được biểu diễn bằng cách gắn thêm một hình thoi đặc vào một đầu của liên kết, phía cái toàn thể



# Liên quan phụ thuộc vs liên kết

**Dependency** (references)

It means there is no conceptual link between two objects. e.g. EnrollmentService object references Student & Course objects (as method parameters or return types)

```
public class EnrollmentService {
    public void enroll(Student s, Course c){}
}
```

**Association** (has-a)

It means there is almost always a link between objects (they are associated). Order object ***has a*** Customer object

```
public class Order {
    private Customer customer
}
```

In OOP terms:
Association --> **A *has-a* C** object (as a member variable)
Dependency --> **A *references* B** (as a method parameter or return type)

# Liên quan kết nhập

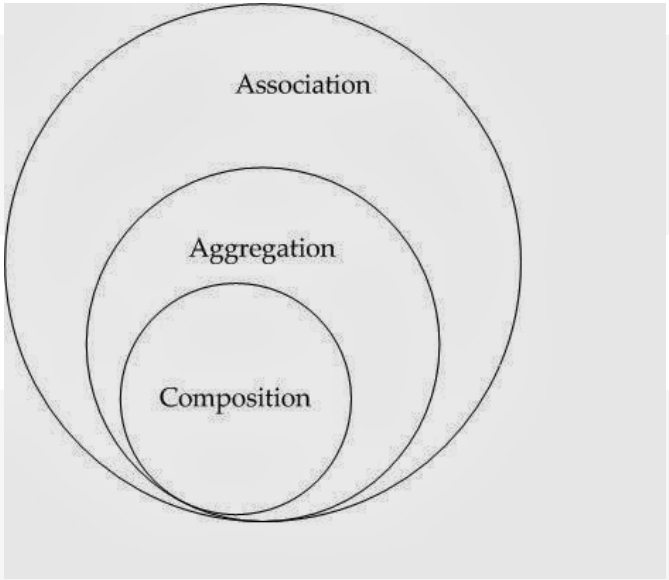
**Aggregation** (has-a + whole-part)

Special kind of association where there is whole-part relation between two objects. they might live without each other though.

```
public class PlayList {
    private List<Song> songs;
}
```

OR

```
public class Computer {
    private Monitor monitor;
}
```

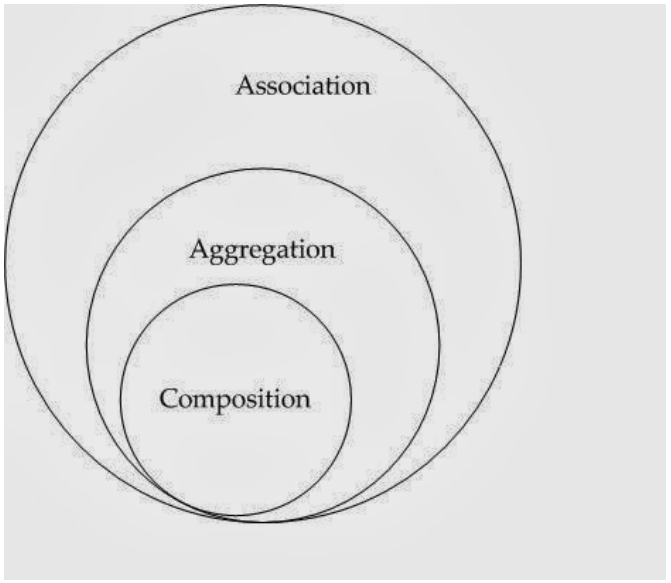



*Note:* the trickiest part is to distinguish aggregation from normal association. Honestly, I think this is open to different interpretations.

# Liên quan hợp thành

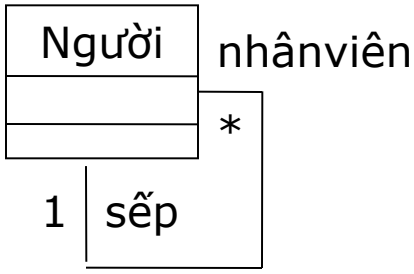
**Composition** (has-a + whole-part + ownership)
Special kind of aggregation. An `Apartment` is composed of some `Room`s. A `Room` cannot exist without an `Apartment`. when an apartment is deleted, all associated rooms are deleted as well.

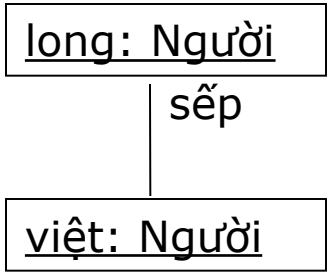
```
public class Apartment{
    private Room bedroom;
    public Apartment() {
       bedroom = new Room();
    }
}
```

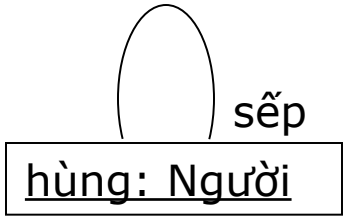


# Biểu đồ lớp và biểu đồ đối tượng

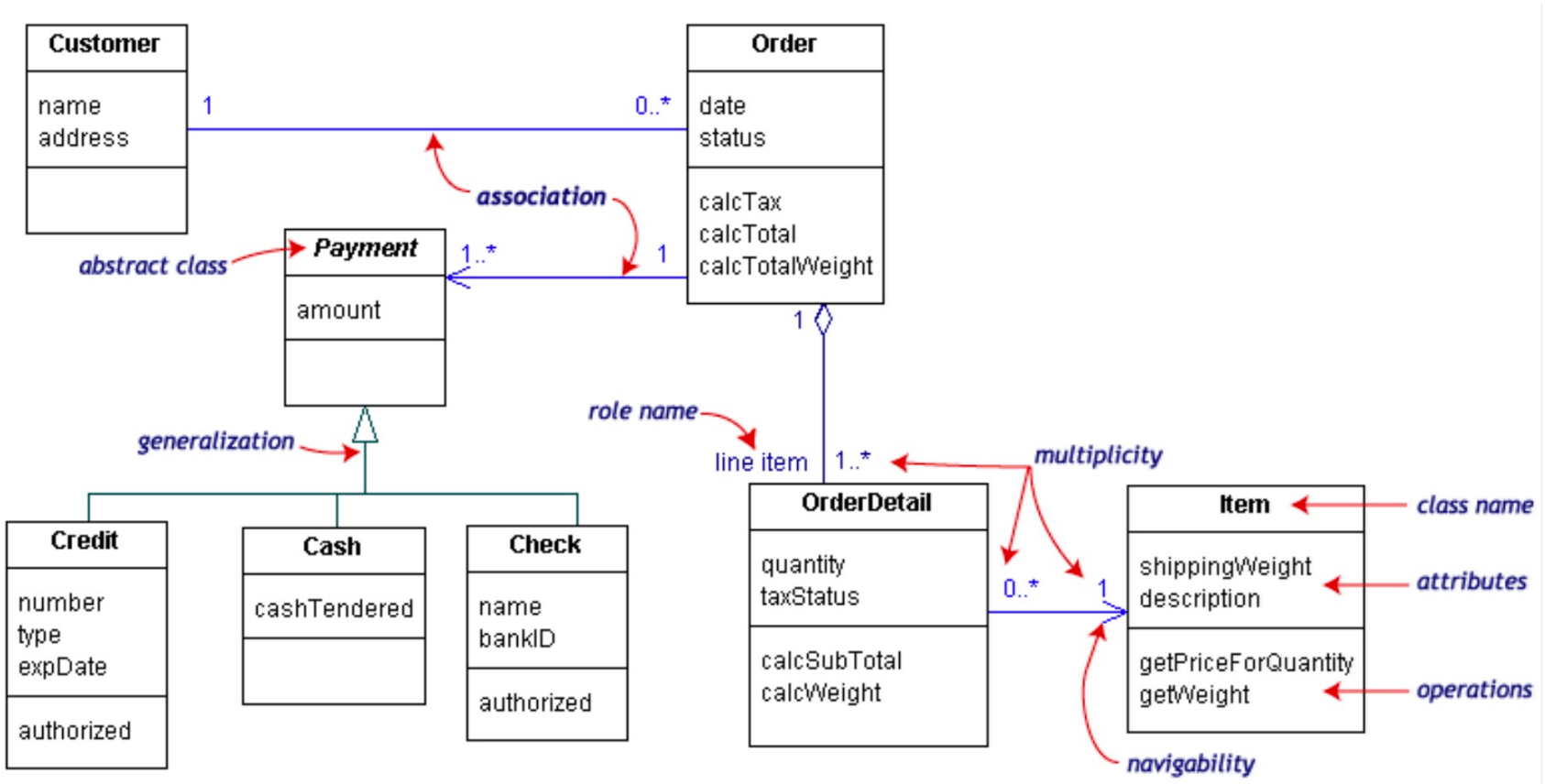
- Biểu đồ lớp là một biểu đồ thể hiện một tập hợp các lớp và các mối liên quan (liên kết, kết nhập, hợp thành, khái quát hoá, phụ thuộc, và thực hiện) có thể có giữa chúng
- Biểu đồ lớp được dùng để mô hình hoá cấu trúc tĩnh, bao gồm mọi phần tử khai báo
- Biểu đồ đối tượng diễn tả lại cấu trúc tĩnh trong biểu đồ lớp một cách cụ thể
  - **Các đối tượng thay cho các lớp, Các kết nối thay cho các liên kết**



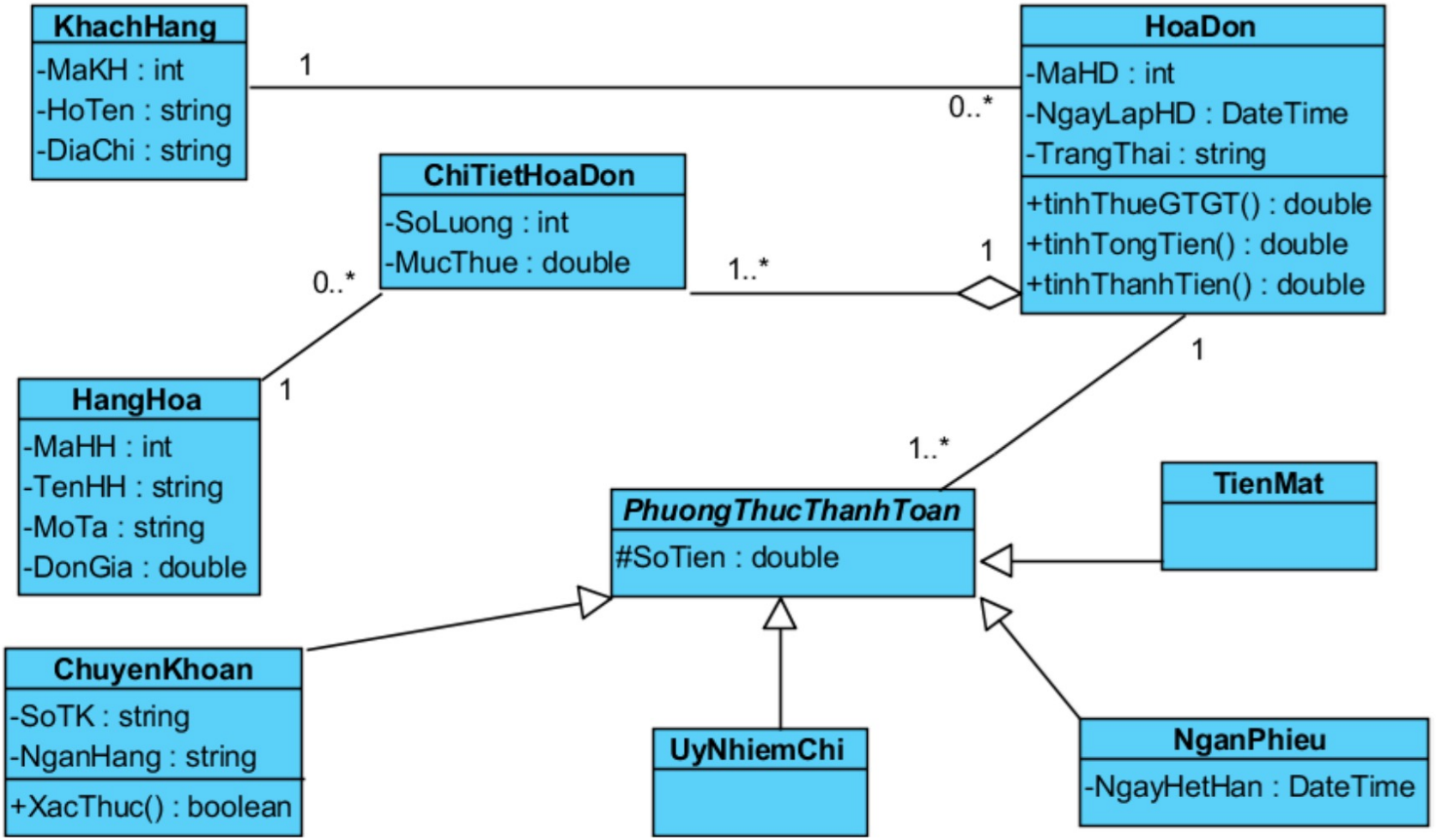



# Ví dụ


Customer
name
address
1
0..*
Order
date
status
association
calcTax
calcTotal
calcTotalWeight
abstract class
Payment
amount
1..*
1
1
role name
line item
1..*
multiplicity
generalization
Credit
number
type
expDate
authorized
Cash
cashTendered
Check
name
bankID
authorized
OrderDetail
quantity
taxStatus
calcSubTotal
calcWeight
0..*
1
Item
shippingWeight
description
getPriceForQuantity
getWeight
class name
attributes
operations
navigability

