

# Phân tích thiết kế Hệ thống

Giảng viên: Nguyễn Bá Ngọc

## Chương 3

Hà Nội-2021


# **Chương 3**

## **Mô hình hóa cấu trúc**

# Nội dung

- Các khái niệm
- Các kỹ thuật
- Các biểu đồ
- Mô tả ràng buộc với OCL
- Đặc tả lớp với thẻ CRC
- Mẫu phân tích

# Nội dung

- 
- Các khái niệm
  - Các kỹ thuật
  - Các biểu đồ
  - Mô tả ràng buộc với OCL
  - Đặc tả lớp với thẻ CRC
  - Mẫu phân tích

# Lĩnh vực vấn đề

- Lĩnh vực vấn đề (*Problem domain*) - Miền ứng dụng được quan tâm trong phạm vi hệ thống
  - Bao gồm tất cả các thông tin, các điều kiện ràng buộc, v.v..
  - .. cần thiết để giải quyết vấn đề.

# Lớp lĩnh vực

- Lớp lĩnh vực (*domain class*) mô tả các đối tượng của lĩnh vực vấn đề, các thứ mà người dùng tiếp xúc khi thực hiện các hoạt động nghiệp vụ
  - Cần được ghi nhớ và xử lý bởi hệ thống
  - Ví dụ, sản phẩm, hóa đơn, thông tin khách hàng, khoản thanh toán, v.v.
- (*Lớp lĩnh vực là 1 tầng kiến trúc hệ thống*)

*Làm cách nào để xác định các lớp lĩnh vực?*

# Nội dung

- Các khái niệm
- Các kỹ thuật
- Các biểu đồ
- Mô tả ràng buộc với OCL
- Đặc tả lớp với thẻ CRC
- Mẫu phân tích

# Mô tả bài toán

- Xác định các thành phần dữ liệu cần lưu trữ
- Kết hợp các thành phần và xác định các đối tượng
- Xác định các mối quan hệ giữa các đối tượng
- Xác định các lớp lĩnh vực
- Xây dựng mô hình cấu trúc



# Các kỹ thuật

- Kỹ thuật danh từ (Noun): Xác định tất cả các danh từ có trong tất cả các nguồn, sau đó phân tích và xác định các lớp lĩnh vực, các thuộc tính, loại bỏ những thứ không cần ghi nhớ/chỉ giữ lại những thứ cần được ghi nhớ.
- Tổng hợp kiến thức (BrainStorming): Tất cả mọi người cùng tham gia đóng góp, cùng biên soạn danh sách các mục dữ liệu quan trọng.
  - Thông tin có thể được tổng hợp và phân loại theo danh mục và được sử dụng làm cơ sở xác định các lớp lĩnh vực.

# Kỹ thuật danh từ

- Tìm, phân loại, và thiết lập danh sách các danh từ có trong các thảo luận hoặc tài liệu.
- Thường dẫn đến 1 danh sách dài với nhiều danh từ, bao gồm cả những thứ không cần lưu trong hệ thống.
  - Danh từ có thể là tên của thuộc tính, đối tượng, lớp và cả những thứ khác
- Có thể đem lại kết quả ban đầu đáng kể, kết quả thu được có thể tiếp tục được mở rộng, ví dụ dựa trên kinh nghiệm người dùng.

# Các bước áp dụng kỹ thuật danh từ

1. Sử dụng các mô hình chức năng (đặc tả ca sử dụng, biểu đồ ca sử dụng v.v.. ), và các tài liệu khác về hệ thống (bao gồm các thông tin nhập và xuất), xác định tất cả các danh từ và lập danh mục danh từ.
2. Sử dụng các nguồn thông tin khác hiện có: Các biểu mẫu, các báo cáo hiện có v.v., để mở rộng danh mục.
3. Kiểm tra các danh từ đã tìm được.
4. Kiểm tra danh mục và ghi chú phân loại từng danh từ cần được thêm vào, có thể giữ, loại bỏ hoặc cân nhắc thêm v.v..
5. Cùng kiểm tra danh sách với người dùng, các bên liên quan, và các thành viên thực hiện dự án và sau đó thiết lập danh sách các thứ trong lĩnh vực nghiệp vụ.

# Một số câu hỏi hỗ trợ phân loại danh từ

*Các câu hỏi có thể được sử dụng để thực hiện bước 3*

- Để quyết định giữ 1 danh từ
  - Nó có nằm trong phạm vi của hệ thống đang được phát triển?
  - Hệ thống có cần ghi nhớ nhiều hơn 1 thứ tương tự?
- Để quyết định loại 1 danh từ
  - Nó có phải 1 từ đồng nghĩa với 1 thứ khác đã được xác định?
  - Chỉ là đầu ra do hệ thống cung cấp từ những thông tin khác đã được xác định?
  - Chỉ là đầu vào dẫn tới việc lưu thông tin khác đã được xác định?
- Để xác định thuộc tính và đối tượng
  - Nó có phải thành phần của thứ khác đã được xác định?
  - Nó có phải thứ người dùng có thể cần đến?

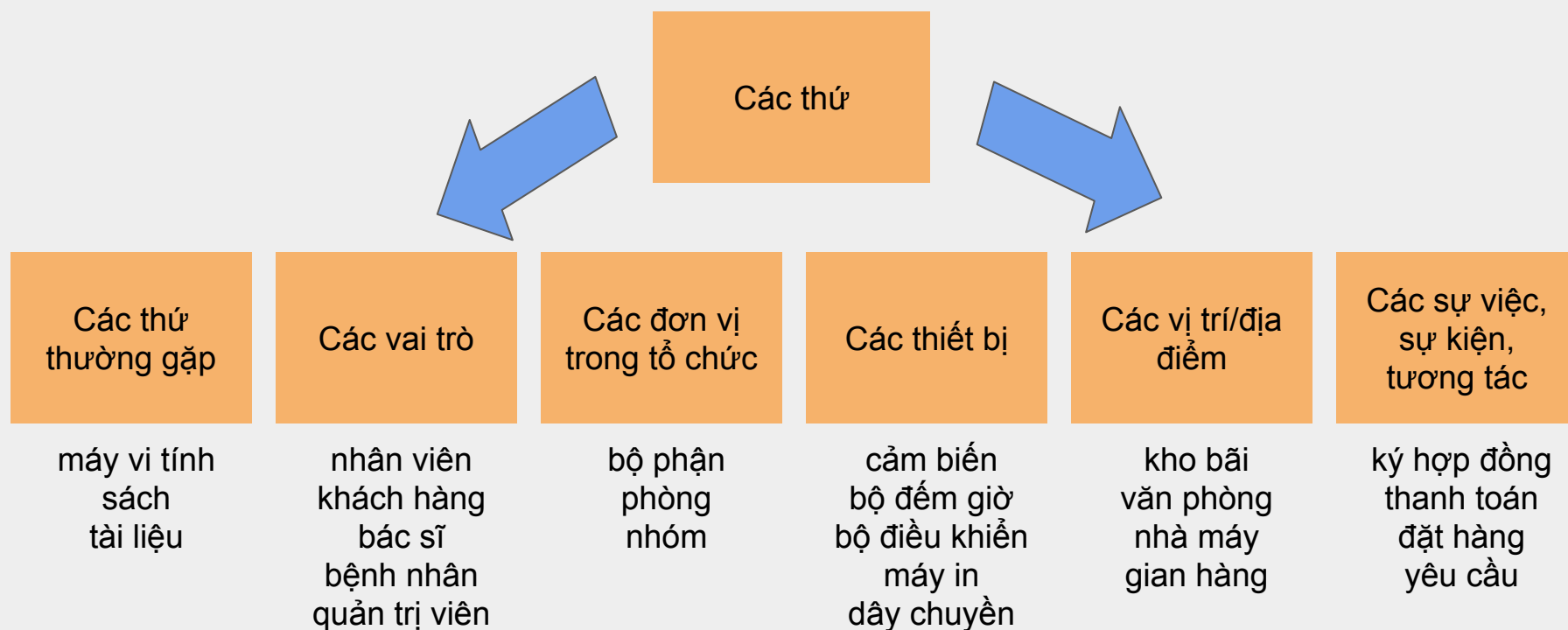
## Ví dụ 3.1. Các danh từ

Các danh từ đã được phát hiện	Ghi chú
Accounting	
Back order	
Bank	
Catalog	
Catalog details	
Change request	
Color	
Confirmation	
Credit card information	
Customer account	
Management	
Marketing	
Fulfillment reports	

# Các bước tổng hợp kiến thức

1. Lựa chọn người dùng và 1 tập ca sử dụng liên quan
2. Phỏng vấn người dùng và xác định các thứ liên quan đến ca sử dụng và thông tin về chúng cần được quản lý bởi hệ thống.
3. Sử dụng các danh mục để quản lý và đưa ra các câu hỏi: Bạn có lưu thông tin về những thứ quan trọng nào? Các hoạt động có được gắn với địa điểm nào không? Vai trò nào của người dùng cần được ghi nhớ?
4. Tiếp tục tổng hợp kiến thức từ tất cả các nhóm người dùng và những người liên quan để mở rộng kết quả.
5. Hợp nhất các kết quả, loại trùng lặp, và lập danh sách các thứ.

## Ví dụ 3.2. Danh mục phân loại

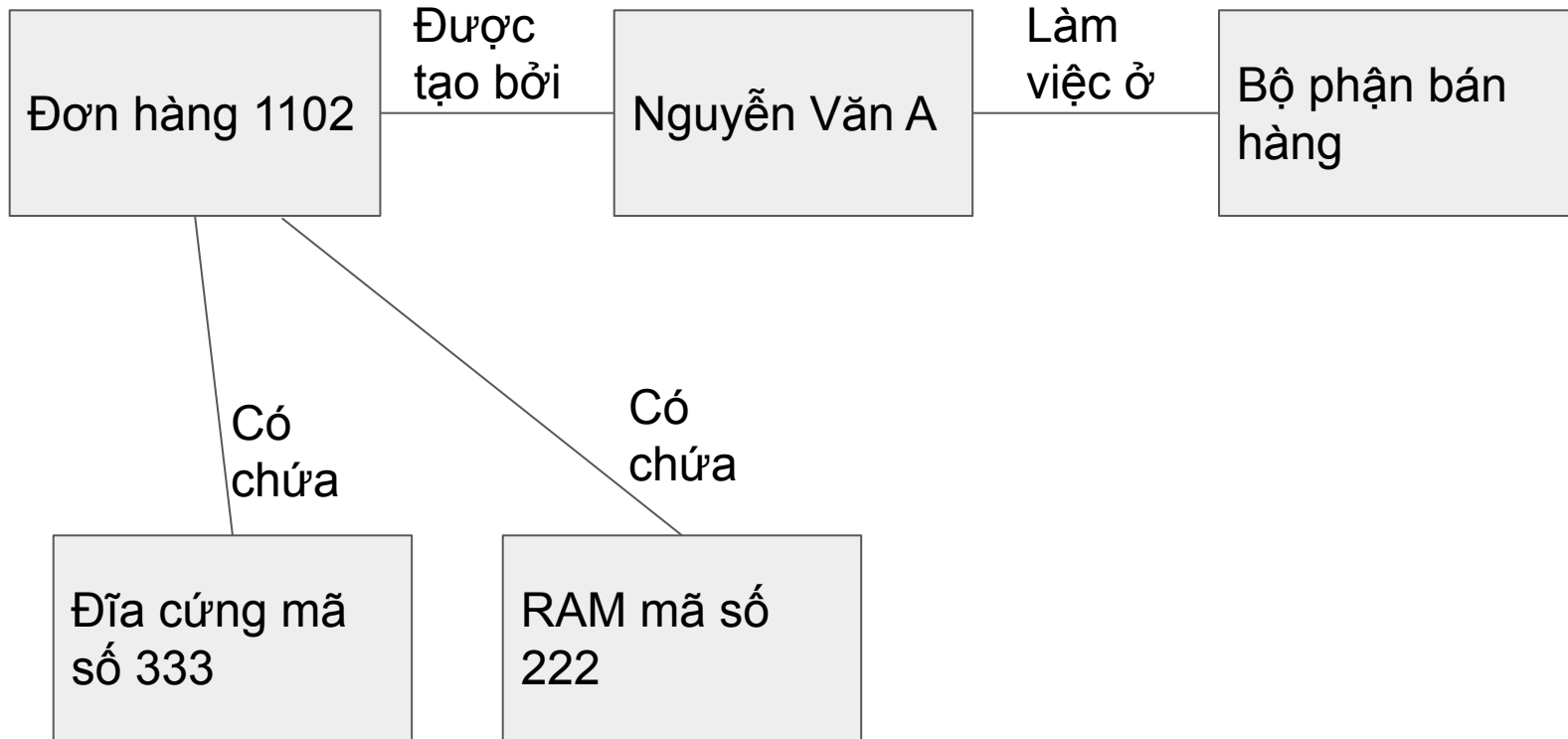


## Ví dụ 3.3. Các thuộc tính và giá trị

Tất cả khách hàng đều có các thuộc tính	Giá trị của các thuộc tính		
ID	101	102	103
Họ-đệm	Nguyễn Văn	Trần Thị	Vũ Văn
Tên	A	C	D
SĐT	0988888888	0967777777	097333333



## Ví dụ 3.4. Các mối liên hệ



# Nội dung

- Các khái niệm
- Các kỹ thuật
- Các biểu đồ
- Mô tả ràng buộc với OCL
- Đặc tả lớp với thẻ CRC
- Mẫu phân tích

# Biểu đồ lớp: Các thành phần thường gặp

## Lớp

Các thành phần của lớp: Tên, thuộc tính, phương thức

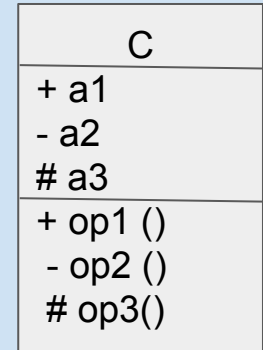


(Có thể lược bỏ thuộc tính và/hoặc phương thức nếu rỗng)

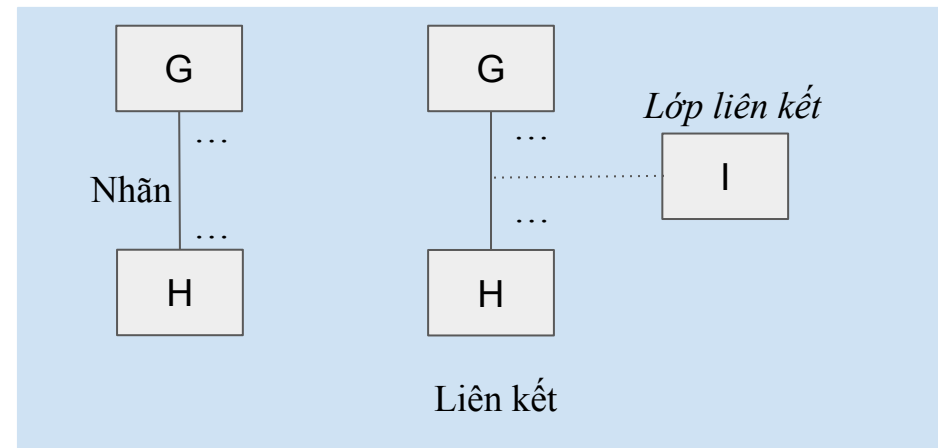
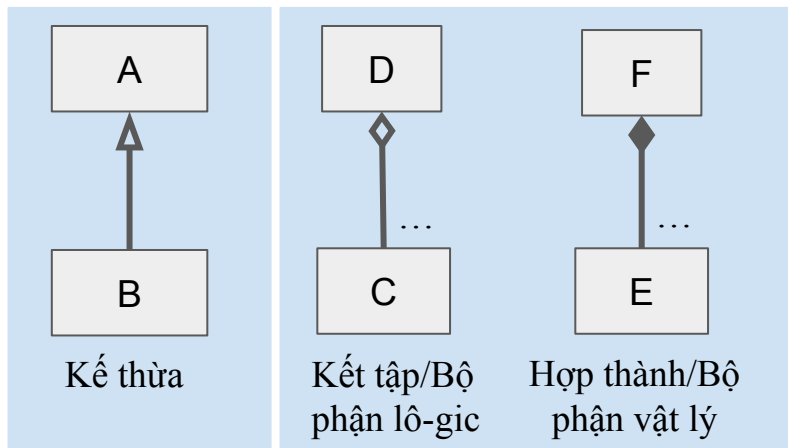
Các giới hạn truy cập:

+: Công khai  
- : Riêng tư  
#: Bảo vệ

*Ví dụ lớp*



## Quan hệ

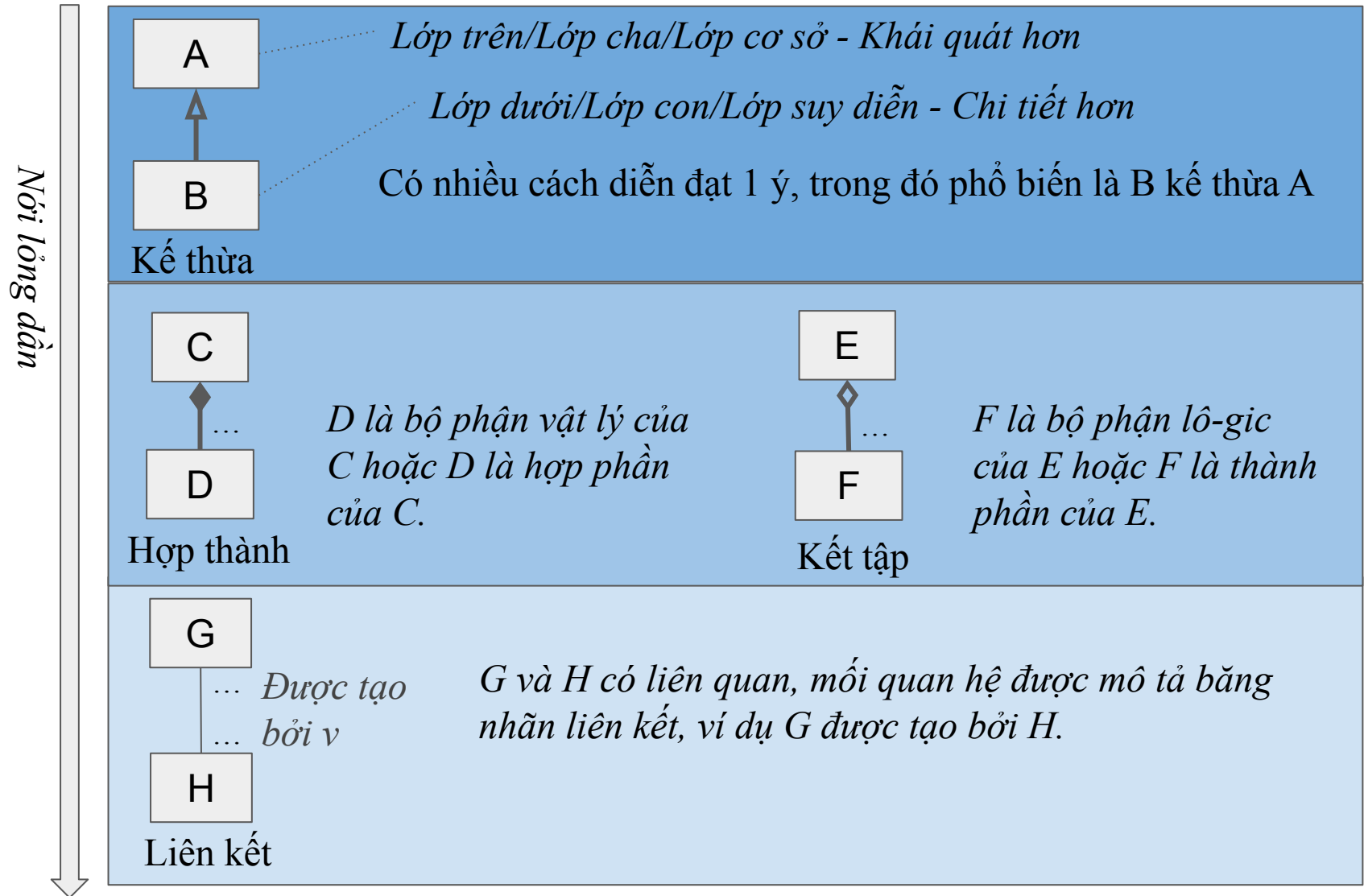


(ở vị trí ... là cơ số - mô tả số lượng đối tượng thuộc lớp ở mỗi đầu của mỗi quan hệ)

# Phân loại thuộc tính

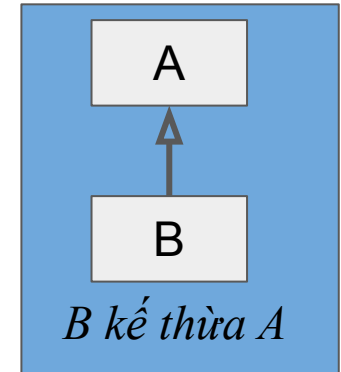
- Khóa
  - Một (bộ) giá trị của (nhóm) thuộc tính khóa xác định duy nhất 1 đối tượng.
  - Ví dụ mỗi mã khách hàng xác định đúng 1 khách hàng.
- Thuộc tính đơn trị
  - Giá trị của thuộc tính không được chia nhỏ thành các thành phần dữ liệu nhỏ hơn nữa.
- Thuộc tính đa trị
  - Bao gồm nhiều thành phần dữ liệu nhỏ hơn.
  - Ví dụ địa chỉ có thể bao gồm số nhà, đường phố, quận v.v..
- Thuộc tính suy diễn
  - Có thể tính được từ những giá trị của các thuộc tính khác.
  - Ví dụ tuổi của khách hàng có thể được tính từ ngày sinh.

# Các mối quan hệ



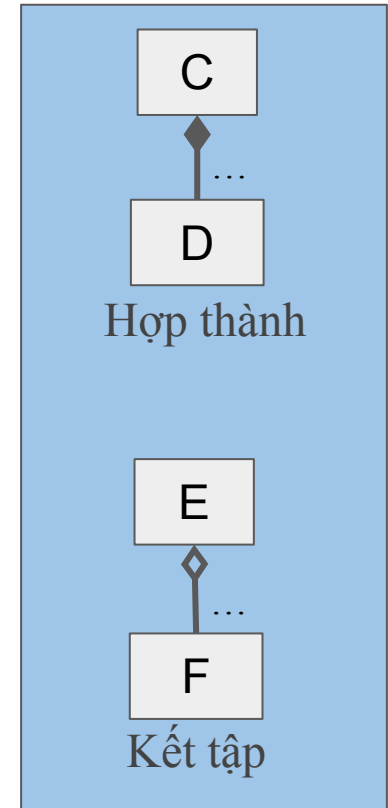
# Quan hệ kế thừa

- Hướng mũi tên đi từ lớp dưới đến lớp trên.
- Cho phép tái sử dụng lớp trên, bổ xung thành phần mới và có thể định nghĩa lại phương thức.
  - Trong các triển khai, B thường sở hữu 1 đối tượng kiểu A.
  - B có các giao diện như A, nhưng hành vi có thể không tương thích (do có thể định nghĩa lại, dù không mong đợi).



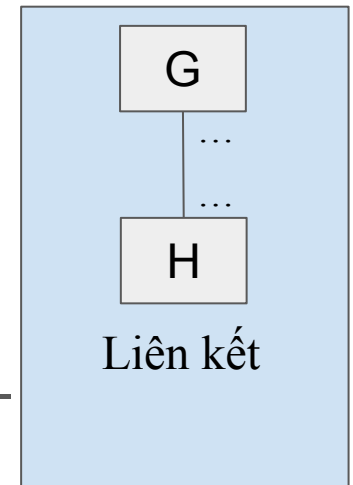
# Các quan hệ bộ phận-tổng thể

- Mũi tên đi từ bộ phận tới tổng thể.
- Gắn kết các bộ phận với cái tổng thể.
- Được chia thành 2 mức
  - Hợp thành - chặt hơn:
    - Tổng thể sở hữu bộ phận
    - Đối tượng bộ phận bị hủy khi hủy đối tượng tổng thể.
    - Mỗi bộ phận chỉ có thể là hợp phần của 1 tổng thể.
    - Diễn đạt: D là hợp phần của C, hoặc D là bộ phận vật lý của C.
  - Kết tập - lỏng hơn:
    - Tổng thể không sở hữu bộ phận.
    - Bộ phận vẫn có thể tồn tại sau khi hủy cái tổng thể.
    - Mỗi bộ phận có thể là thành phần của nhiều tổng thể - chia sẻ bộ phận.
    - Diễn đạt: F là thành phần của E, hoặc F là bộ phận lô-gic của E.



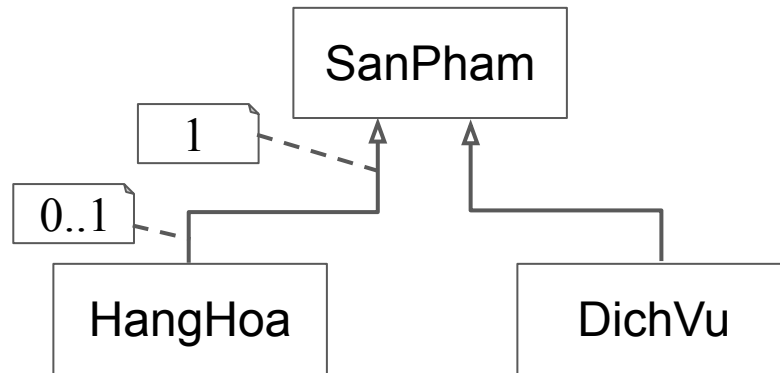
# Quan hệ liên kết

- Loại quan hệ lỏng nhất, ý nghĩa phụ thuộc vào trường hợp cụ thể.
- Được biểu diễn bằng đường nét liên kết nối 2 lớp cùng với các mô tả:
  - Tên quan hệ - Có thể đọc khác nhau theo mỗi chiều - Có thể bổ xung ký tự ghi chú chiều đọc.
  - Các vai trò của các đối tượng ở mỗi đầu.
  - Các tô điểm ở mỗi đầu liên kết.
  - Các cơ sở - Tự do.