# Ví dụ (2)


KhachHang
-MaKH : int
-HoTen : string
-DiaChi : string
1
0..*
HoaDon
-MaHD : int
-NgayLapHD : DateTime
-TrangThai : string
+tinhThueGTGT() : double
+tinhTongTien() : double
+tinhThanhTien() : double
ChiTietHoaDon
-SoLuong : int
-MucThue : double
0..*
1..*
1
1
HangHoa
-MaHH : int
-TenHH : string
-MoTa : string
-DonGia : double
1
1..*
PhuongThucThanhToan
#SoTien : double
TienMat
ChuyenKhoan
-SoTK : string
-NganHang : string
+XacThuc() : boolean
UyNhiemChi
NganPhieu
-NgayHetHan : DateTime

# Phát hiện các lớp lĩnh vực

# Phát hiện các lớp lĩnh vực

- Mục đích và trình tự tiến hành
- Nhận định các khái niệm lĩnh vực
- Thêm các liên kết và các thuộc tính
- Khái quát hóa các lớp

# Mục đích và trình tự tiến hành

- Xuất phát từ các khái niệm về các sự vật trong lĩnh vực ứng dụng, ta trừu tượng hoá chúng thành các lớp gọi là các lớp lĩnh vực
- Các lớp lĩnh vực này thường chỉ dùng để phản ánh và mô phỏng các sự vật trong thế giới thực, cho nên vai trò của chúng cũng thường chỉ là lưu giữ và cung cấp các thông tin về các sự vật đó
- Trình tự tiến hành:
  - Nhận định các khái niệm của lĩnh vực
  - Xác định các liên kết và các thuộc tính
  - Khái quát hoá các khái niệm

# Nhận định các khái niệm của lĩnh vực (1)

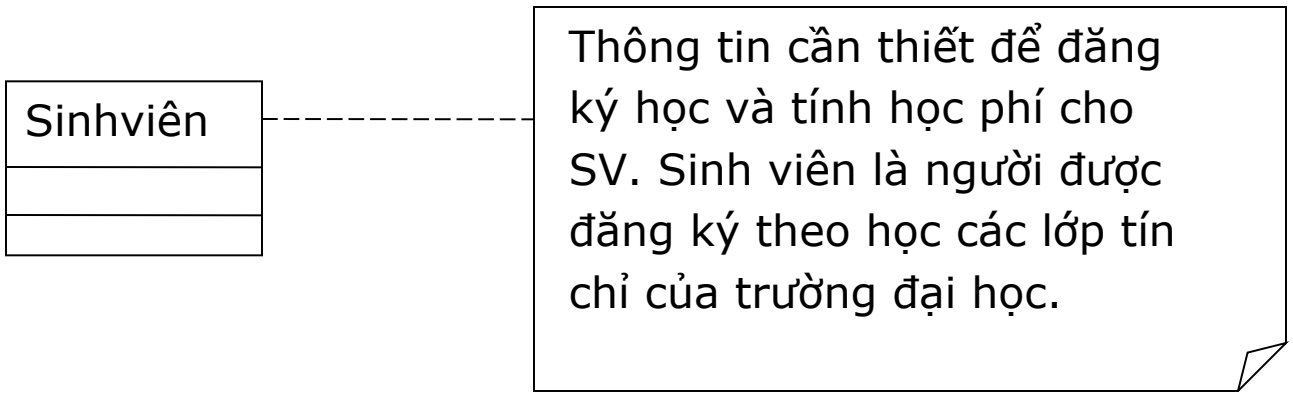
- Nguồn tìm kiếm
  - Các khái niệm của lĩnh vực là những khái niệm về các sự vật (cụ thể hay trừu tượng) mà các người dùng, các chuyên gia nghiệp vụ sử dụng khi nói đến lĩnh vực đó
  - Để tìm kiếm các khái niệm của lĩnh vực, ta dựa vào:
    - Các kiến thức về lĩnh vực nghiệp vụ
    - Các cuộc phỏng vấn trao đổi với các người dùng và chuyên gia
    - Tài liệu mô tả tổng quan về hệ thống và nhu cầu
    - Các tài liệu mô tả các ca sử dụng (đã được lập ở bước trước của quy trình phát triển phần mềm)

# Nhận định các khái niệm của lĩnh vực (2)

- Cách xác định các khái niệm
  - Đọc tài liệu mô tả hệ thống (phát biểu yêu cầu):
    - Các danh từ sẽ có thể là đối tượng hay thuộc tính,
    - Các động từ sẽ có thể là các thao tác
  - Dựa vào đó, xác định các đối tượng sau đâu:
    - Các thực thể (ví dụ: xe đạp, máy bay, cảm biến, ...)
    - Các vai trò (ví dụ: làm mẹ, giảng dạy, giám sát, ...)
    - Các sự kiện (ví dụ: hạ cánh, ngắt, đăng ký xe máy, ...)
    - Các tương tác (ví dụ: cho vay, thảo luận, ...)
    - Các tổ chức (ví dụ: công ty, khoa, lớp, ...)

# Nhận định các khái niệm của lĩnh vực (3)

- Đặt tên và gán trách nhiệm
  - Tiếp đến là đặt tên và gán trách nhiệm cho mỗi lớp vừa được xác định
  - Trách nhiệm mô tả vai trò và mục đích sử dụng của lớp, chứ không phải là cấu trúc của lớp
    - Mặc dù sau này trách nhiệm sẽ cho phép ta xác định cấu trúc (thuộc tính và liên kết) cùng với hành vi (các thao tác) của lớp



# Nhận định các khái niệm của lĩnh vực (4)

- Đặt tên và gán trách nhiệm…
  - Việc gán tên và trách nhiệm cho một lớp cũng giúp kiểm tra xem việc chọn lựa lớp là có hợp lý không:
    - Nếu chọn được tên và gán được trách nhiệm rõ ràng, chặt chẽ thì lớp đề cử là tốt;
    - Nếu chọn được tên, song trách nhiệm lại giống trách nhiệm của một lớp khác, thì nên gộp hai lớp đó làm một;
    - Nếu chọn được tên, song trách nhiệm lại quá dông dài, thì nên tách lớp đó ra thành nhiều lớp;
    - Khó chọn được tên hợp lý hay khó mô tả trách nhiệm, thì nên phân tích sâu thêm, để chọn những biểu diễn thích hợp.

# Thêm các liên kết và các thuộc tính (1)

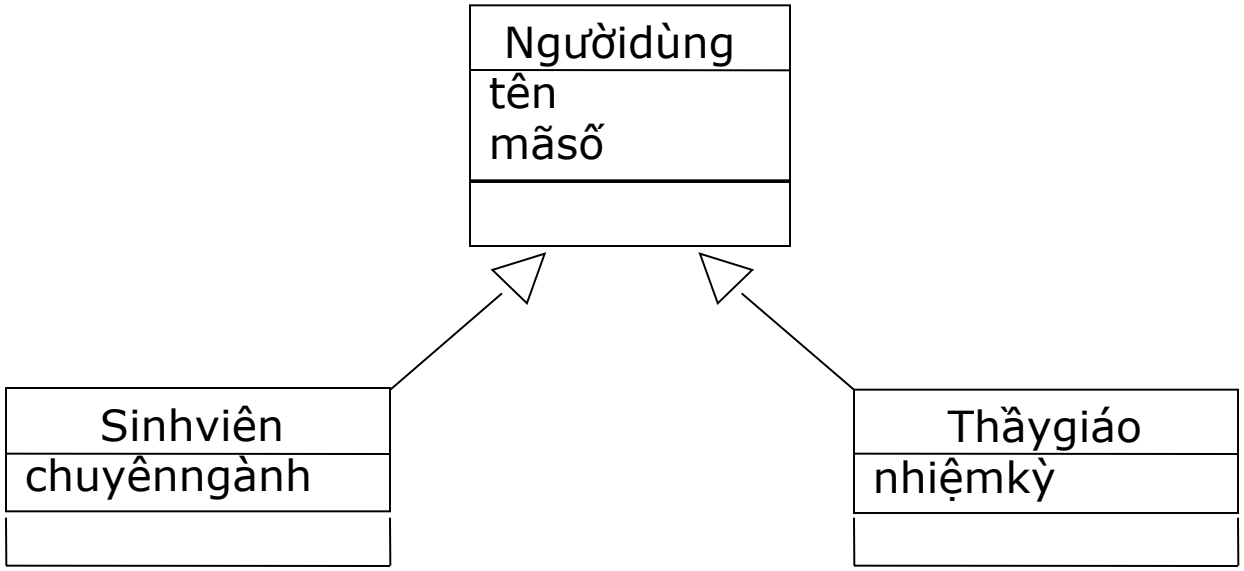
- Trước tiên, nhiều thuộc tính và liên kết của các lớp lĩnh vực đã có thể phát hiện trực tiếp:
  - Từ tài liệu mô tả hệ thống và nhu cầu,
  - Từ ý kiến của các chuyên gia lĩnh vực và người dùng, và
  - Từ các trách nhiệm của các lớp mà ta vừa xác định ở bước trước.
- Sau này, sẽ bổ sung thêm các liên kết và các thuộc tính, cũng như bổ sung thêm các thao tác cho các lớp, khi ta nghiên cứu sâu vào hành vi (tương tác và ứng xử) của hệ thống

# Thêm các liên kết và các thuộc tính (2)

- Ví dụ: Từ mô tả trách nhiệm của lớp “Sinhviên”, ta có:
  - Câu "Thông tin cần thiết để đăng ký học và tính học phí" giúp ta suy ra được một số thuộc tính như tên, mã SV, địa chỉ, ... cho lớp “Sinhviên”
  - Câu "Sinh viên là người được đăng ký theo học các lớp tín chỉ của trường đại học" giúp ta suy ra là có liên kết giữa lớp “Sinhviên” và lớp “Lớptínchỉ”, với tên liên kết có thể là "đăng ký"

# Khái quát hóa các lớp

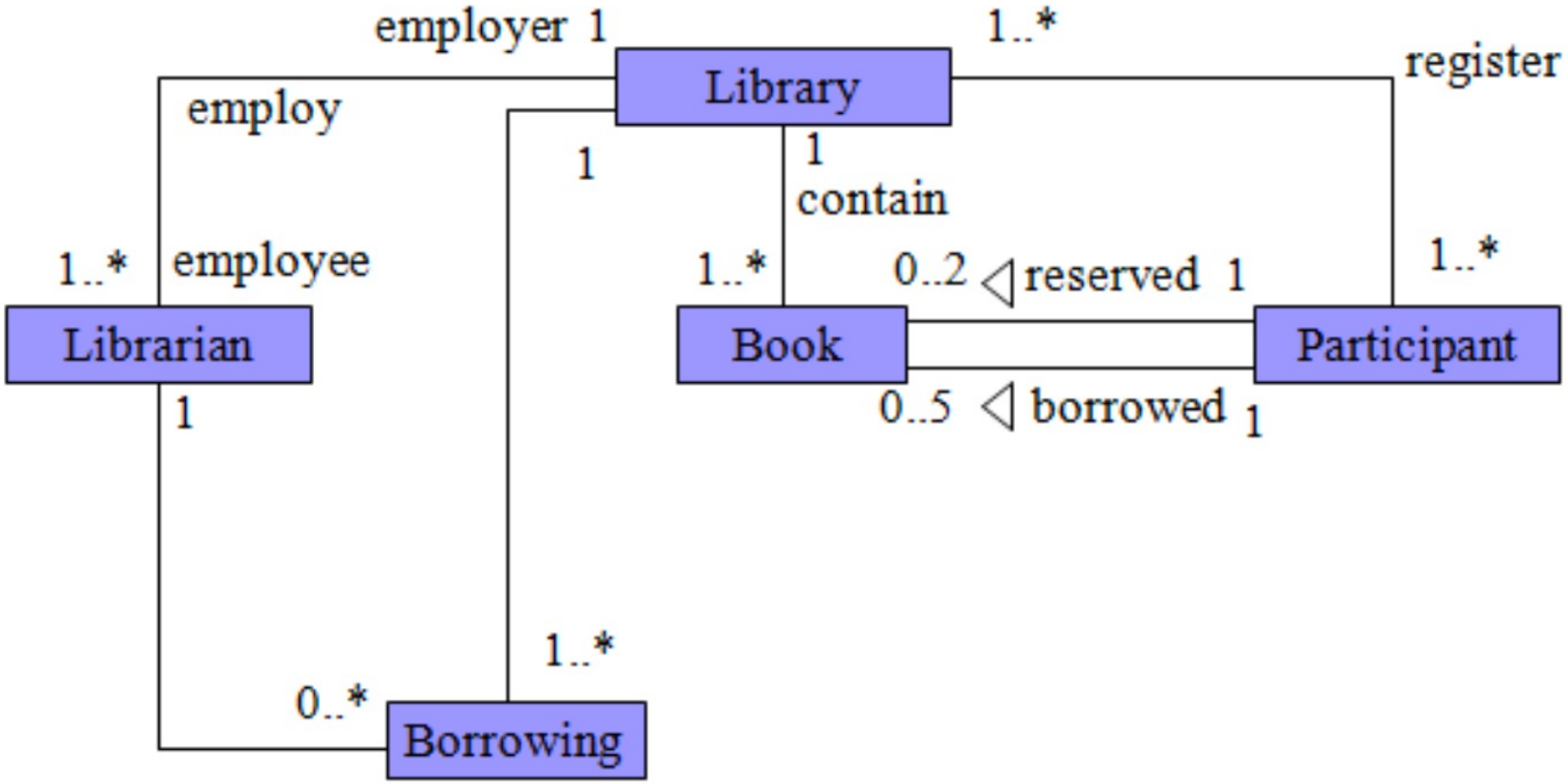
- Để cải thiện tối ưu mô hình biểu diễn lớp, ta tìm cách rút ra các phần chung giữa các lớp để lập thành lớp khái quát hơn
- Chẳng hạn các lớp “Sinhviên” và “Thầygiáo” có những thuộc tính chung (ví dụ: tên, mã số, …) => có thể xác lập một lớp khái quát hóa hơn (ví dụ: lớp “Ngườidùng”)



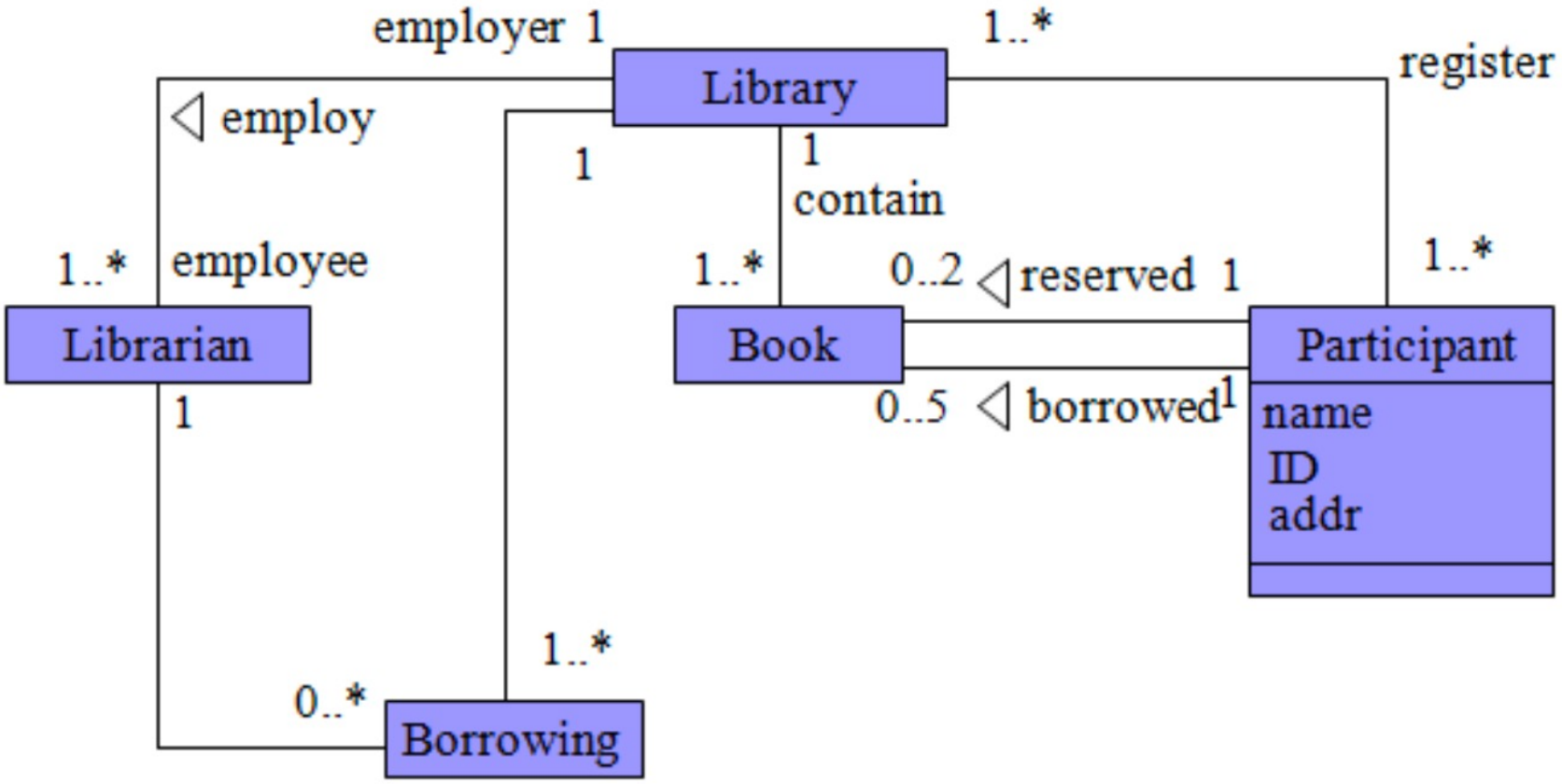
# Bài tập

- **Mô hình hóa biểu đồ lớp cho hệ thống quản lý thư viện**
  - Người quản lý thư viện mong muốn tự động hóa việc *mượn sách*
  - Họ yêu cầu một phần mềm cho phép người sử dụng biết sách hiện có, có thể đặt mượn 2 quyển sách, những người tham gia mượn sách có thể biết sách nào đã mượn hoặc đã đặt
  - Những người tham gia mượn sách sở hữu một password để *truy nhập*
  - Việc mượn sách được thực hiện bởi các thủ thư, sau khi xác định người mượn sách, họ biết được người này có được phép mượn hay không? (tối đa 5 quyển), người này được ưu tiên? (đã đặt trước)