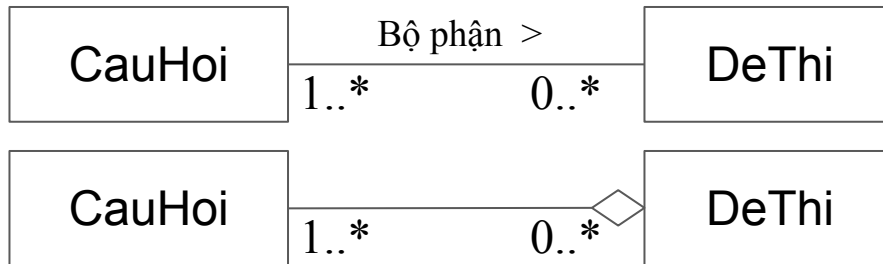


## Ví dụ 3.5. Quan hệ kế thừa

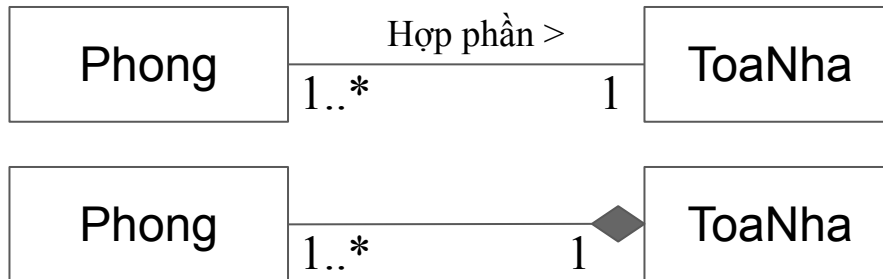


*Hàng hóa và Dịch Vụ là các loại Sản Phẩm / HangHoa và DichVu kế thừa SanPham*

## Ví dụ 3.6. Quan hệ bộ phận-tổng thể



*Mỗi câu hỏi có thể là thành phần của 1 hoặc nhiều đề thi, và cũng có thể không thuộc đề thi nào.*



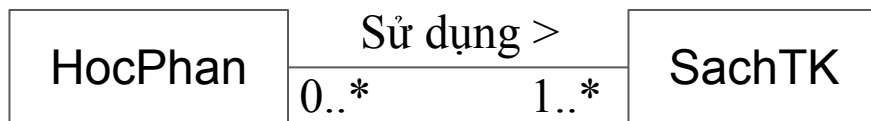
*Mỗi phòng phải thuộc đúng 1 tòa nhà.*

*Các đầu mũi tên kim cương (cú pháp hoa mỹ) được ưa chuộng hơn các nhãn để biểu diễn quan hệ bộ phận-tổng thể.*

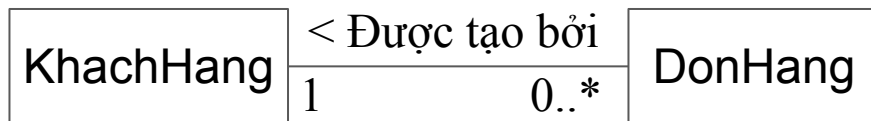
## Ví dụ 3.7. Mối quan hệ liên kết



*Mỗi trưởng phòng quản lý đúng 1 phòng.*

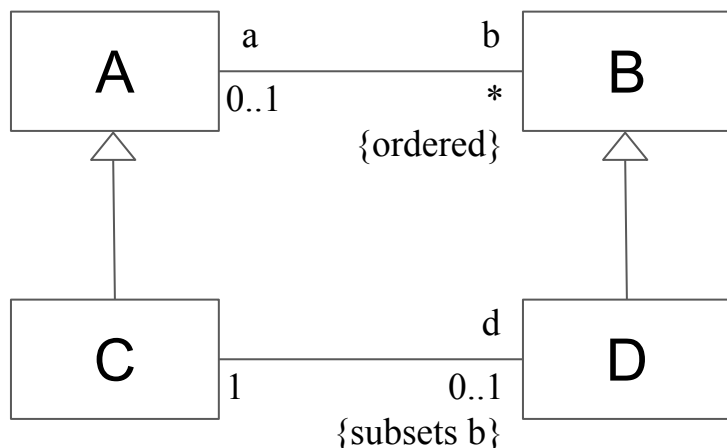


*Mỗi học phần sử dụng ít nhất 1 sách tham khảo.*



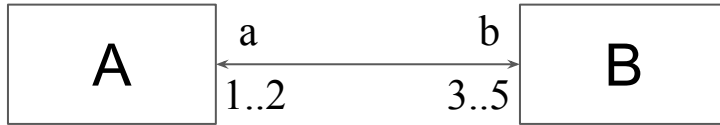
*Mỗi đơn hàng được tạo bởi đúng 1 khách hàng.*

## Ví dụ 3.8. Các tô điểm cho đầu liên kết



- a, b, d là các tên của các đầu liên kết.
- Các cơ số 0..1 trên a, \* trên b, 1 ở đầu lớp C (không đặt tên), và 0..1 trên d.
- Đầu b là 1 tập có thứ tự.
- Đầu d là tập con của đầu b, tương đương với:  
**context C inv: b->includesAll(d)**

## Ví dụ 3.9. Khả năng truy cập



*Liên kết có thể truy cập cả 2 đầu*



*Liên kết không truy cập được đầu nào*



*Liên kết không mô tả truy cập*



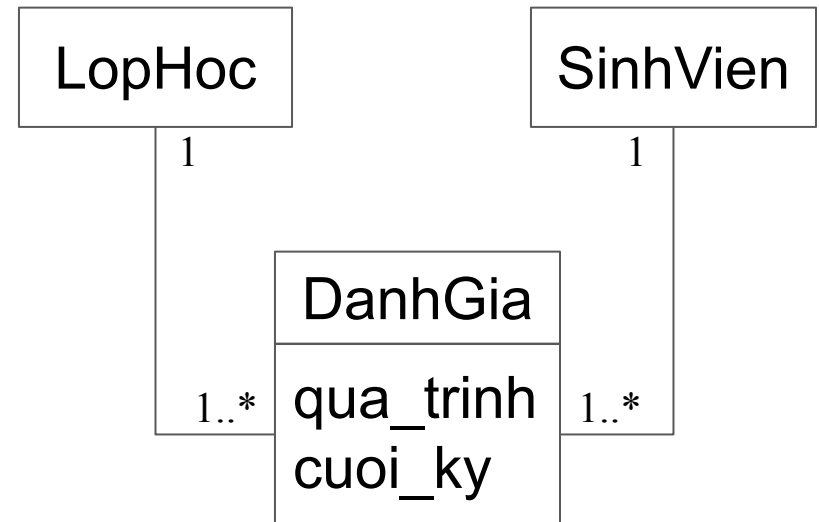
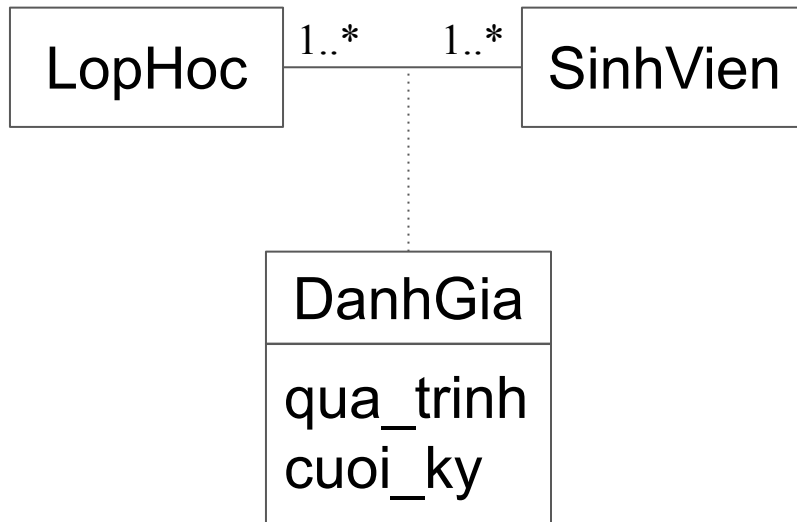
*Liên kết có thể truy cập đầu h nhưng không truy cập được đầu g*



*Liên kết có thể truy cập đầu j, chưa xác định đầu i.*

## Ví dụ 3.10. Lớp liên kết

*Biểu diễn các thuộc tính không phải thành phần của 2 đầu liên kết và đồng thời chỉ có nghĩa trong liên kết.*



# Các phương thức

- Biểu diễn hành vi/trách nhiệm của đối tượng, những việc mà đối tượng có thể/cần thực hiện.
- Để giản lược biểu đồ các phương thức có vai trò hỗ trợ thường không được biểu diễn
  - Tạo hoặc hủy đối tượng
    - Hàm tạo
    - Hàm Hủy
  - Đọc hoặc thiết lập giá trị (get/set)
    - Lấy giá trị của thuộc tính
    - Thiết lập giá trị thuộc tính

# Giới hạn truy cập

*Thường được áp dụng để hỗ trợ đảm bảo tính nhất quán của các biểu diễn*

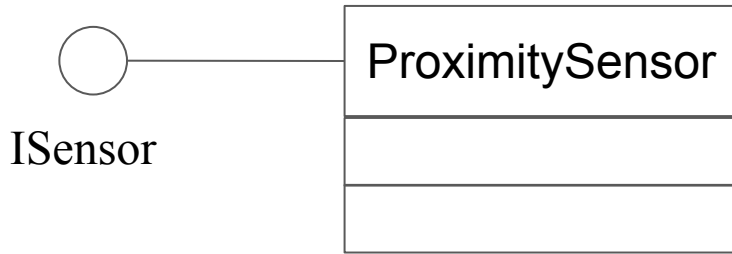
- Công khai (+: public): Không giới hạn quyền truy cập
  - Thường được sử dụng cho các phương thức
- Riêng tư (-: private): Giới hạn trong phạm vi lớp
  - Thường được sử dụng cho các thuộc tính.
- Bảo vệ (#: protected): Giới hạn trong phạm vi cây kế thừa
  - *(Có thể có khác biệt giữa các môi trường, ví dụ Java vs. C++)*.
  - Mức độ giới hạn nằm giữa riêng tư và công khai.



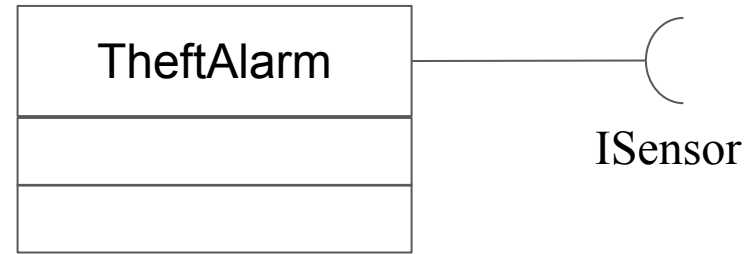
# Giao diện

- Giao diện là 1 kiểu lớp đặc biệt:
  - Các phương thức phải được thiết lập chế độ truy cập công khai.
  - Không tạo được đối tượng của giao diện.
- Giao diện mô tả 1 tập tính năng nhưng không cung cấp triển khai của các tính năng đó.
  - Được triển khai bằng các lớp cụ thể theo cơ chế như kế thừa.
- Giao diện có thể được biểu diễn như lớp với nhãn loài hoặc bằng biểu tượng riêng.

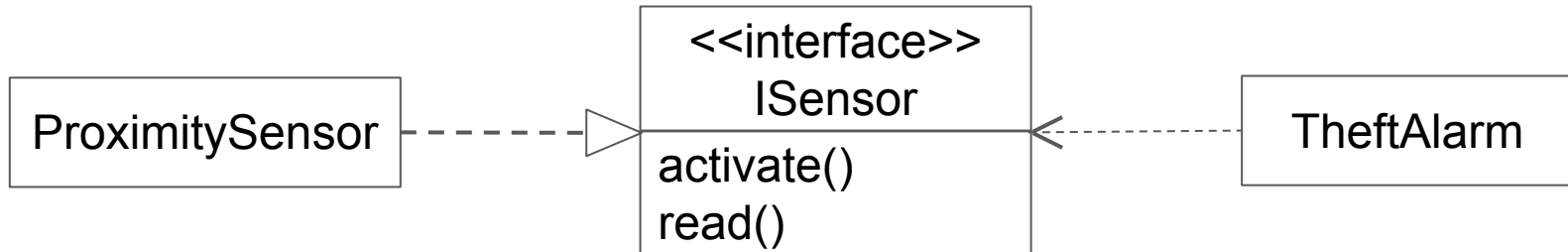
## Ví dụ 3.11. Giao diện



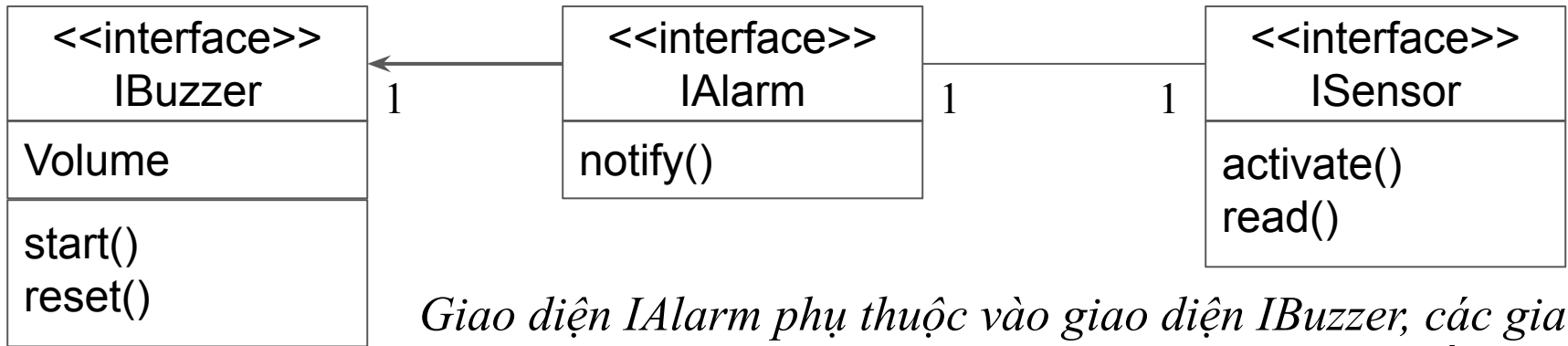
ProximitySensor triển khai giao diện  
ISensor



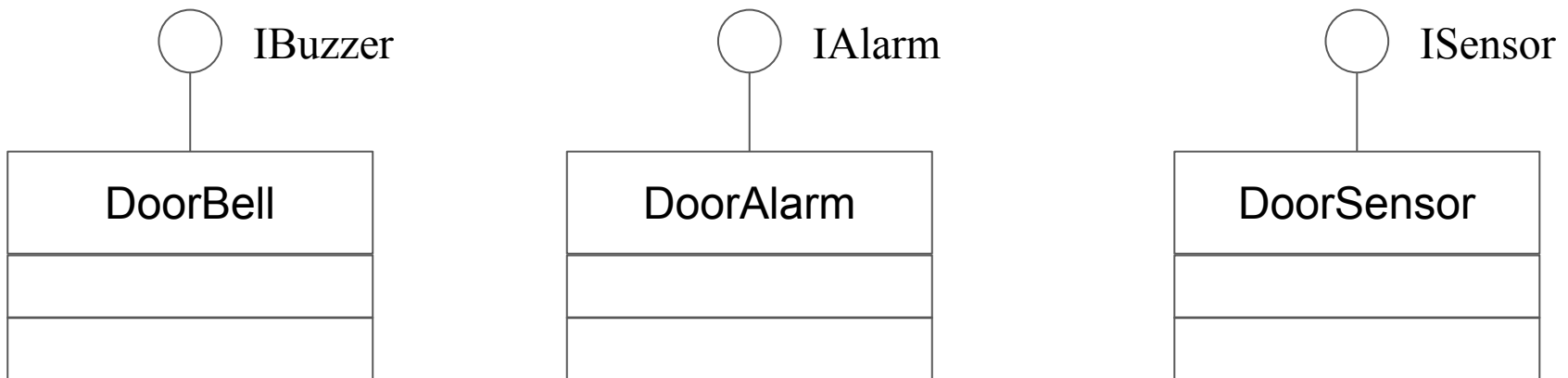
TheftAlarm phụ thuộc vào giao diện  
ISensor



## Ví dụ 3.12. Kết hợp nhiều giao diện



*Giao diện IAlarm phụ thuộc vào giao diện IBuzzer, các giao diện IAlarm và ISensor hình thành 1 giao thức 2 chiều.*

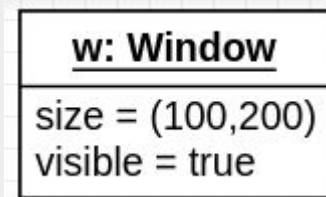


*Các lớp DoorBell, DoorAlarm và DoorSensor tuy được biểu diễn độc lập, nhưng các đối tượng vẫn có thể tương tác thông qua các giao diện.*

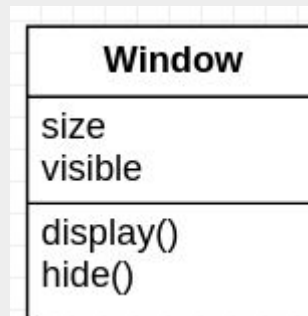
# Biểu đồ đối tượng

- Biểu diễn các đối tượng với các giá trị cụ thể của các thuộc tính và quan hệ giữa các đối tượng
  - Đối tượng = Trường hợp cụ thể (phần tử) của lớp
- Có thể hỗ trợ cho tiến trình xây dựng biểu đồ lớp: Phân tích những tương tác giữa các đối tượng thường đơn giản hơn phân tích những mối quan hệ giữa các lớp.