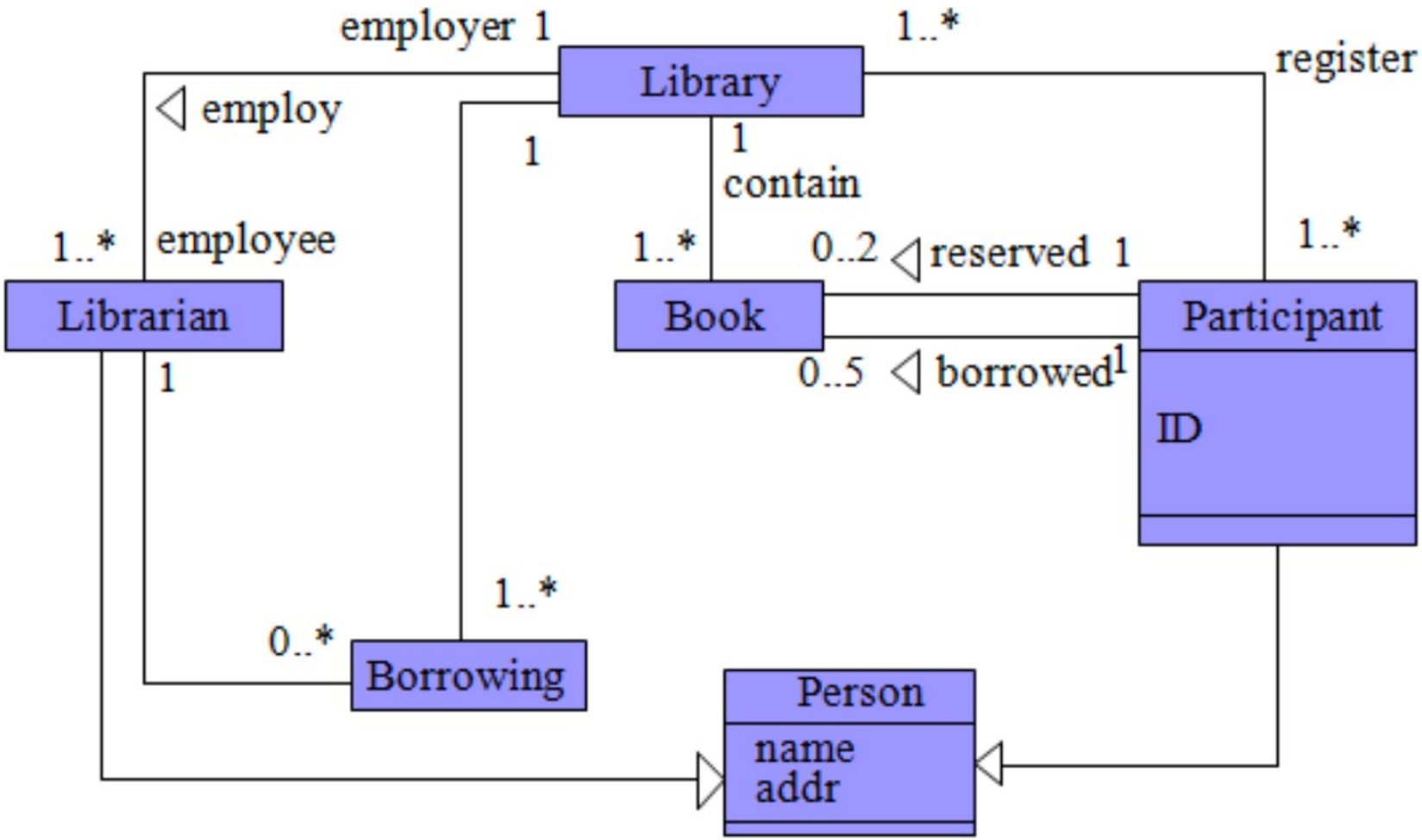
# Xác định các liên kết



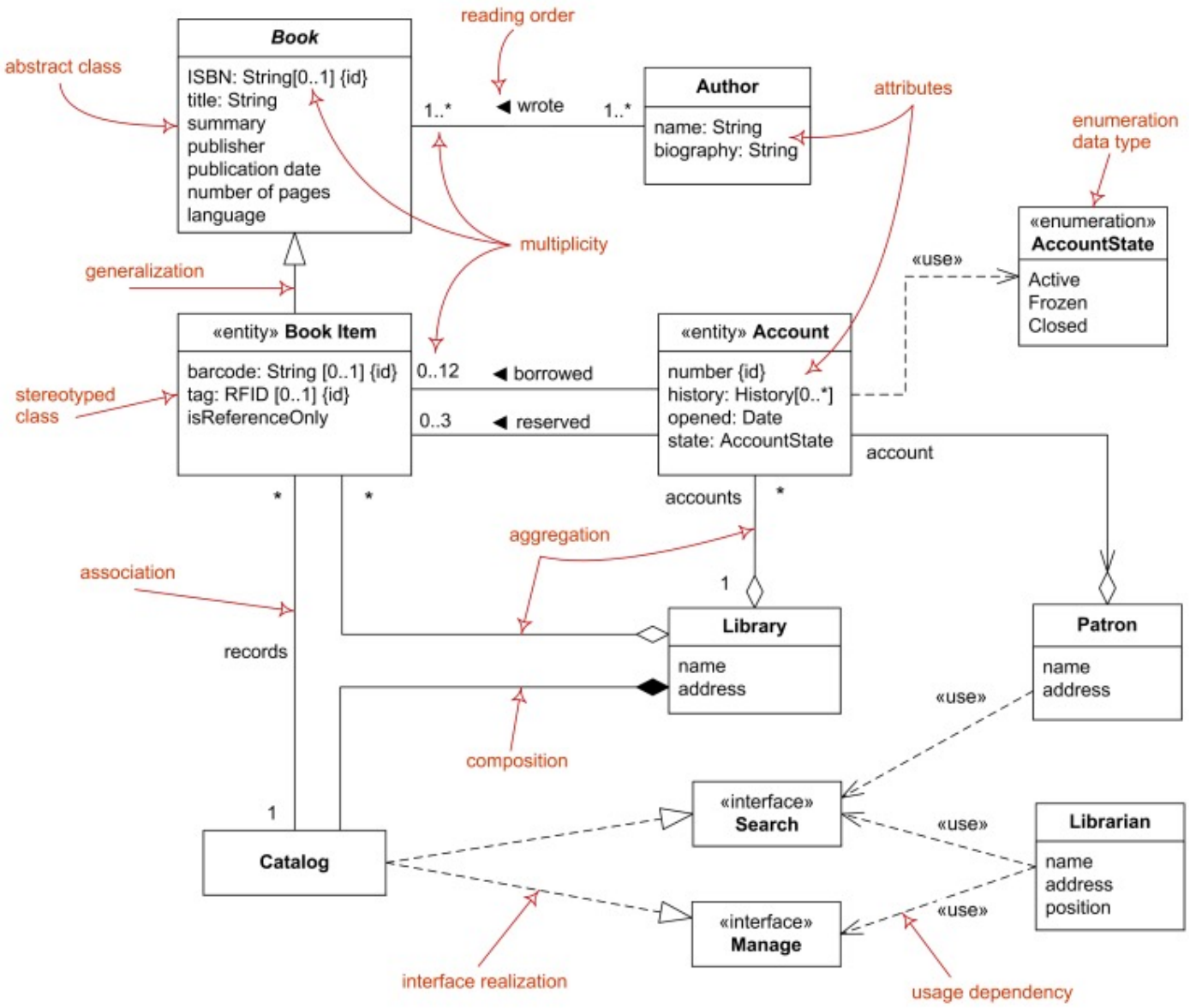
# Xác định các thuộc tính



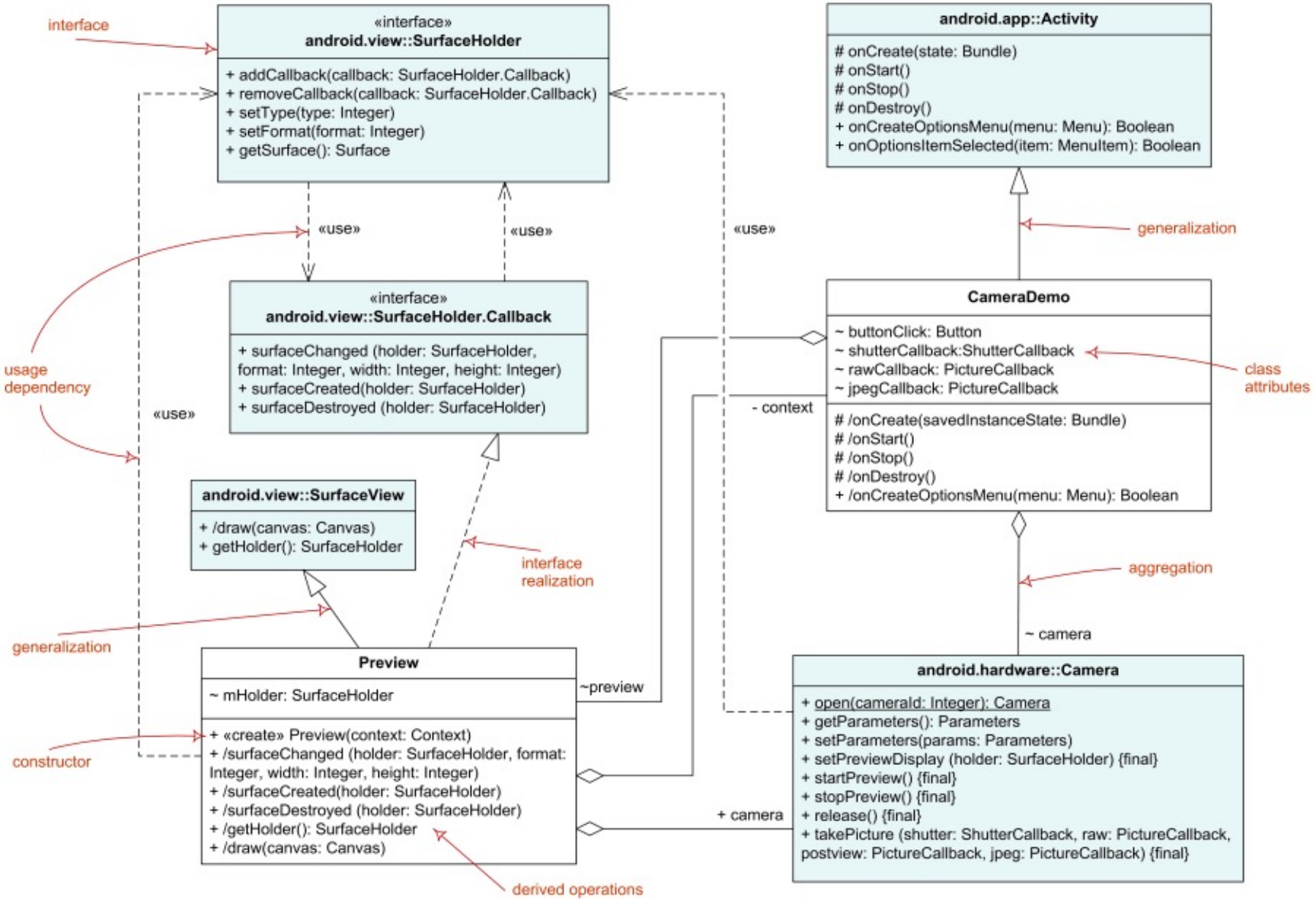
# Tổng quát hóa bằng thừa kế



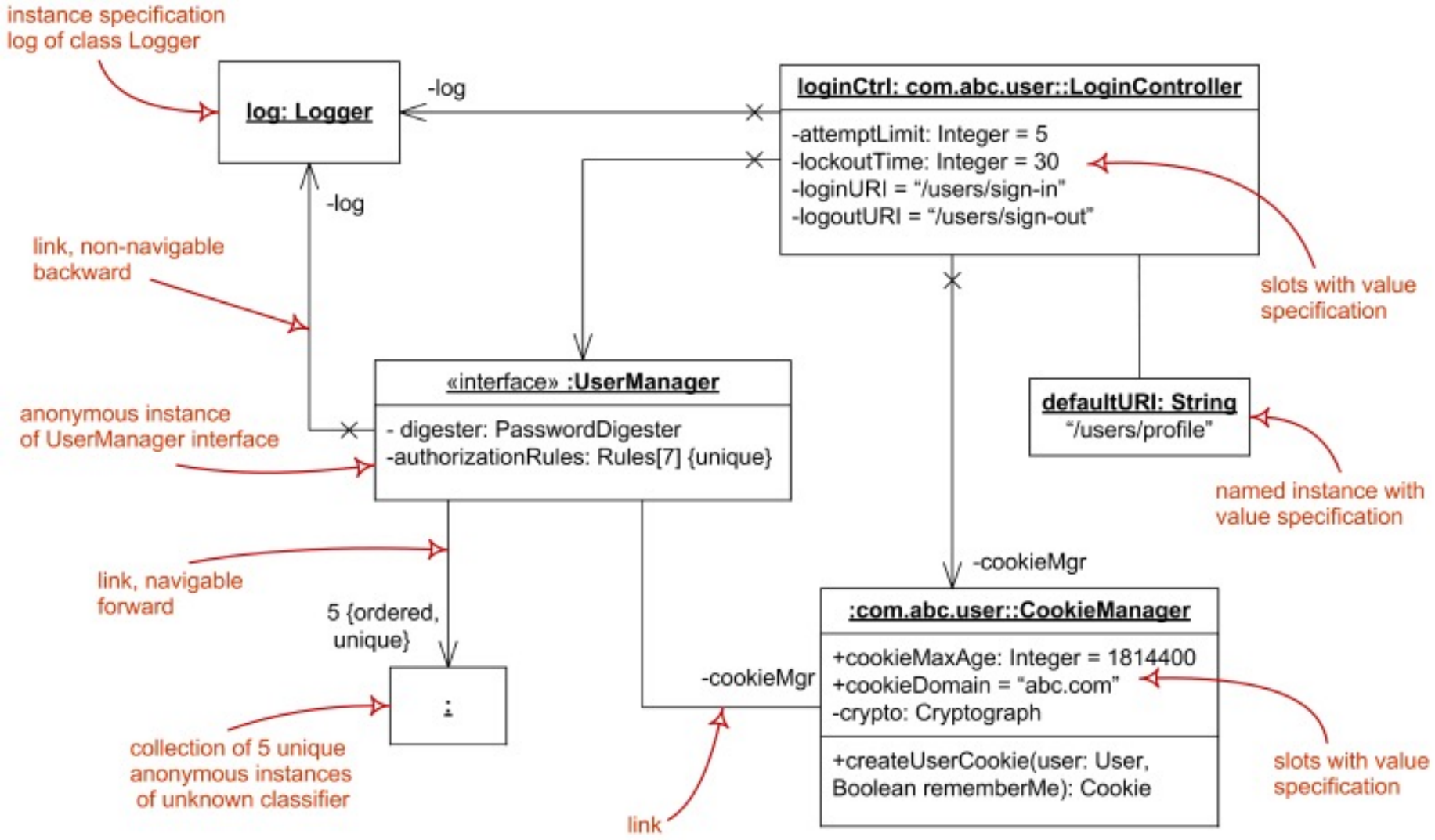
# Domain Model Diagram



# Diagram of Implementation Classes



# Object Diagram



# Phát hiện các lớp tham gia ca sử dụng

# Phát hiện các lớp tham gia ca sử dụng

- Mục đích của việc phát hiện các lớp tham gia ca sử dụng
- Quy trình (các bước) giúp phát hiện các đối tượng/lớp tham gia ca sử dụng
- Lập biểu đồ lớp cho ca sử dụng

# Mục đích của phát hiện các lớp tham gia ca sử dụng

- Trong bước mô hình hoá lĩnh vực (bước trước vừa thực hiện), ta đã tiến hành nghiên cứu lĩnh vực, mà không xem xét tới ứng dụng. Các lớp phát hiện được chỉ là các lớp lĩnh vực.
  - Như nhau đối với mọi hệ thống phần mềm (ứng dụng) thuộc lĩnh vực đó!
- Đặc thù của mỗi ứng dụng nằm ở các ca sử dụng. Vậy để nghiên cứu ứng dụng, ta phải đi sâu phân tích cấu trúc và hành vi của các ca sử dụng.
- Ca sử dụng được xem như là một hợp tác của một số đối tượng, bao gồm các lớp lĩnh vực và các lớp riêng (cụ thể) của ứng dụng
- Mục đích của bước này chính là phân tích cấu trúc tĩnh của các ca sử dụng. Ta sẽ phải lần lượt thực hiện các việc sau:
  - 1. Phát hiện các lớp tham gia ca sử dụng
  - 2. Thêm các mối liên quan giữa các lớp để lập một biểu đồ lớp cho mỗi ca sử dụng

# Phát hiện các đối tượng/lớp tham gia ca sử dụng (1)

- Các ca sử dụng được nghiên cứu (được phân tích) để phát hiện các lớp/đối tượng tham gia từng ca sử dụng đó
- Các lớp tham gia ca sử dụng được gọi chung là các lớp phân tích, gồm 3 loại:
  - Lớp biên
  - Lớp điều khiển
  - Lớp thực thể

# Phát hiện các đối tượng/lớp tham gia ca sử dụng (2)

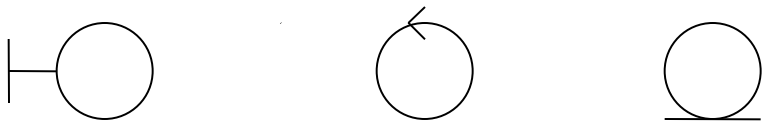
- Các lớp biên (boundary), còn được gọi là lớp đối thoại (dialog):
  - Đó là các lớp nhằm chuyển đổi thông tin giao tiếp (trao đổi) giữa các tác nhân và hệ thống:
    - Điển hình là các màn hình giao tiếp với các người dùng, cho phép thu thập thông tin hay xuất (hiển thị) các kết quả
    - Đó cũng có thể là các giao diện (phần cứng, phần mềm) chuyển đổi tương tự/số giữa hệ thống và các thiết bị mà nó điều khiển hay thu thập thông tin
  - Đối với mỗi cặp (tác nhân, ca sử dụng), thì phải có ít nhất một lớp biên. Lớp biên chính này lại có thể cần đến các lớp biên phụ trợ để nó uỷ thác (giao lại) một phần nào đó trong các trách nhiệm quá lớn của nó
  - Thường thì các đối tượng biên có vòng đời (lifecycle) kéo dài cùng với ca sử dụng liên quan

# Phát hiện các đối tượng/lớp tham gia ca sử dụng (3)

- Các lớp điều khiển (control):
  - Đó là các lớp quản lý và kiểm soát sự diễn tiến trong một ca sử dụng; có thể nói đó là cái "động cơ" làm cho ca sử dụng tiến triển được
  - Các lớp điều khiển chứa các quy tắc nghiệp vụ và ở vị trí trung gian giữa các lớp biên với các lớp thực thể, cho phép từ màn hình (giao tiếp với người dùng) có thể thao tác được các thông tin chứa đựng trong các thực thể
  - Cứ mỗi ca sử dụng, ta cần lập ít nhất một lớp điều khiển
  - Các lớp điều khiển, khi chuyển đến giai đoạn thiết kế, thì không nhất thiết là sẽ còn tồn tại như một lớp thực sự, vì nhiệm vụ của nó có thể bị phân tán vào các lớp khác, song trong giai đoạn phân tích, thì nhất thiết phải có chúng để bảo đảm không bỏ sót các chức năng hay hành vi trong các ca sử dụng

# Phát hiện các đối tượng/lớp tham gia ca sử dụng (4)

- Các lớp thực thể (entity):
  - Đây là các lớp nghiệp vụ, được xác định trực tiếp từ mô tả lĩnh vực, và sẽ được khẳng định khi được xuất hiện trong các ca sử dụng
  - Các lớp thực thể là các lớp trường cữu (persistent class), nghĩa là các lớp mà các dữ liệu và các mối liên quan của chúng còn được lưu lại (thường là ở trong cơ sở dữ liệu hay trong các tập tin) sau khi ca sử dụng của chúng đã kết thúc
  - Lớp thực thể được chọn tham gia ca sử dụng khi thông tin chứa đựng trong lớp thực thể đó được đề cập trong ca sử dụng
- Biểu diễn: <<boundary>> <<control>> <<entity>>



# Lập biểu đồ lớp cho ca sử dụng (1)

- Mục đích và yêu cầu:
  - Mục đích cuối cùng của Bước 4 trong quy trình RUP (phát hiện các lớp tham gia các ca sử dụng) là lập một biểu đồ lớp cho mỗi ca sử dụng, để phản ánh cấu trúc tĩnh của sự hợp tác
  - Biểu đồ lớp tham gia ca sử dụng sẽ là cái nền để trên đó diễn ra các hoạt động tương tác giữa các lớp, mà ta sẽ đi sâu tìm hiểu trong bước sau

# Lập biểu đồ lớp cho ca sử dụng (2)

- Biểu đồ các lớp tham gia một ca sử dụng phải bao gồm đủ các lớp đã được phát hiện cùng các yếu tố cấu trúc (thuộc tính, thao tác, liên kết) cần thiết của mỗi lớp
  - Các thuộc tính phải lưu giữ đủ những thông tin về các đối tượng, cần dùng trong hợp tác
  - Các thao tác cung cấp các khả năng dịch vụ cần thiết của mỗi lớp khi tham gia vào hợp tác
  - Các liên kết tạo ra các cầu nối giữa các lớp, cho phép chúng biết nhau và trao đổi với nhau khi hợp tác

# Lập biểu đồ lớp cho ca sử dụng (3)

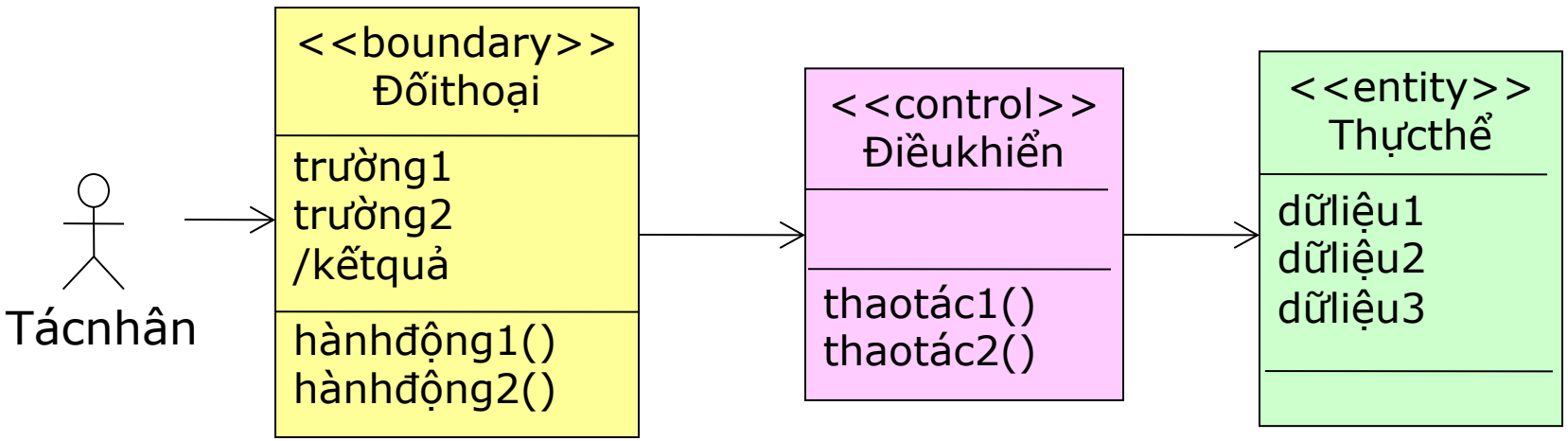
- Bổ sung các thuộc tính và thao tác:
  - Các lớp thực thể tạm thời chỉ có các thuộc tính. Các thuộc tính này diễn tả các thông tin trường cửu (persistent) của hệ thống
  - Các lớp điều khiển lại chỉ có các thao tác. Các thao tác này thể hiện các xử lý nghiệp vụ của ứng dụng, các quy tắc nghiệp vụ, các hành vi của hệ thống
  - Các lớp biên có cả thuộc tính và thao tác:
    - Các thuộc tính diễn tả các trường để thu thập thông tin hay để xuất kết quả. Các kết quả được phân biệt bằng ký hiệu thuộc tính dẫn xuất.
    - Các thao tác biểu diễn những hành động mà người dùng thực hiện trên màn hình giao diện.

# Lập biểu đồ lớp cho ca sử dụng (4)

- Bổ sung các liên kết:
  - Các lớp biên chỉ được nối với các lớp điều khiển hay với các lớp biên khác. Nói chung thì các liên kết là một chiều từ lớp biên đến lớp điều khiển, trừ khi lớp điều khiển lại tạo lập một đối thoại mới (chẳng hạn một trang hiển thị các kết quả).
  - Các lớp thực thể chỉ được nối với các lớp điều khiển hay lớp thực thể khác. Liên kết với các lớp điều khiển luôn luôn là một chiều (từ lớp điều khiển tới lớp thực thể).
  - Các lớp điều khiển được phép truy cập tới mọi loại lớp (kể cả các lớp điều khiển khác)

# Lập biểu đồ lớp cho ca sử dụng (5)

- Thêm các tác nhân:
  - Cuối cùng, ta thêm các tác nhân vào biểu đồ lớp
  - Một tác nhân chỉ được nối với một (hay một số) lớp biên



# Bài tập tổng hợp (1)

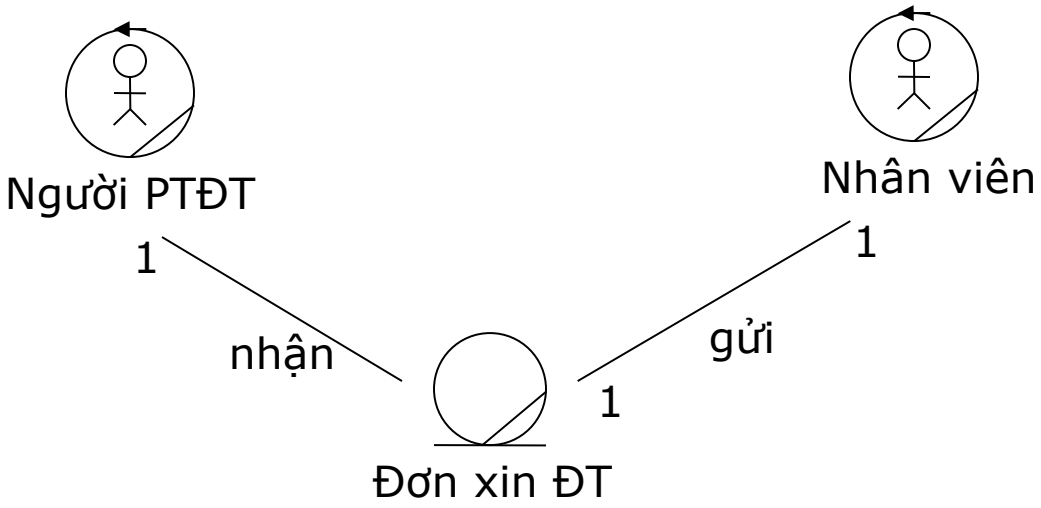
- Mô hình hóa cấu trúc tĩnh được thực hiện dựa vào văn bản phát biểu bài toán. Phát biểu đó được biên tập lại có lược bớt như sau:
  1) QTĐT bắt đầu khi người phụ trách đào tạo (PTĐT) nhận được một đề nghị đi đào tạo từ một nhân viên (NV).
  2) Người PTĐT xem xét đề nghị này và đưa ra trả lời đồng ý hay không đồng ý.
  3) Nếu đồng ý, người PTĐT tìm trong một danh sách các lớp đào tạo để chọn một lớp đào tạo phù hợp.
  4) Người PTĐT thông báo nội dung của lớp đào tạo cho NV đã xin đi đào tạo, cùng với một danh sách các kỳ học sẽ mở tới đây.
  5) Khi người NV đã chọn kỳ học, người PTĐT gửi một yêu cầu đăng ký cho NV đó tới cơ sở đào tạo.
  6) Người PTĐT kiểm tra lại hoá đơn mà cơ sở đào tạo gửi tới, trước khi chuyển cho kế toán trả tiền.