Một đối tượng w thuộc lớp  
Window

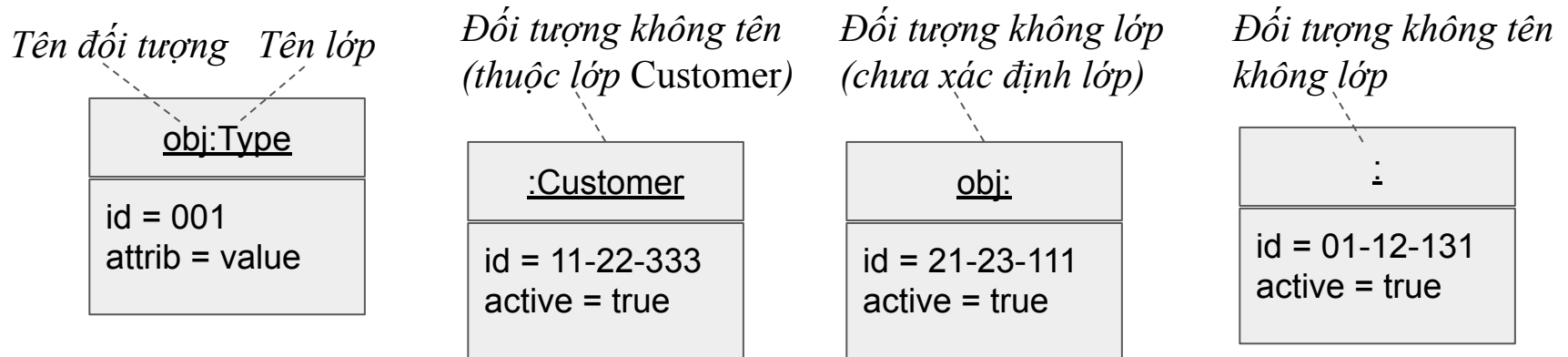


Lớp Window (Cửa sổ)

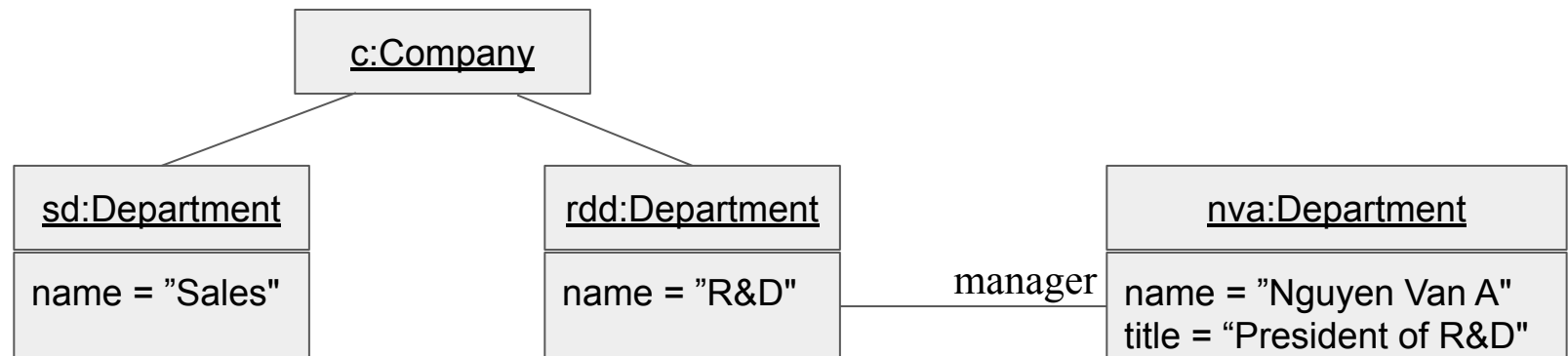


# Biểu đồ đối tượng<sub>(2)</sub>

## Biểu diễn đối tượng



## Ví dụ biểu đồ đối tượng

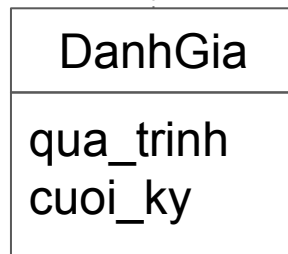
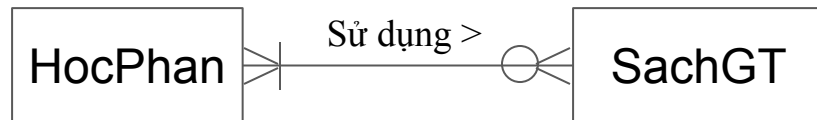
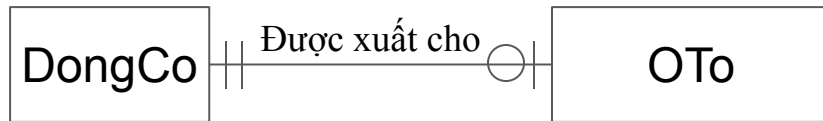


# Biểu đồ thực thể liên kết vs. Biểu đồ lớp

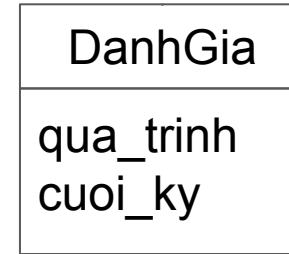
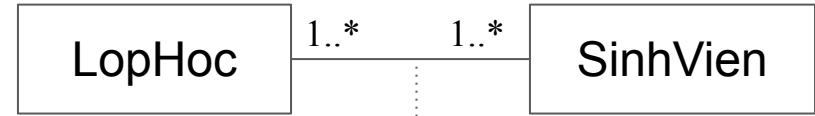
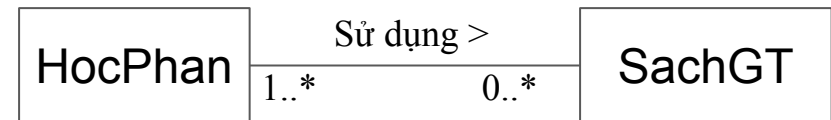
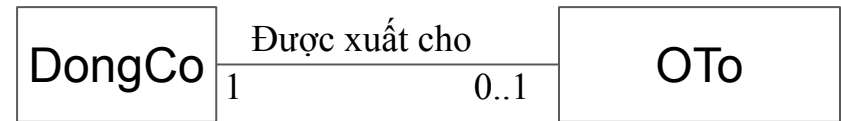
- Tổng quan:
  - Biểu đồ thực thể-liên kết (ERD) mô tả các kiểu thực thể cùng với quan hệ giữa các thực thể.
    - Có nhiều hệ ký hiệu khác nhau;
    - Thường được sử dụng trong thiết kế CSDL quan hệ.
  - Biểu đồ lớp mô tả các lớp cùng với quan hệ giữa các đối tượng.
    - Là 1 biểu đồ UML, hệ ký hiệu trong quy chuẩn quốc tế.
- Phạm vi diễn đạt:
  - Có nhiều điểm tương đồng, tuy nhiên biểu đồ lớp có thể biểu diễn nhiều thông tin hơn. Có thể bảo toàn ý nghĩa khi chuyển đổi biểu đồ thực thể-liên kết thành biểu đồ lớp.
    - Kiểu thực thể => Lớp lĩnh vực chỉ chứa thuộc tính, không có phương thức.
    - Liên kết => Liên kết
    - Thực thể liên kết => Lớp liên kết.
    - V.V..

# Ví dụ 3.13. ERD vs. biểu đồ lớp

## Biểu đồ Thực thể-Liên kết



## Biểu đồ lớp



(Cơ số trong ERD chỉ có thể là 1 trong 4 trường hợp {0..1, 1, 0..\*, 1..\*})

# Nội dung

- Các khái niệm
- Các kỹ thuật
- Các biểu đồ
- Mô tả ràng buộc với OCL
- Đặc tả lớp với thẻ CRC
- Mẫu phân tích



# OCL là gì?

- Ngôn ngữ ràng buộc đối tượng - Object Constraint Language
  - Được phát triển từ năm 1995 ở IBM.
  - Sau đó IBM đề xuất OMG thông qua quy chuẩn.
  - Được sử dụng để đặc tả UML
  - Phiên bản mới nhất (song hành với UML 2.4.1):
    - OCL 2.4: <https://www.omg.org/spec/OCL/2.4/PDF>
    - UML 2.4.1: <https://www.omg.org/spec/UML/2.4.1>
- Có thể được sử dụng để bổ xung các chi tiết cho biểu đồ lớp
  - Các bất biến đối với các thuộc tính, tiền điều kiện và hậu điều kiện đối với các phương thức, v.v..

# Các ràng buộc và ngữ cảnh

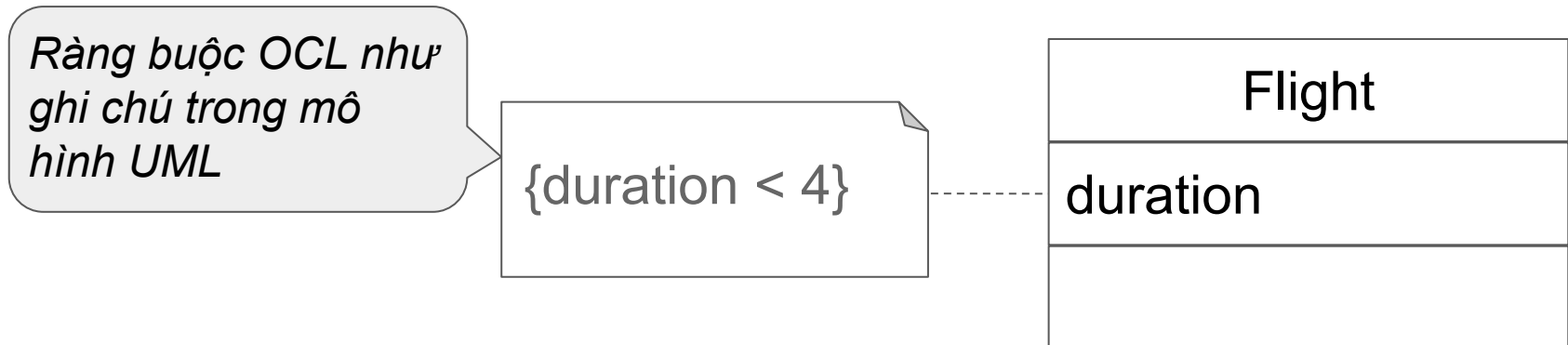
- Ràng buộc là 1 giới hạn đối với 1 hoặc nhiều thành phần của 1 mô hình hướng đối tượng hoặc hệ thống
  - Ràng buộc được thiết lập ở mức lớp, nhưng ý nghĩa của nó được áp dụng cho các đối tượng.
  - Tiêu biểu như: Bất biến, tiền điều kiện, hậu điều kiện.
- Ngữ cảnh kết nối ràng buộc OCL với thành phần cụ thể (lớp, lớp liên kết, giao diện, v.v..) trong mô hình UML.
- Ràng buộc có thể được thêm vào như ghi chú trên biểu đồ UML hoặc tách biệt trong mục riêng.

# Bất biến

- Bất biến là ràng buộc phải đúng đối với đối tượng trong suốt thời gian tồn tại của nó.
- Cú pháp:  
**context** <lớp>  
**inv** [<tên ràng buộc>]:  
<Biểu thức Boolean OCL>

# Bất biến - các mô tả tương đương

- **context** Flight **inv**: self.duration < 4
  - self - Đối tượng đang được kiểm tra ràng buộc.
- **context** Flight **inv**: duration < 4
- **context** Flight **inv** flightDuration: duration < 4
  - flightDuration - Tên được đặt cho ràng buộc



# Tiền điều kiện và hậu điều kiện

- Các ràng buộc được áp dụng cho các thao tác
- Tiền điều kiện phải đúng trước khi thực hiện 1 thao tác.
- Hậu điều kiện phải đúng sau khi thực hiện 1 thao tác.
- Cú pháp:

**context** <lớp>::<phương thức>(<các tham số>)

**pre|post** [<tên ràng buộc>]:

<Biểu thức Boolean OCL>

## Ví dụ 3.14. Mô tả điều kiện

**context** Flight::shiftDeparture(t: Integer)

**pre:**  $t > 0$

**post:**  $\text{result} = \text{departureTime}@pre + t + \text{duration}$

-- Chúng ta muốn trả về thời gian đến đã cập nhật

-- @pre - chỉ được sử dụng trong hậu điều kiện, để chỉ định trạng thái ban đầu.