# Bài tập tổng hợp (2)

**Bước 8:** *MHH câu phát biểu thứ 1, sử dụng các biểu tượng của Jacobson*.

*"QTĐT bắt đầu khi người phụ trách đào tạo (PTĐT) nhận được một đề nghị đi đào tạo từ một nhân viên (NV)".*

- Chú ý các danh từ
- QTĐT đã được xác định từ Bước 1 (của bài 5 – Phân tích chức năng) là một quy trình nghiệp vụ, vậy không phải là lớp
- Còn lại các danh từ: "người PTĐT", "đề nghị ĐT", "nhân viên" sẽ được mô hình hóa thành các lớp

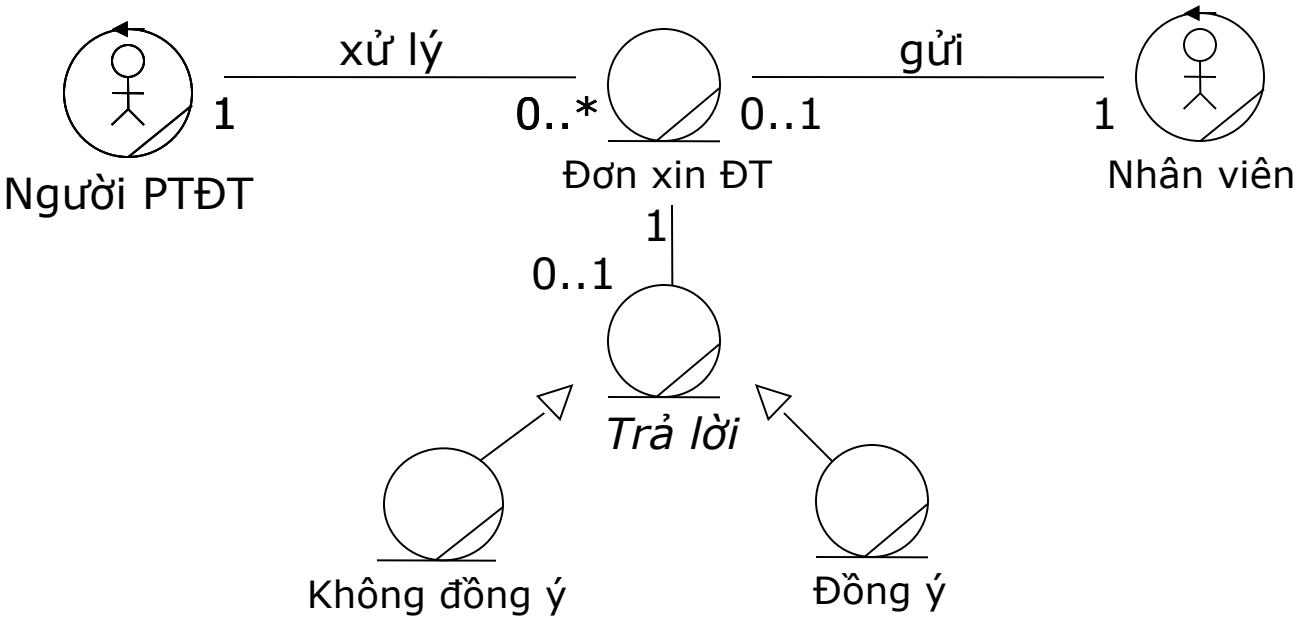


# Bài tập tổng hợp (3)

**Bước 9:** *MHH câu phát biểu thứ 2.*

*"Người PTĐT xem xét đề nghị này và đưa ra trả lời đồng ý hay không đồng ý".*

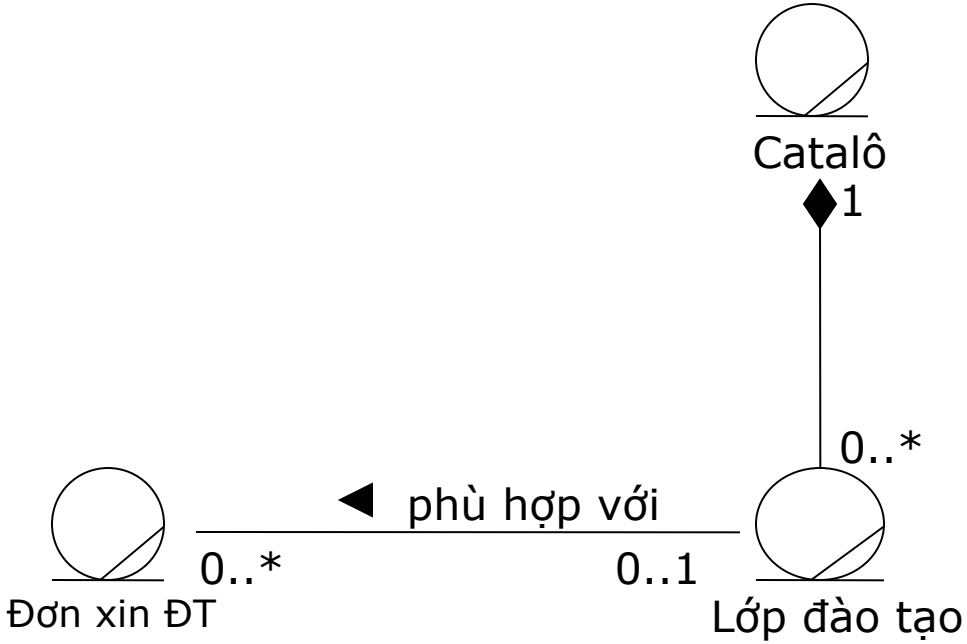
- Mở rộng mô hình trên, có chỉnh sửa vài chi tiết cho thích hợp hơn và thêm một thực thể trừu tượng (Trả lời) cùng với 2 thực thể chuyên biệt (Đồng ý, Không đồng ý)



# Bài tập tổng hợp (4)

**Bước 10:** *MHH câu phát biểu thứ 3.*

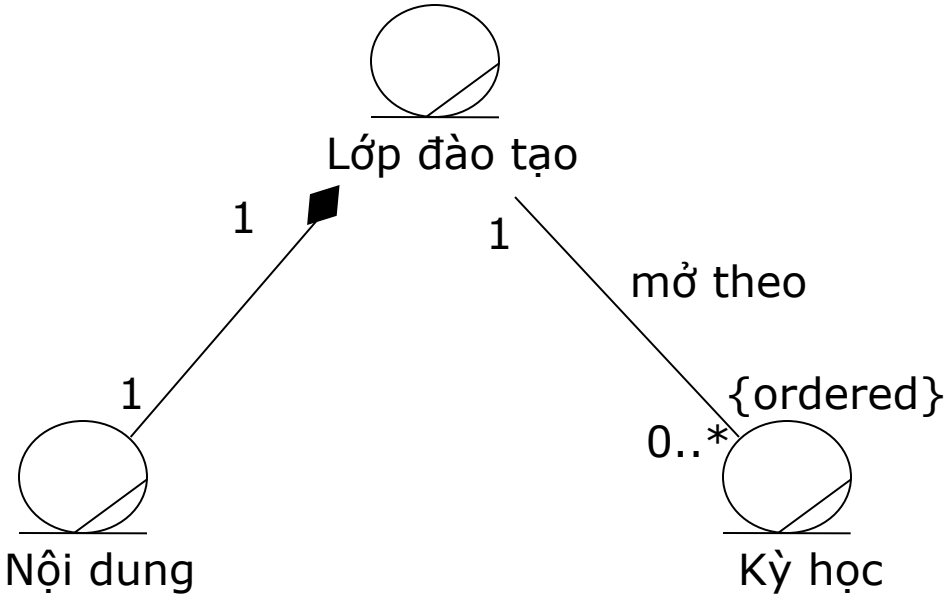
*"Nếu đồng ý, người PTĐT tìm trong một danh sách các cơ sở đào tạo (Catalô) một lớp đào tạo phù hợp".*



# Bài tập tổng hợp (5)

**Bước 11:** *MHH câu phát biểu thứ 4.*

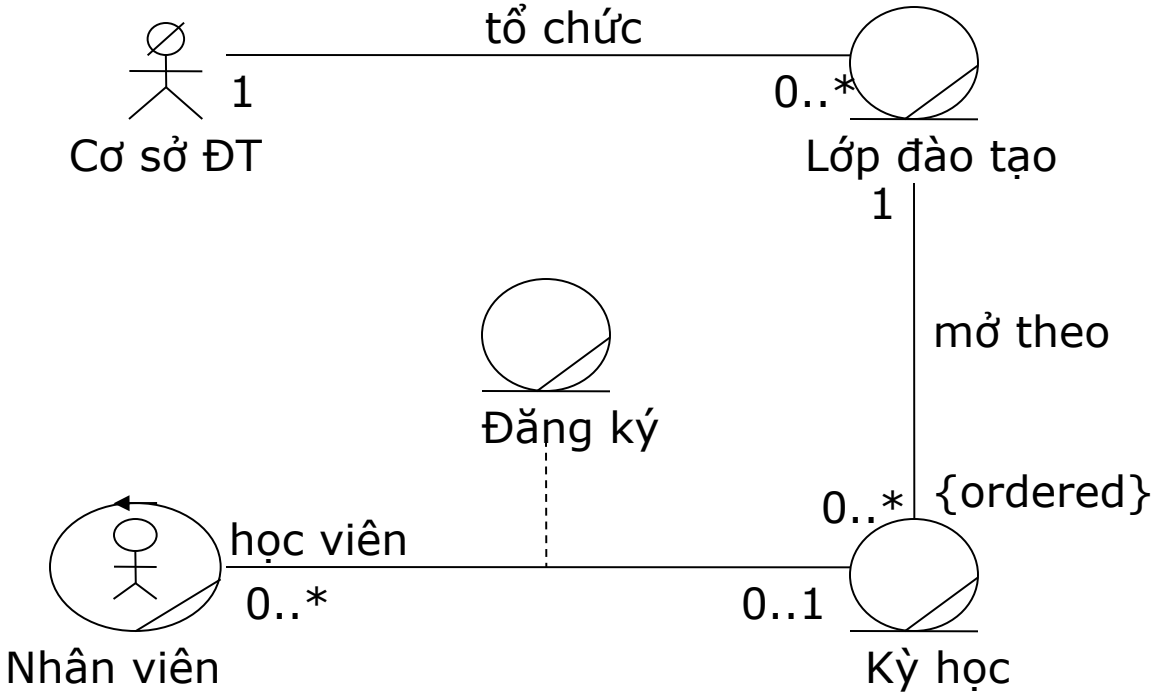
*"Người PTĐT thông báo nội dung của lớp đào tạo cho NV đã xin đi đào tạo, cùng với một danh sách các kỳ học sẽ mở tới đây".*



# Bài tập tổng hợp (6)

**Bước 12:** *MHH câu phát biểu thứ 5.*

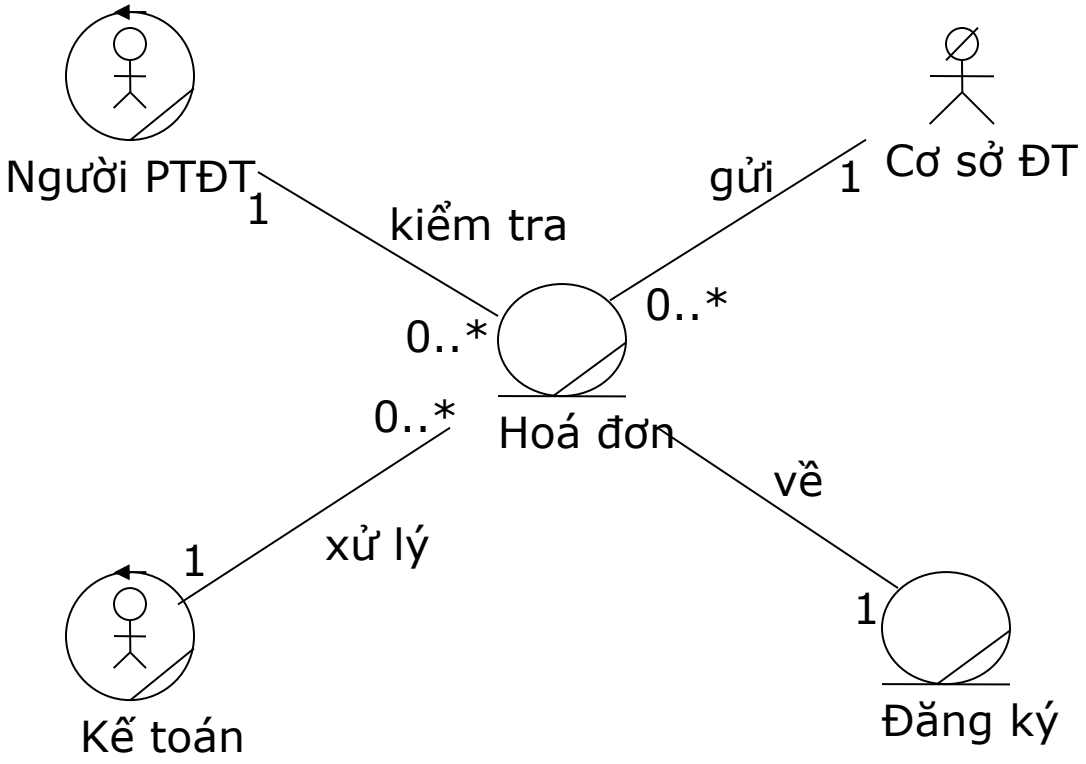
*"Khi người NV đã chọn kỳ học, người PTĐT gửi một yêu cầu đăng ký cho NV đó tới cơ sở đào tạo".*