Flight
departTime /arrivalTime duration
shiftDeparture(t:Integer)

# Kế thừa ràng buộc

Các hệ quả của nguyên lý khả thay Liskov

- Các lớp dưới kế thừa các bất biến của lớp trên
  - Lớp dưới có thể tăng cường các bất biến nhưng không được nói lỏng chúng.
- Lớp dưới có thể nói lỏng tiền điều kiện trong lớp trên
  - **context** SuperClass::foo(i:Integer) **pre:**  $i > 100$
  - **context** SubClass::foo(i:Integer) **pre:**  $i > 0$
  - $\Rightarrow$  Lớp dưới vẫn có thể xử lý dữ liệu đầu vào như lớp trên.
- Lớp dưới có thể tăng cường hậu điều kiện trong lớp trên
  - **context** SuperClass::foo(): Integer **post:**  $result > 0$
  - **context** SuperClass::foo(): Integer **post:**  $result > 10$
  - $\Rightarrow$  Biểu thức gọi phương thức bằng giao diện lớp trên vẫn thu được kết quả  $> 0$  dù đối tượng thuộc lớp dưới.

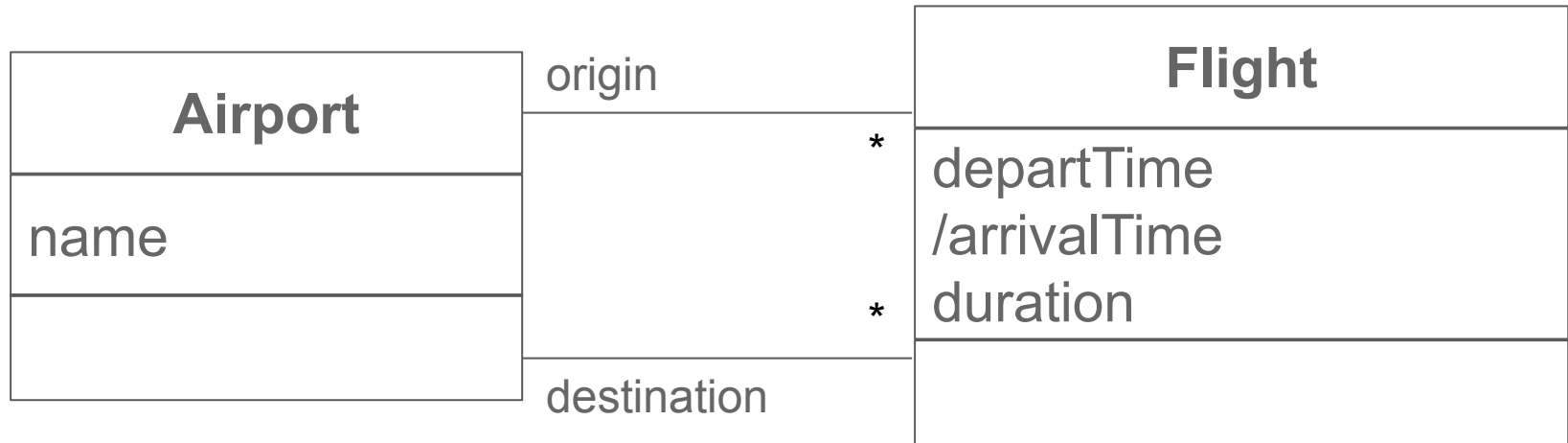
# Các thành phần của biểu thức OCL

- Các kiểu cơ sở:
  - Boolean
  - Integer
  - Real
  - String
- Các lớp từ mô hình UML và các thành phần của chúng
  - Thuộc tính
  - Thao tác truy xuất
- Liên kết từ mô hình UML
  - Bao gồm tên vai trò ở mỗi đầu liên kết



# Các biểu thức duyệt

- Duyệt theo liên kết - Được sử dụng để truy cập các đối tượng liên quan, bắt đầu từ đối tượng ngữ cảnh



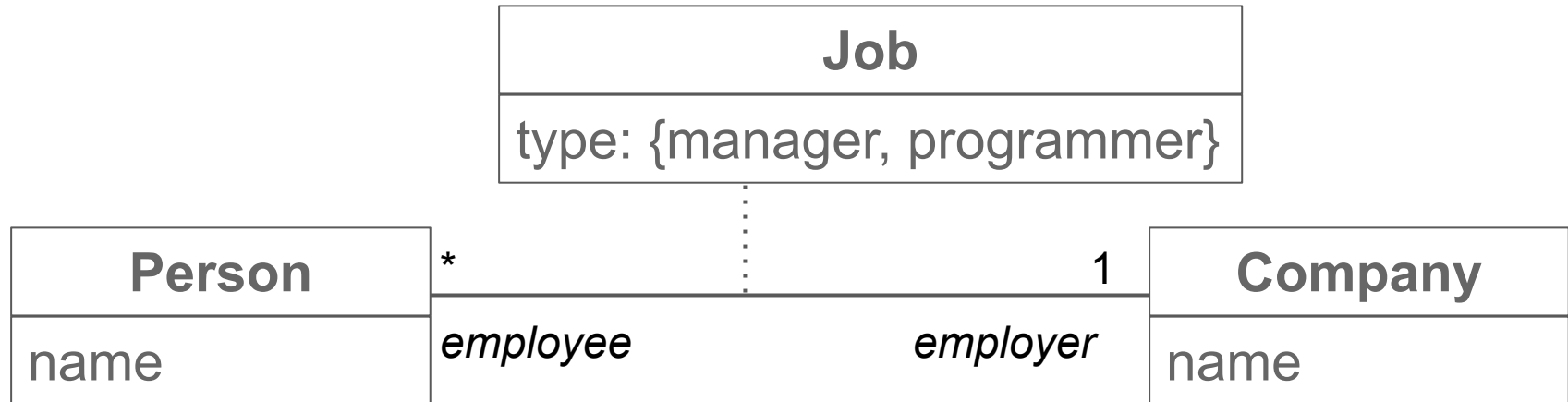
**context** Flight

**inv:** origin != destination

**inv:** origin.name == "Hanoi"

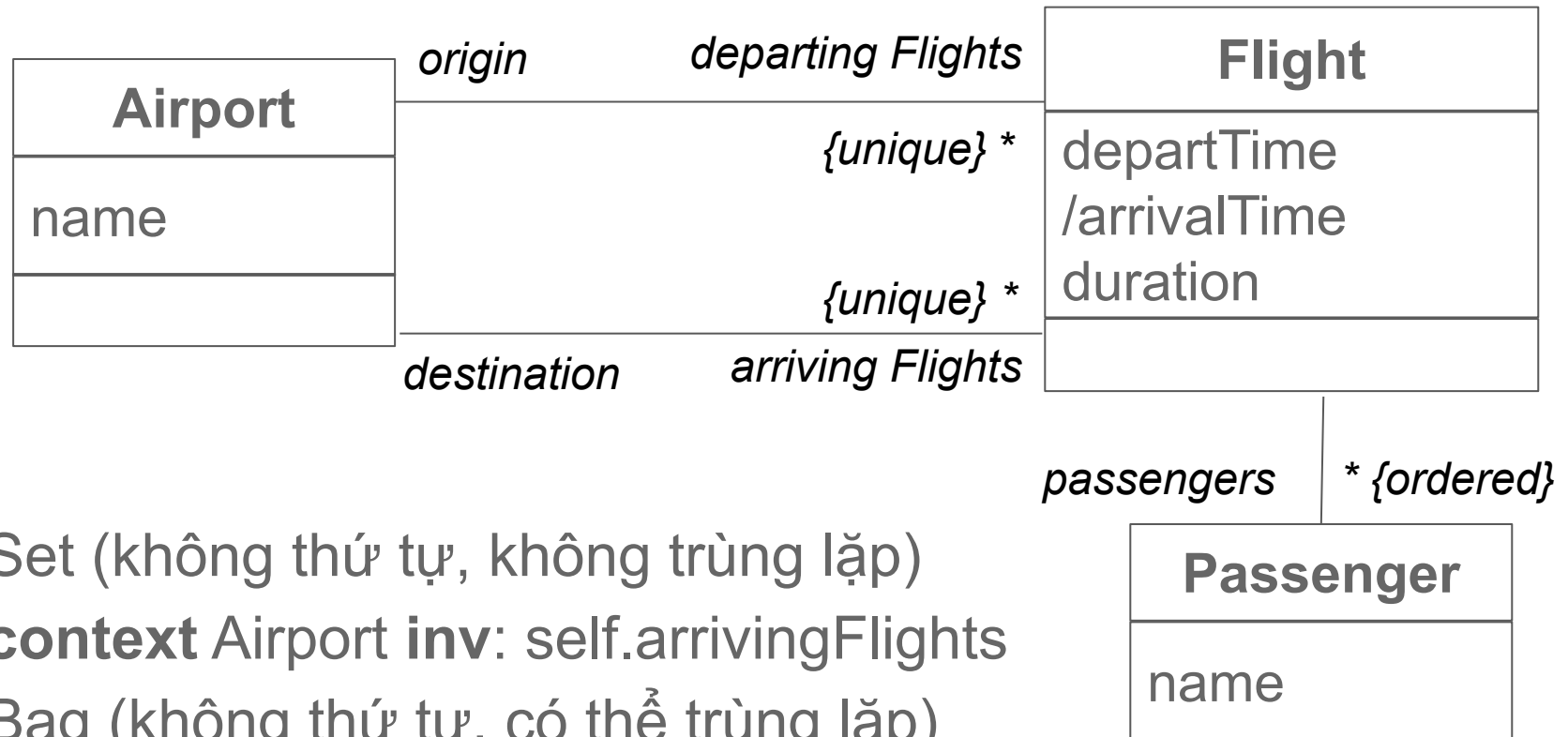
# Duyệt lớp liên kết

- Lớp liên kết không có tên vai trò, vì vậy biểu thức OCL phải sử dụng tên lớp, bắt đầu với chữ thường.



```
context Person inv:  
if self.name = "Ivan Ivanov" then  
    job.type = #manager  
else  
    job.type = #programmer  
endif
```

# Các cấu trúc lưu trữ trong OCL



- Set (không thứ tự, không trùng lặp)  
**context** Airport **inv**: self.arrivingFlights
- Bag (không thứ tự, có thể trùng lặp)  
**context** Airport **inv**: self.arrivingFlights.passengers.name
- Sequence (có thứ tự, có thể trùng lặp)  
**context** Flight **inv**: self.passengers

# Các thao tác với cấu trúc lưu trữ

- **collect:** `collection->collect(expr)` hoặc `collection.expr`
  - Trả về 1 túi (bag) giá trị của biểu thức `expr` trên các phần tử của `collection`.
- **select:** `collection->select(expr)`
  - Trả về tập con của `collection` bao gồm các phần tử có `expr` đúng.
- **forAll:** `collection->forAll(expr)`
  - Thao tác `forAll` đúng nếu `expr` đúng với tất cả các phần tử của `collection`
- **exists:** `collection->exists(expr)`
  - Đúng nếu `expr` đúng với ít nhất 1 phần tử trong `collection`

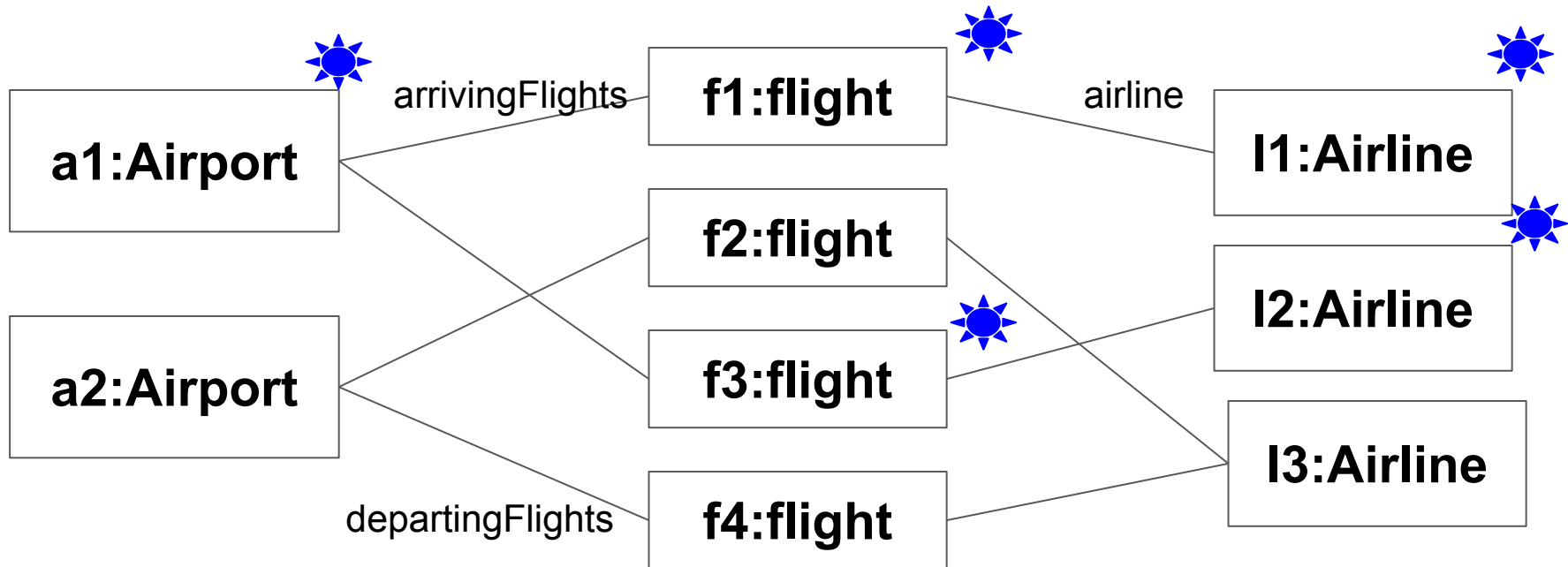
...

## Ví dụ 3.15. Thao tác *collect*



**context** Airport **inv**:

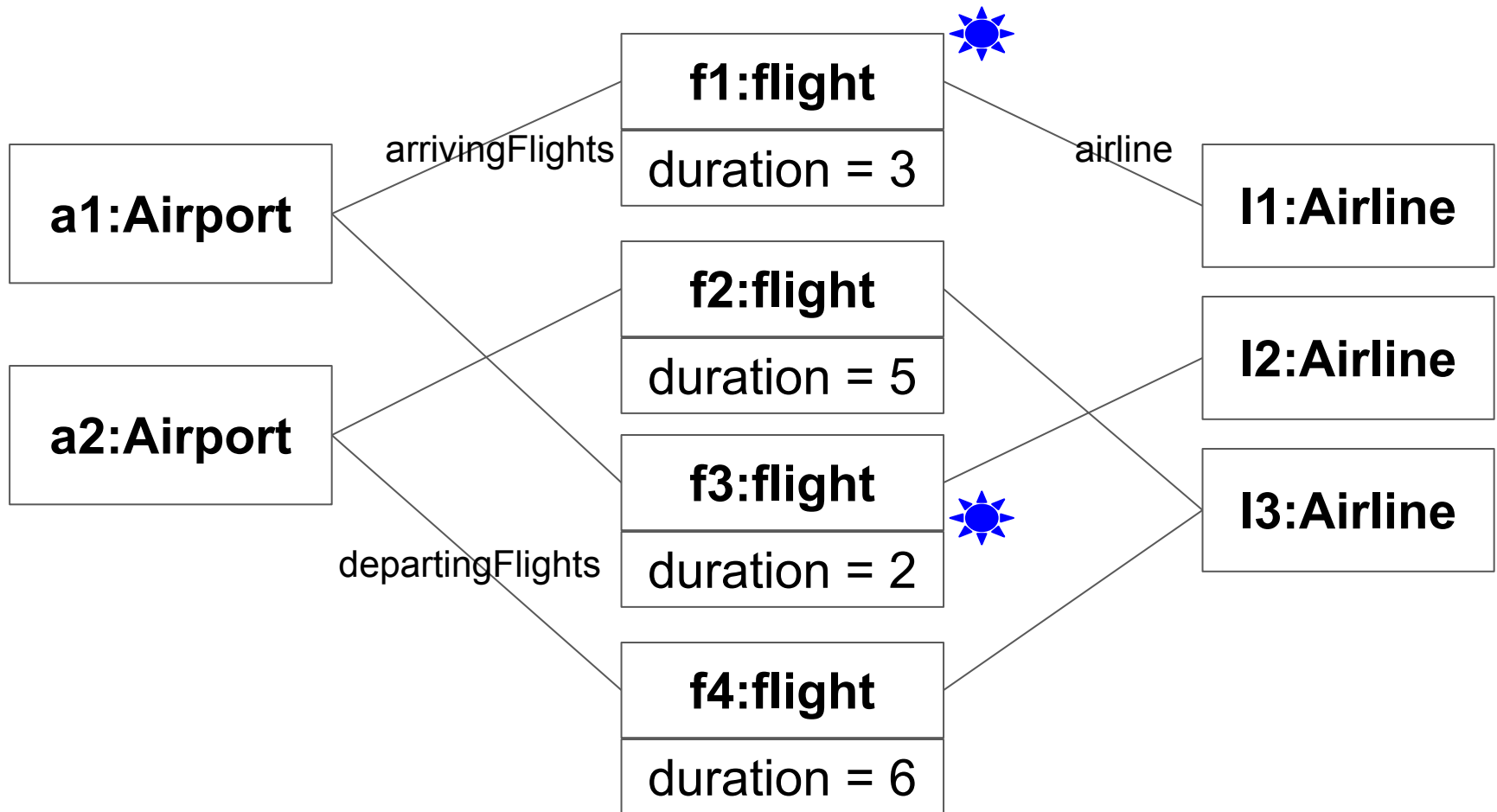
self.arrivingFlights->collect(airline)->notEmpty()



## Ví dụ 3.16. Thao tác *select*

**context** Airport **inv**:

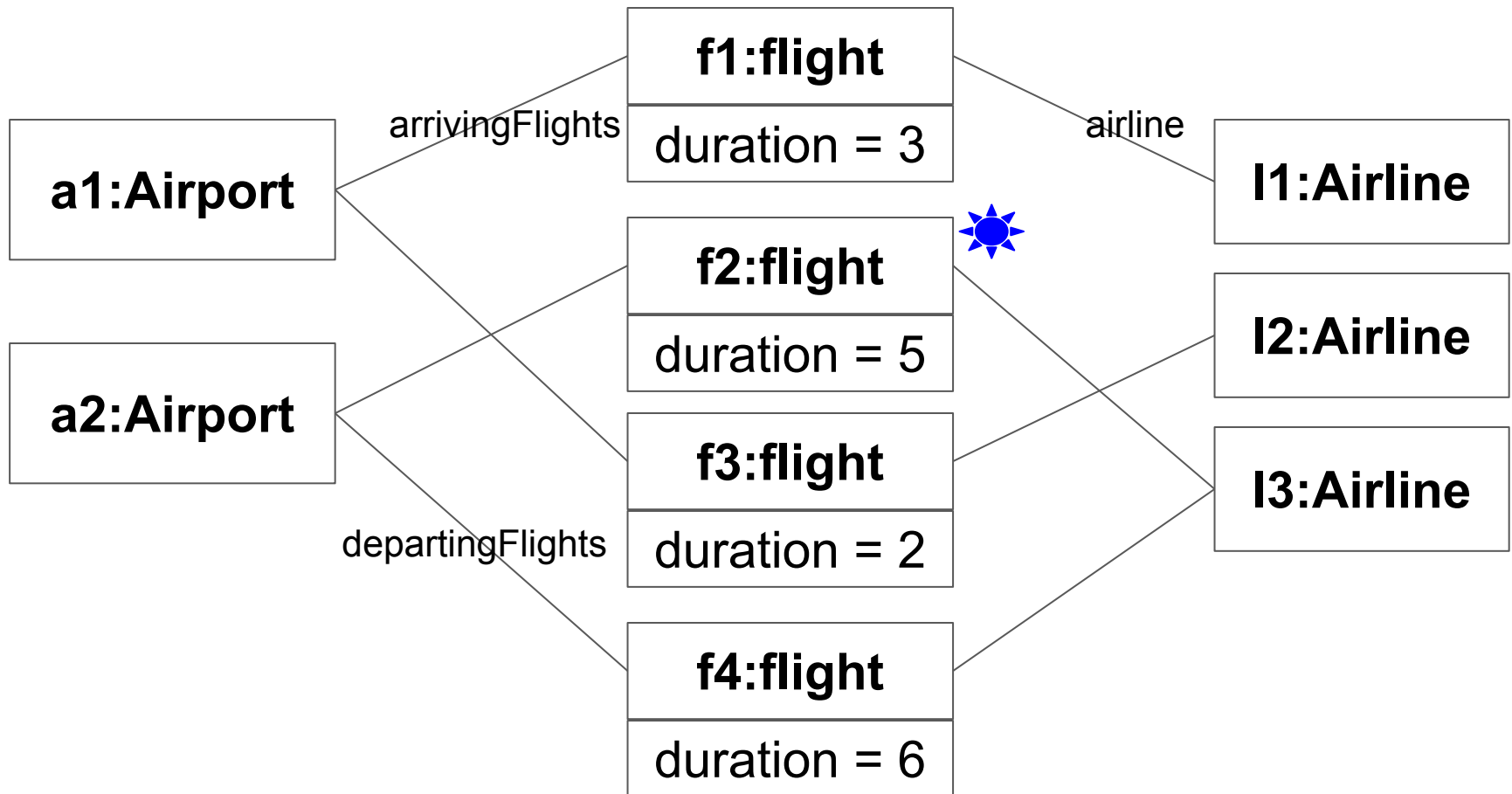
`self.departingFlights->select(duration < 4)->notEmpty()`



## Ví dụ 3.17. Thao tác *exists*

**context** Airport **inv**:

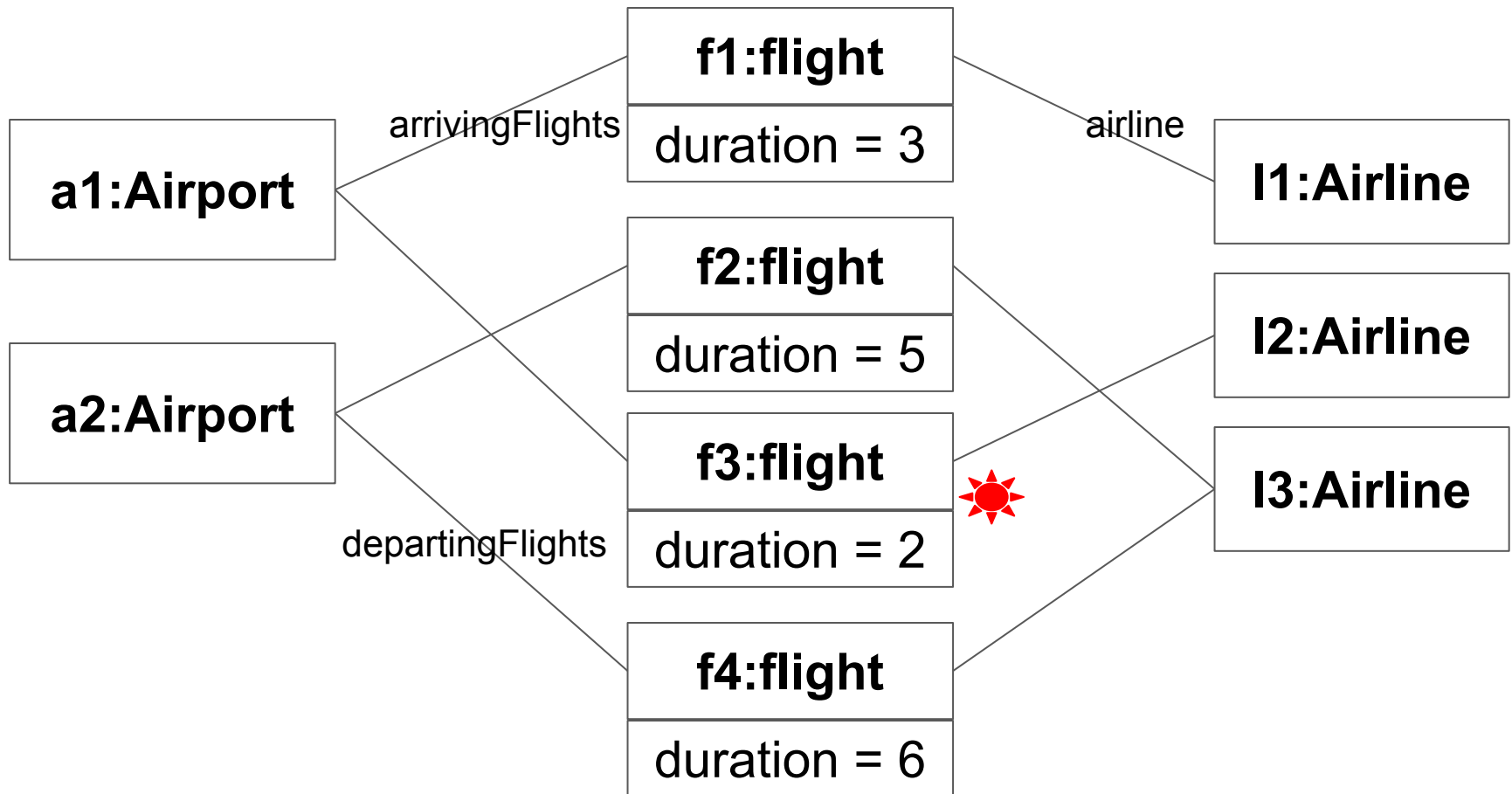
`self.departingFlights->exists(duration = 5)`



## Ví dụ 3.18. Thao tác *forall*

**context** Airport **inv**:

self.departingFlights->forall(duration > 2)





# Nội dung

- Các khái niệm
- Các kỹ thuật
- Các biểu đồ
- Mô tả ràng buộc với OCL
- Đặc tả lớp với thẻ CRC
- Mẫu phân tích



# Thẻ CRC

- Được sử dụng để đặc tả lớp
  - Còn có thể được sử dụng để đóng vai đối tượng trong kiểm tra các kịch bản ca sử dụng
- Các trách nhiệm:
  - Được biểu diễn như các phương thức
  - Biết gì? - các thao tác tra cứu thông tin
  - Làm gì? - những thao tác xử lý
- Lưu dữ liệu gì:
  - Danh sách thuộc tính

# Thẻ CRC<sub>(2)</sub>

- Đối tác:
  - Các đối tượng làm việc cùng nhau để đáp ứng 1 yêu cầu
    - Đối tượng yêu cầu (khách)
    - Đối tượng phản hồi (chủ)
  - Các tương tác được quy định theo hình thức tương tự hợp đồng
    - Nội dung chi tiết được trình bày trong phần thiết kế
- Class Responsibilities and Collaborations

## Ví dụ 3.19. Biểu mẫu thẻ CRC

<b>Tên lớp:</b>	<b>ID:</b>	<b>Kiểu:</b>
<b>Mô tả:</b>		<b>CSD:</b>
<b>Các trách nhiệm</b>		<b>Các đối tác</b>
<b>Các thuộc tính</b>		
<b>Các mối quan hệ</b>		
<b>Kế thừa:</b> <b>Bộ phận-tổng thể:</b> <b>Liên kết:</b>		

## Ví dụ 3.20. Đặc tả lớp với thẻ CRC

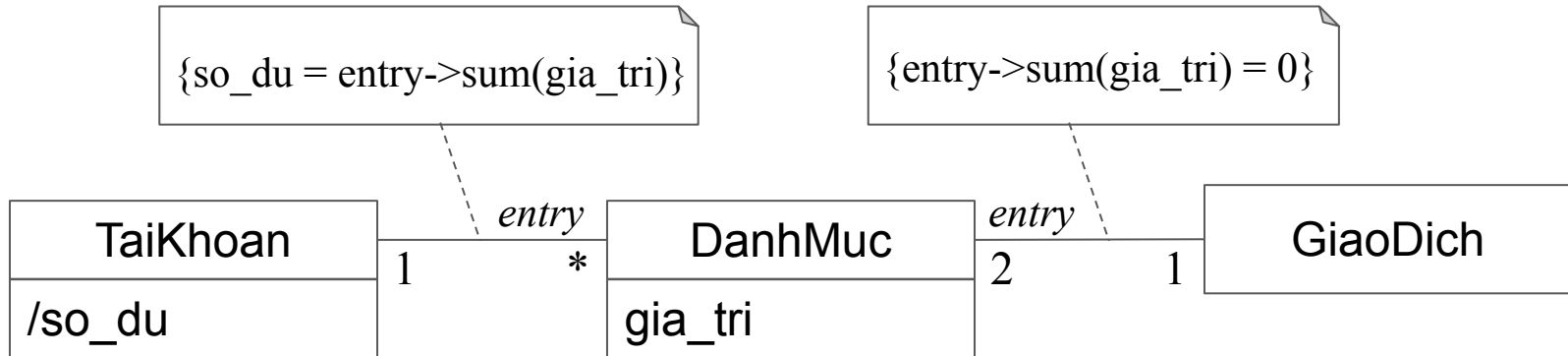
<b>Tên lớp:</b> Bản khảo sát	<b>ID:</b> 1	<b>Kiểu:</b> Lĩnh vực
<b>Mô tả:</b> Danh sách câu hỏi và câu trả lời được sử dụng để xác định dịch vụ.		<b>CSD:</b> UC01
<p style="text-align: center;"><b>Các trách nhiệm</b></p> <p>Hiển thị câu hỏi Lưu câu trả lời</p>		<p style="text-align: center;"><b>Các đối tác</b></p> <p>Câu hỏi khảo sát --</p>
<p style="text-align: center;"><b>Các thuộc tính</b></p> <p>Mã số Danh sách câu hỏi Danh sách câu trả lời</p>		
<p style="text-align: center;"><b>Các mối quan hệ</b></p> <p><b>Kế thừa:</b> --  <b>Bộ phận-tổng thể:</b> Bao gồm các câu hỏi khảo sát  <b>Liên kết:</b> Khách hàng, nhu cầu dịch vụ</p>		

# Nội dung

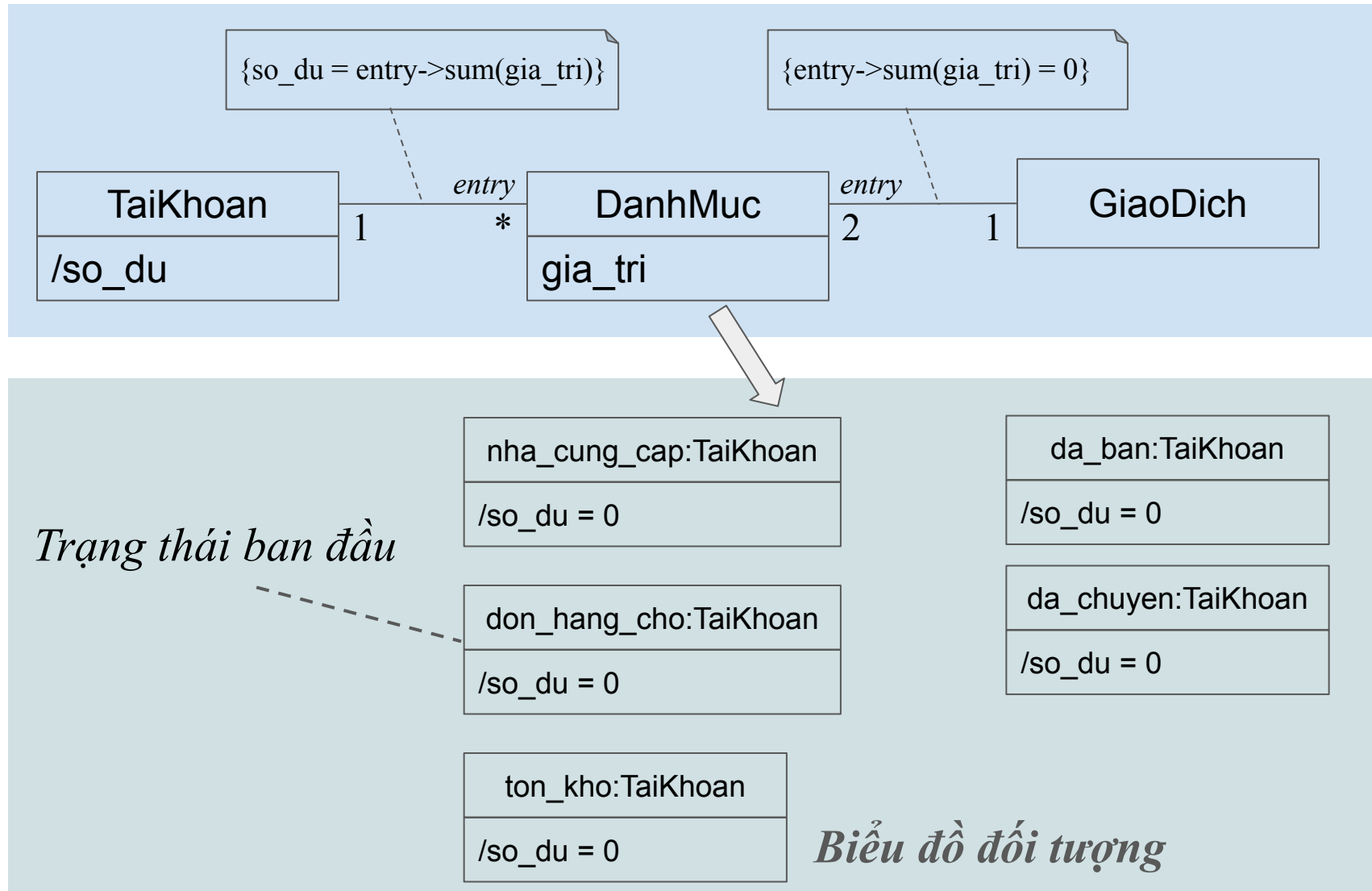
- Các khái niệm
- Các kỹ thuật
- Các biểu đồ
- Mô tả ràng buộc với OCL
- Đặc tả lớp với thẻ CRC
- Mẫu phân tích



# Mẫu tài khoản/giao dịch

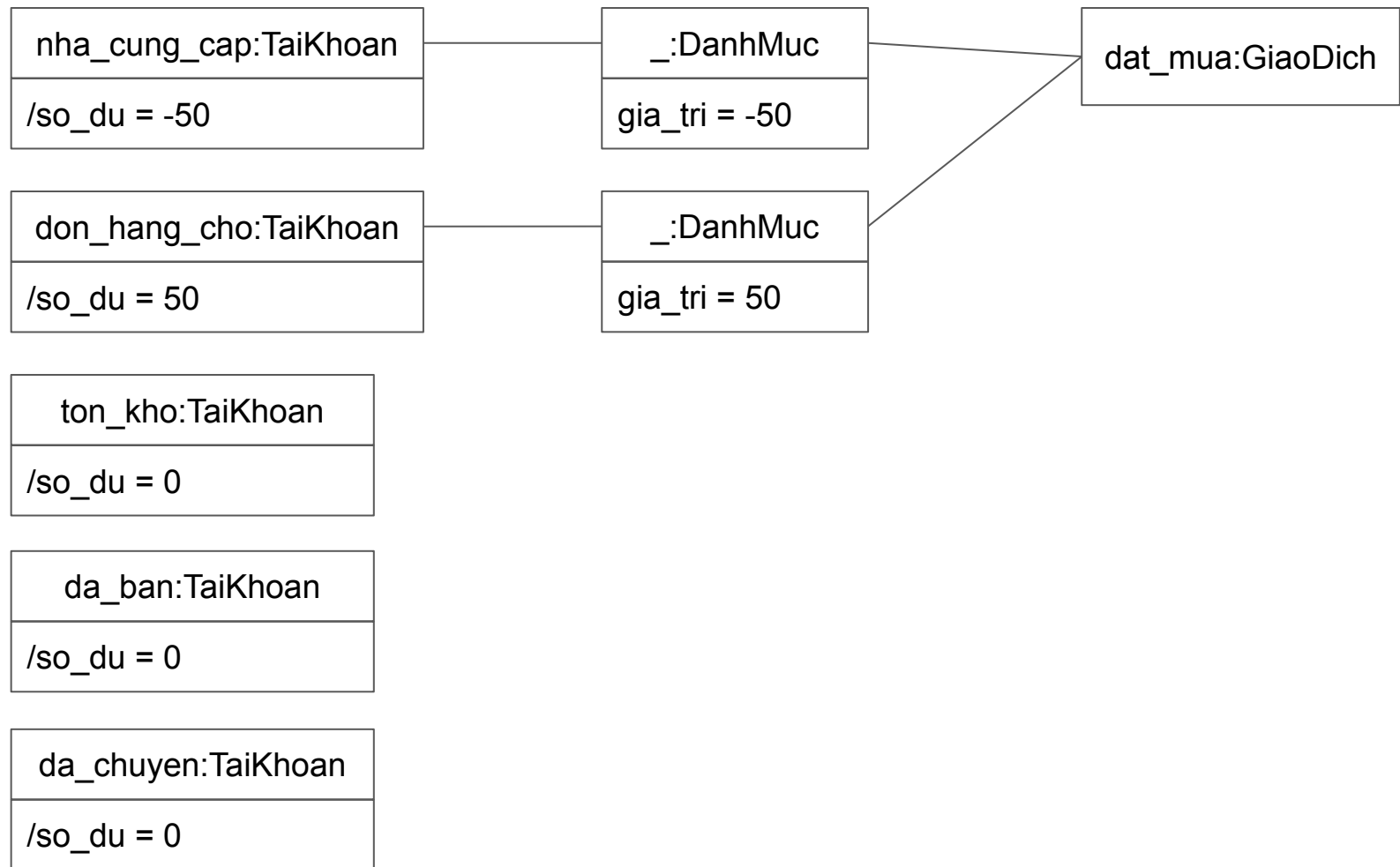


# Ví dụ 3.21. Các đối tượng tài khoản/giao dịch



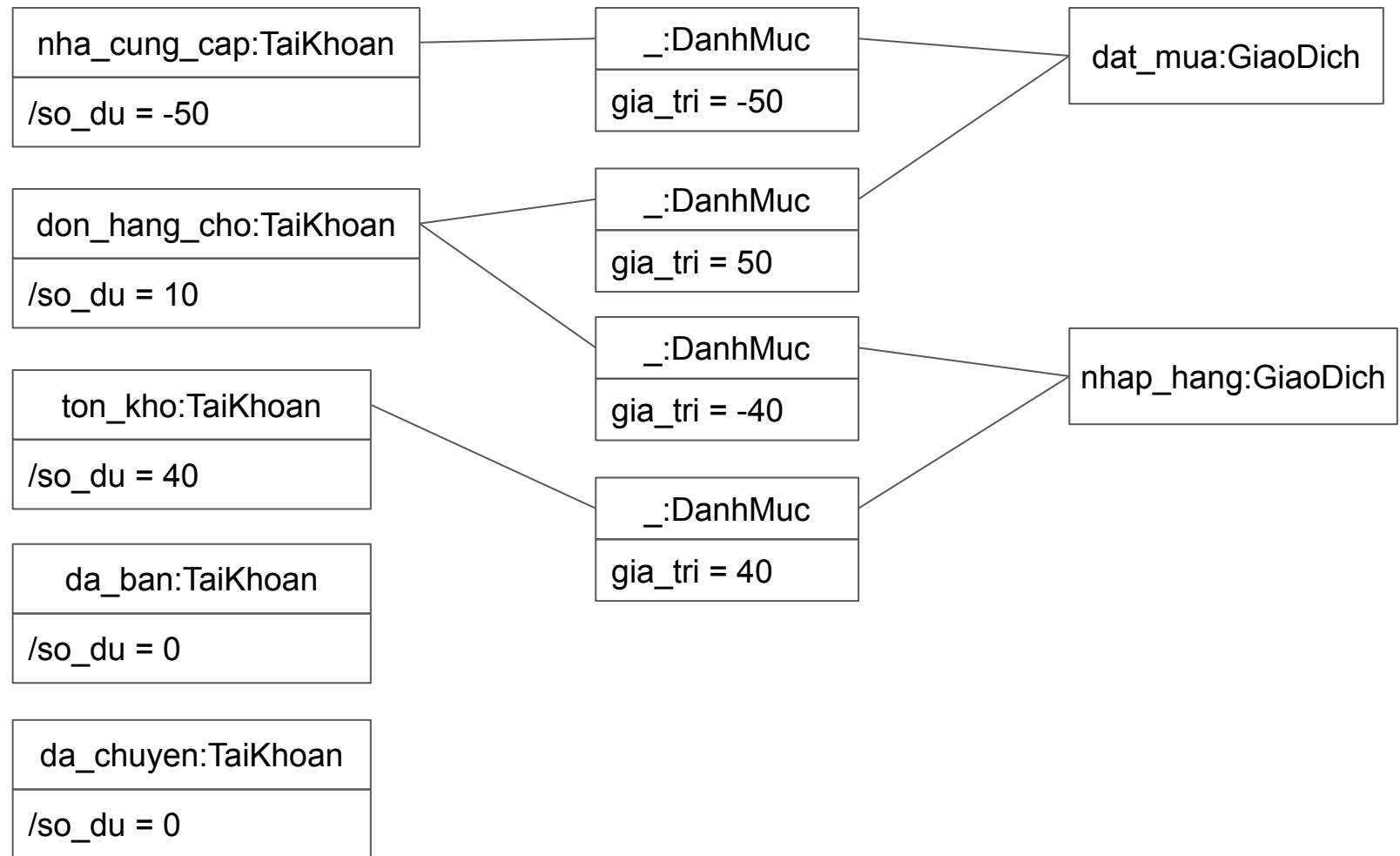


## Ví dụ 3.21. Các đối tượng tài khoản/giao dịch<sup>(2)</sup>



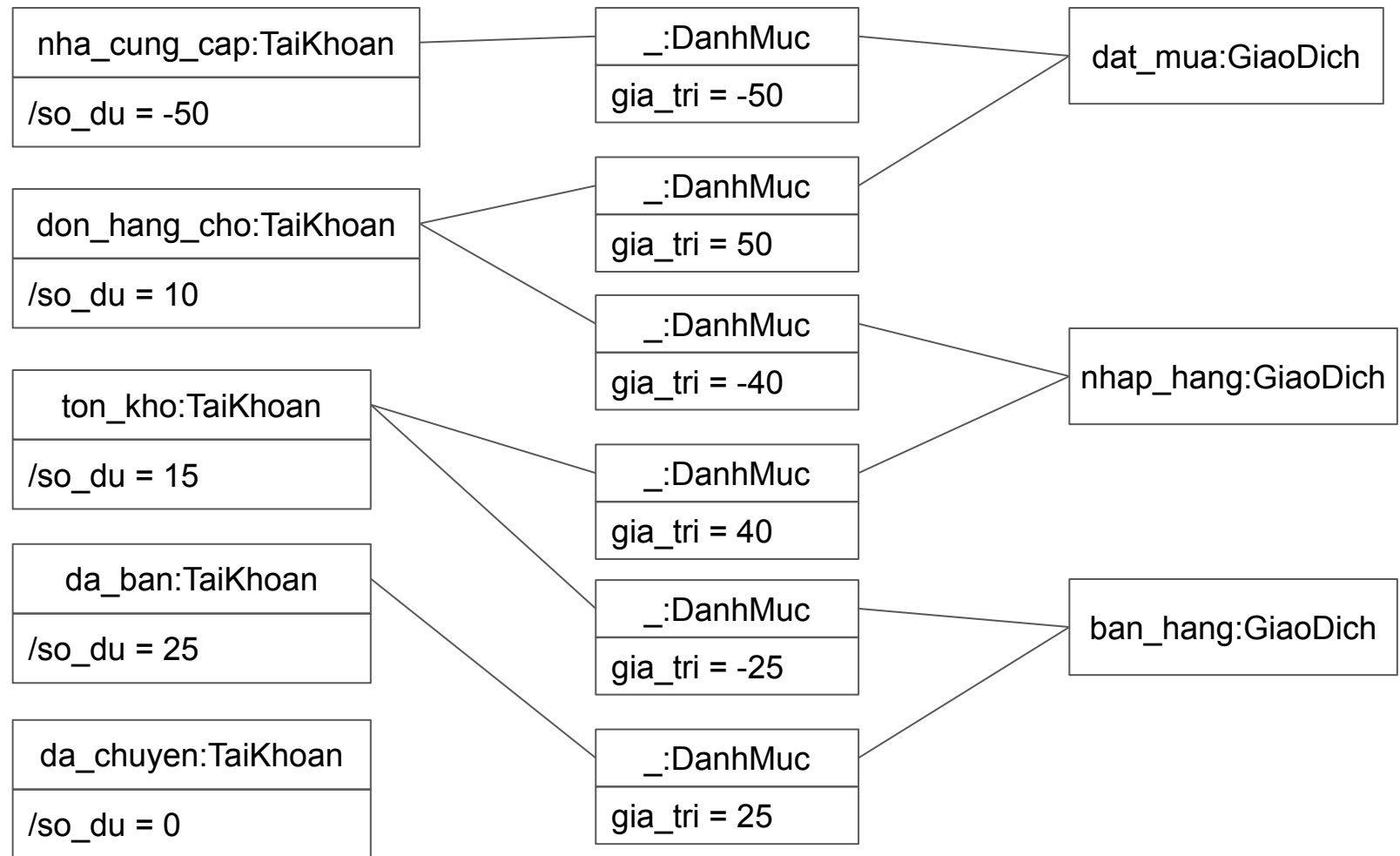
*Trạng thái sau khi đặt mua từ nhà cung cấp*

## Ví dụ 3.21. Các đối tượng tài khoản/giao dịch<sup>(3)</sup>



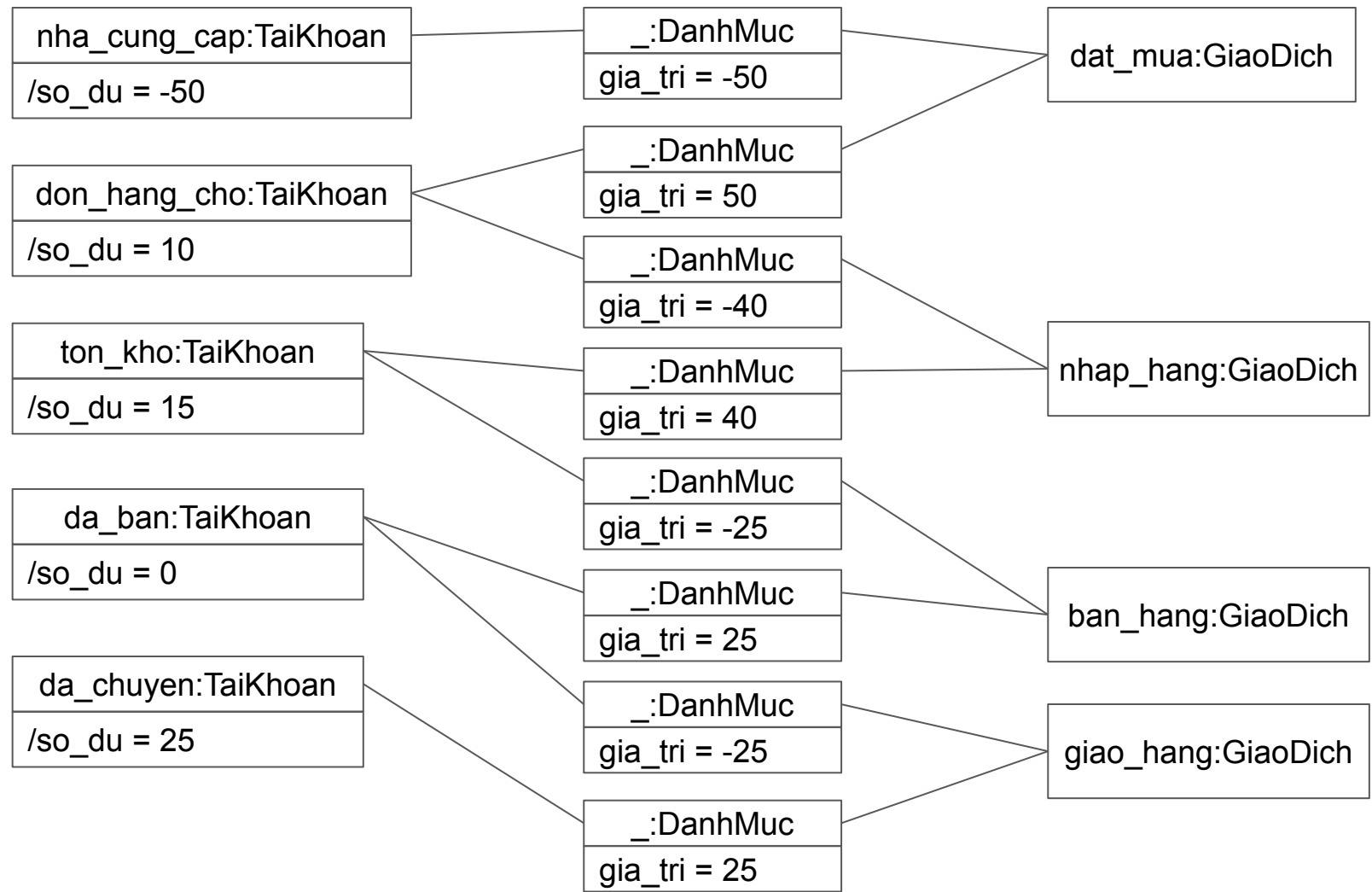
*Trạng thái sau khi nhập 1 phần hàng từ nhà cung cấp*

## Ví dụ 3.21. Các đối tượng tài khoản/giao dịch<sup>(4)</sup>



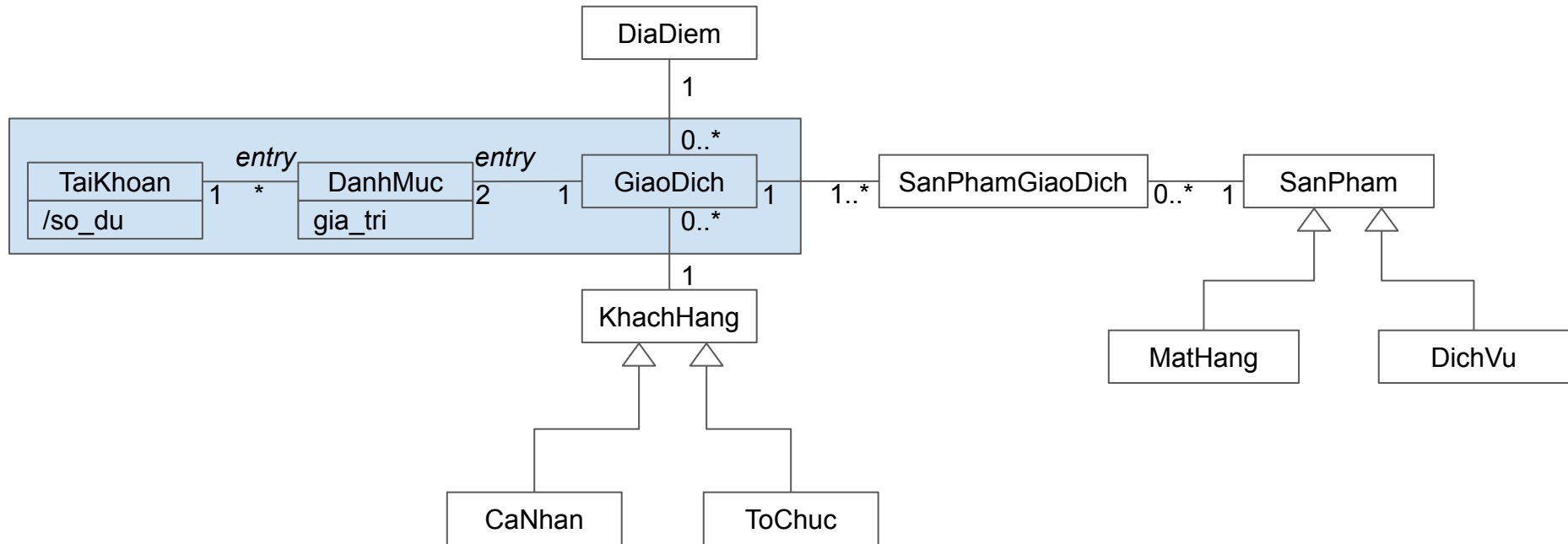
*Trạng thái sau khi bán cho khách hàng*

## Ví dụ 3.21. Các đối tượng tài khoản/giao dịch<sup>(5)</sup>

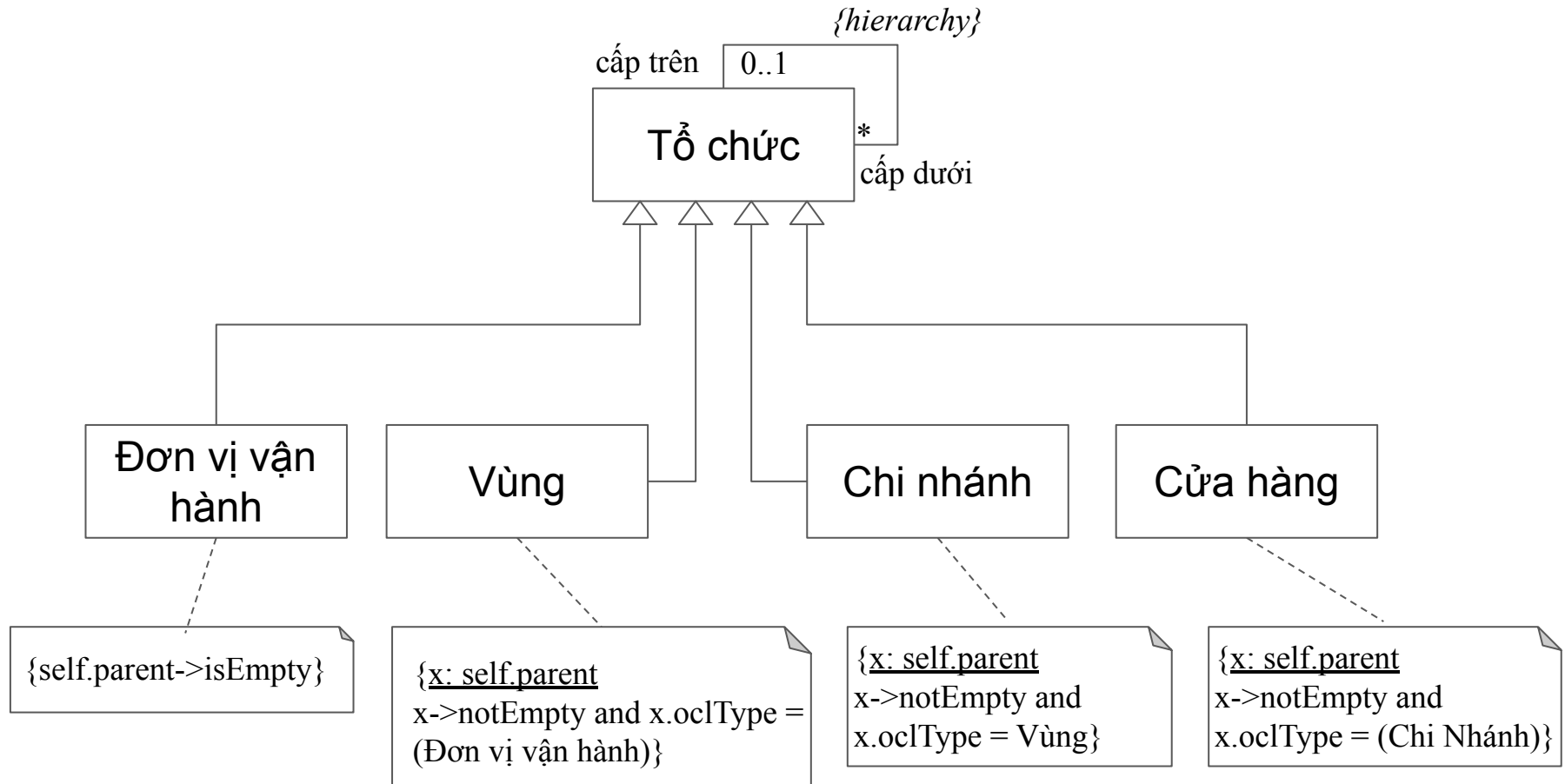


*Trạng thái sau khi giao hàng cho khách*

# Ví dụ 3.22. Sử dụng mẫu tài khoản/giao dịch



# Mẫu phân cấp tổ chức



*Có ích trong trường hợp cần thay đổi cấu trúc phân cấp, ví dụ  
xóa mức vùng*

## Ví dụ 3.23. Các đối tượng phân cấp tổ chức