# Bài tập tổng hợp (7)

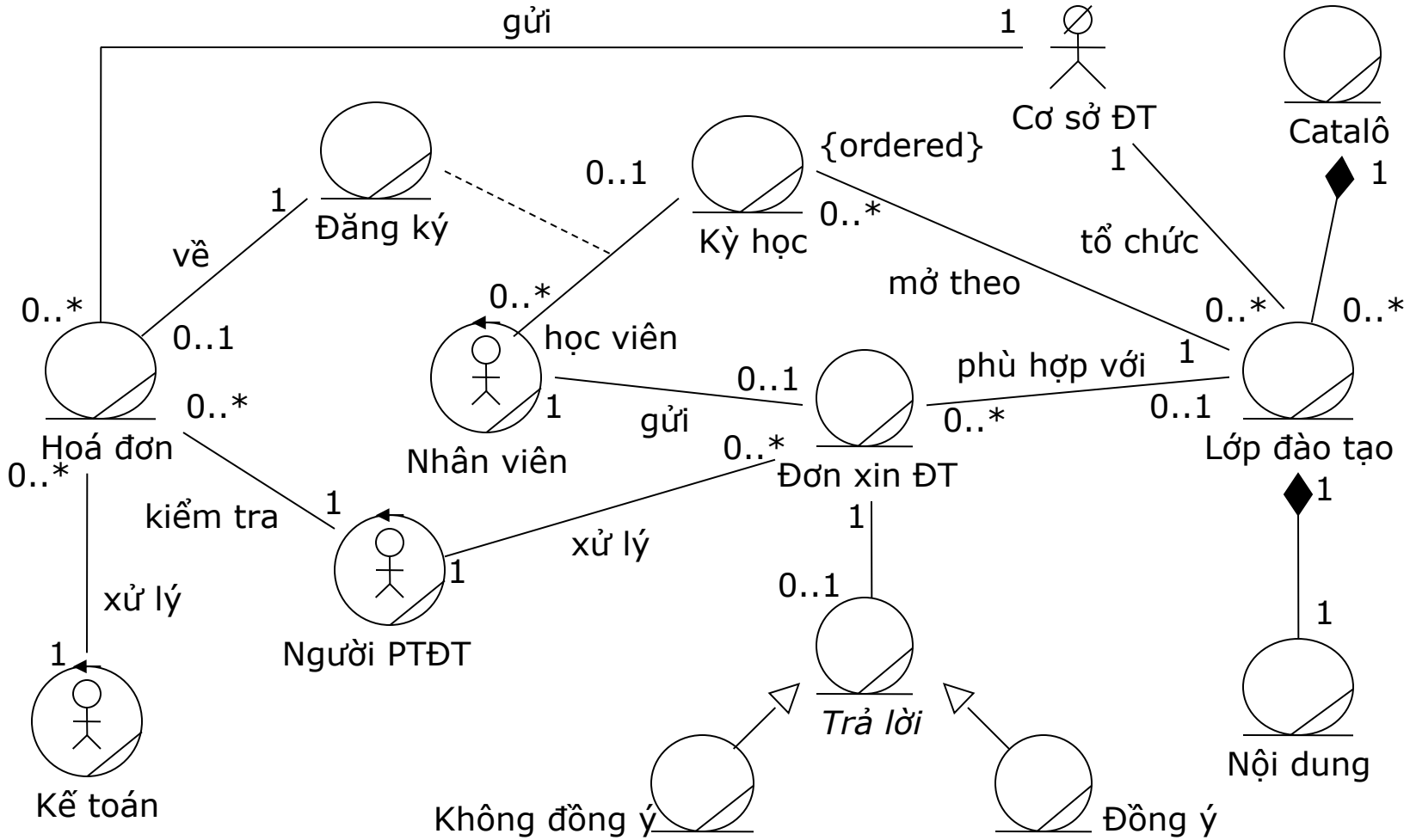
**Bước 13:** *MHH câu phát biểu thứ 6.*

*"Người PTĐT kiểm tra lại hoá đơn mà cơ sở đào tạo gửi tới, trước khi chuyển cho kế toán trả tiền".*



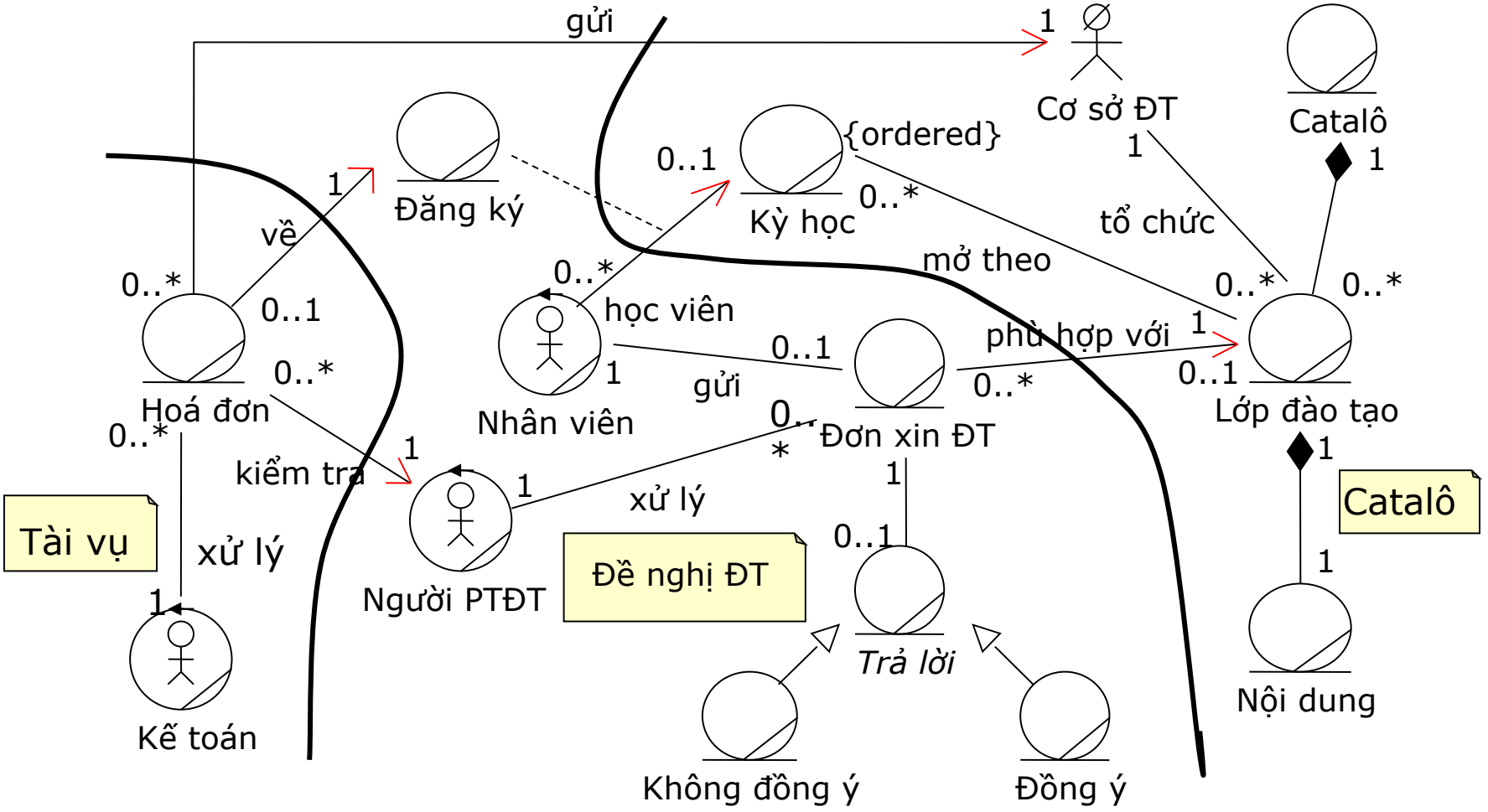
# Bài tập tổng hợp (8)

- **Bước 14:** *Tổng hợp các kết quả trên vào một biểu đồ lớp nghiệp vụ (lĩnh vực).*



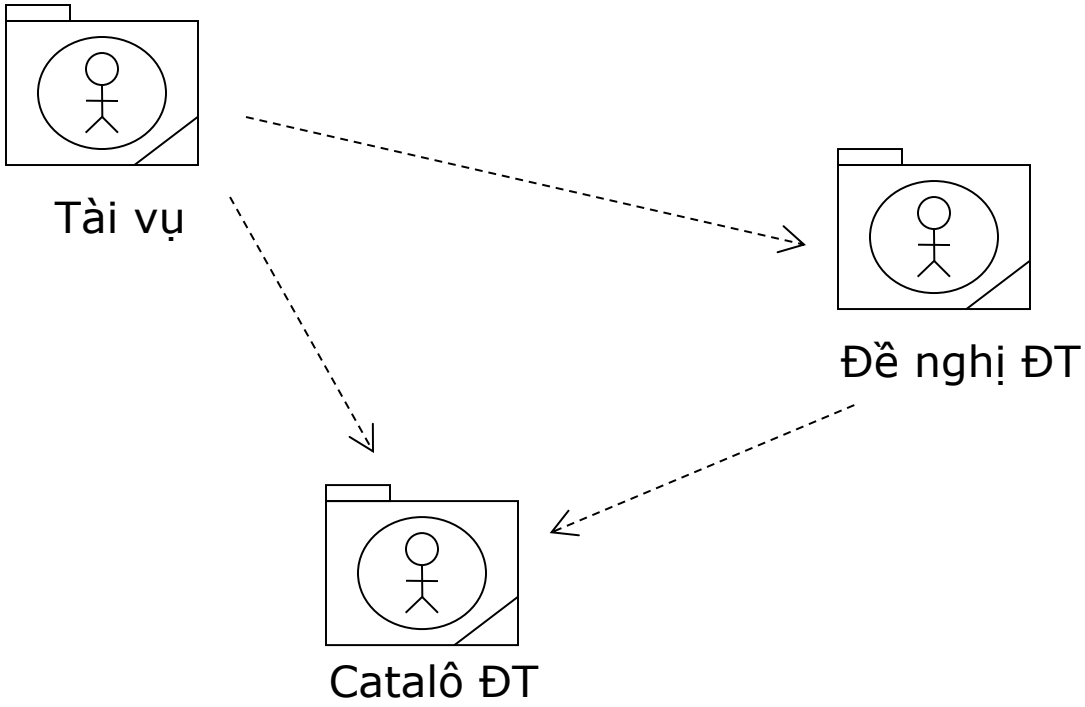
# Bài tập tổng hợp (9)

- **Bước 15:** *Chia cắt mô hình thành các gói theo các đơn vị nghiệp vụ và sửa lại các liên kết cho nhẹ bớt sự phụ thuộc giữa các gói.*



# Bài tập tổng hợp (10)

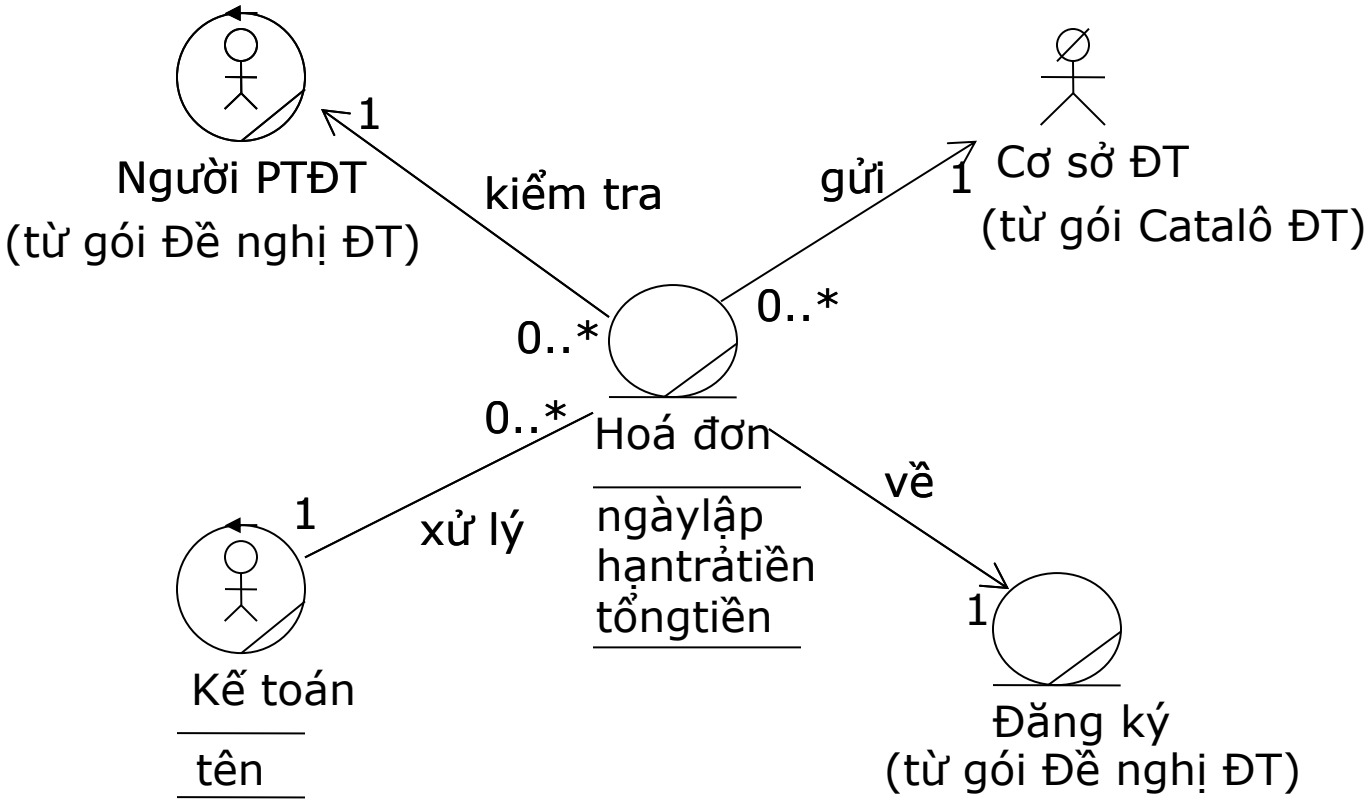
- Biểu đồ gói thu được như sau (các phụ thuộc là một chiều):



# Bài tập tổng hợp (11)

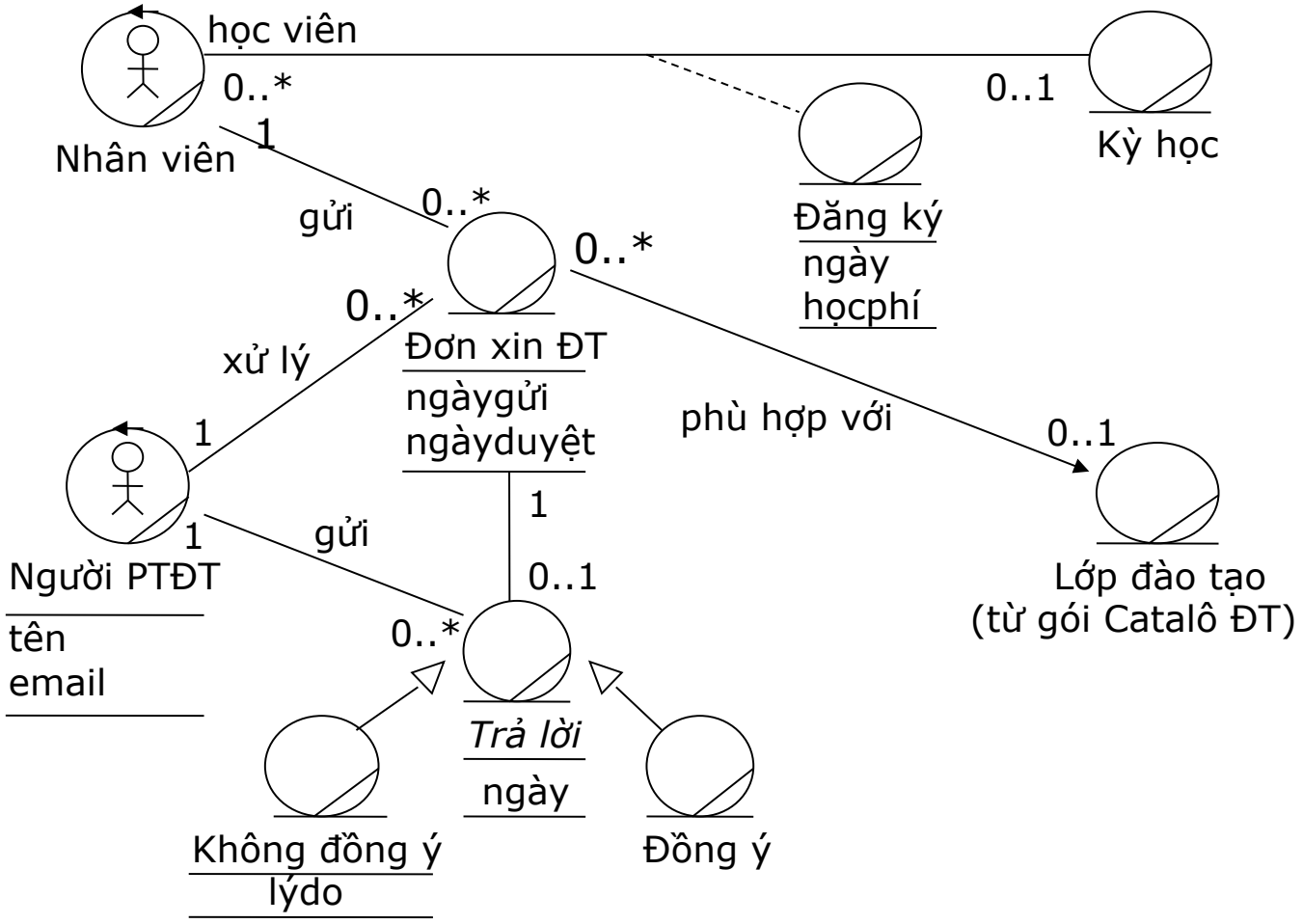
**Bước 16:** *Thêm các thuộc tính trường cữu (persistent) vào các lớp (vẽ riêng cho từng gói)*

- Biểu đồ lớp của gói "Kế toán":



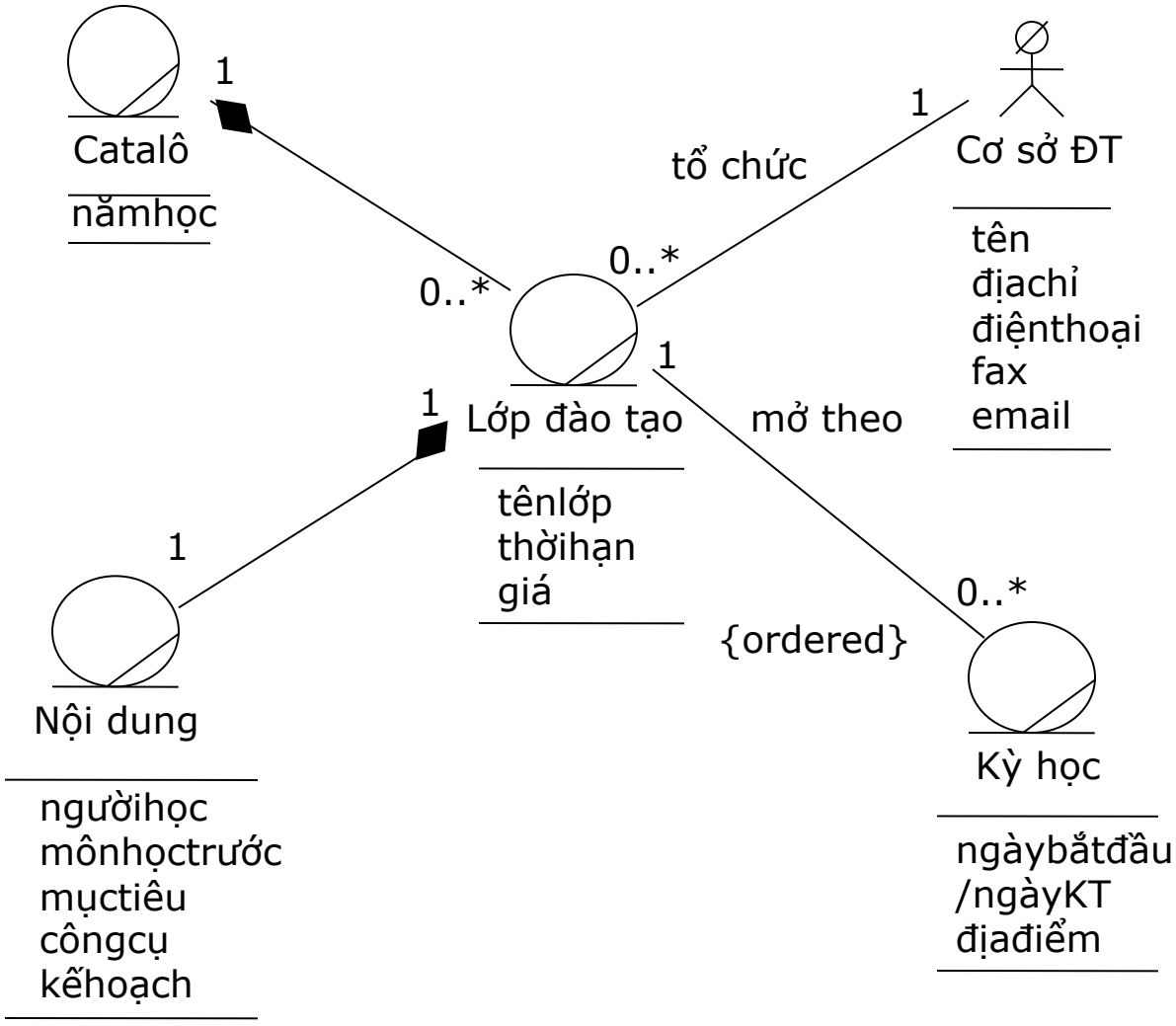
# Bài tập tổng hợp (12)

- Biểu đồ lớp của gói "Đề nghị ĐT":



# Bài tập tổng hợp (13)

- Biểu đồ lớp của gói "Catalô ĐT":



# Bài tập tổng hợp (14)

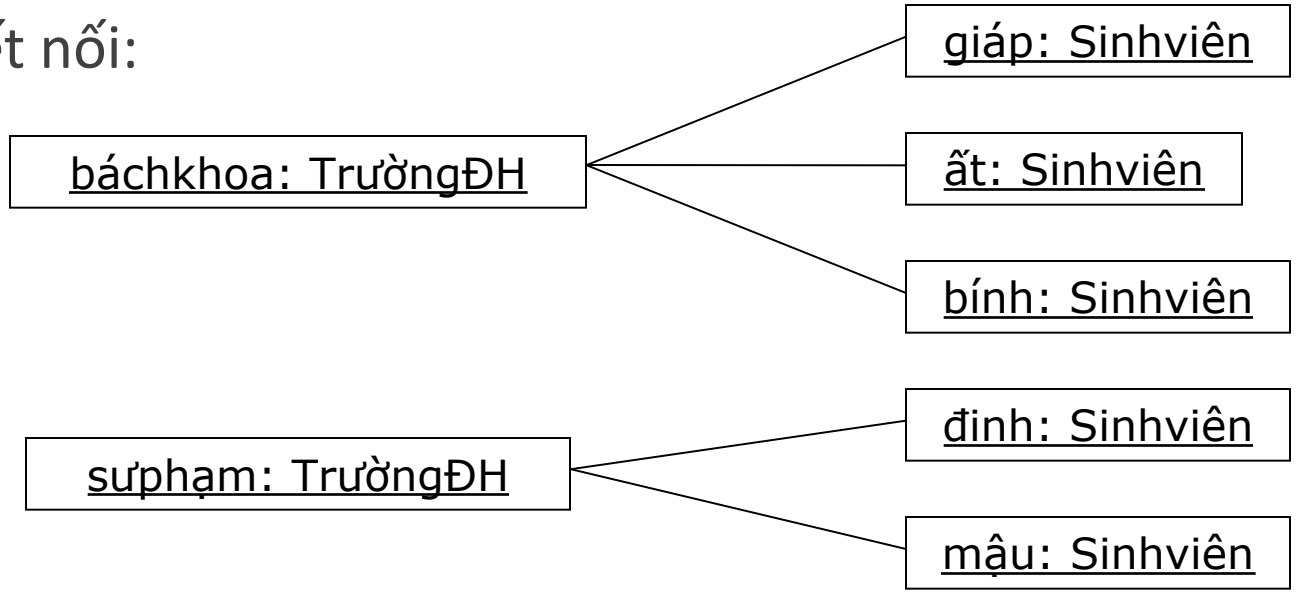
- Đến đây thì ta đã hoàn thành việc phát hiện các lớp lĩnh vực (lớp thực thể) của ứng dụng
- Việc phát hiện các lớp biên và các lớp điều khiển cho mỗi ca sử dụng là không khó.
  - Xem đó là một bài tập ở nhà!
- Việc phát hiện các lớp biên và các lớp điều khiển sẽ tiếp tục được cập nhật khi chúng ta tiến hành việc Phân tích hành vi (được trình bày trong bài học sau)

Thank you for your attention!

Q&A

# Mối liên quan liên kết (2)

- Biểu diễn kết nối:




- Biểu diễn liên kết: