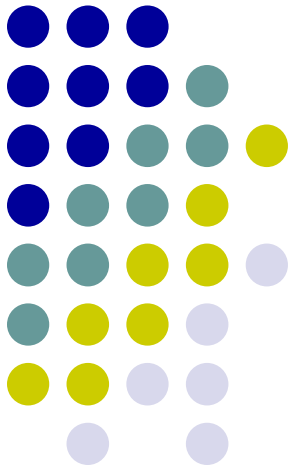
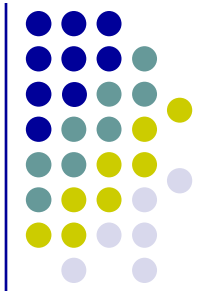
Phân tích hướng đối tượng UML

Giáo viên: Đỗ Thị Mai Hường

Bộ môn : Các hệ thống thông tin

Khoa : CNTT - Học viện kỹ thuật quân sự





Bài 6

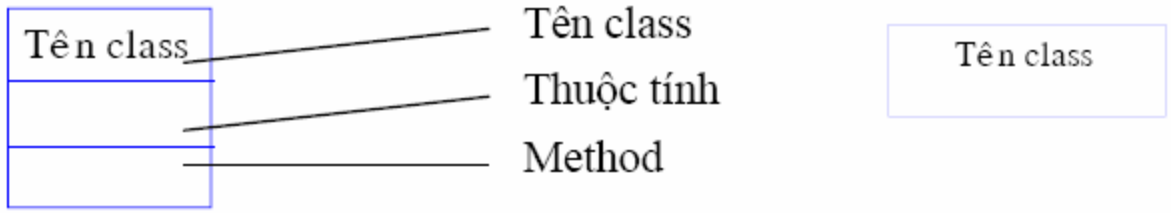
Biểu đồ lớp

Lớp là gì?

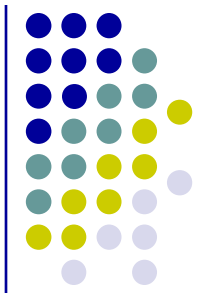
- Đối tượng là cái gì đó tồn tại trong thế giới thực
- Lớp là mô tả thuộc tính, hành vi, ngữ nghĩa của một nhóm đối tượng
  - Lớp xác định thông tin nào được lưu trữ trong đối tượng và hành vi nào đối tượng có
- Thí dụ về lớp: Lớp Employee
  - Đối tượng của lớp có các attribute: Name, Address, Salary
  - Các operation: Thuê mướn, Đuổi việc và Đề bạt nhân viên?
- Ký pháp đồ họa của lớp trong biểu đồ
  - Tên lớp
  - Thuộc tính
  - Thao tác

    - Private

    + Public

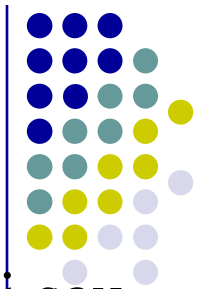


Tìm kiếm lớp như thế nào?



- Việc tìm kiếm đầy đủ lớp là khó khăn
- Khuyến cáo
  - Tìm lớp từ các danh từ trong luồng sự kiện
    - Chú ý rằng danh từ có thể là tác nhân, lớp (, thuộc tính và biểu thức không phải loại trên)
  - Tìm lớp từ biểu đồ tương tác
    - Những cái chung của đối tượng tạo thành lớp
  - Tìm lớp ở các nơi khác
    - Các báo cáo tìm ra trong pha phân tích yêu cầu hình thành lớp giao diện
    - Các thiết bị phần cứng được biểu diễn bởi lớp khác nhau

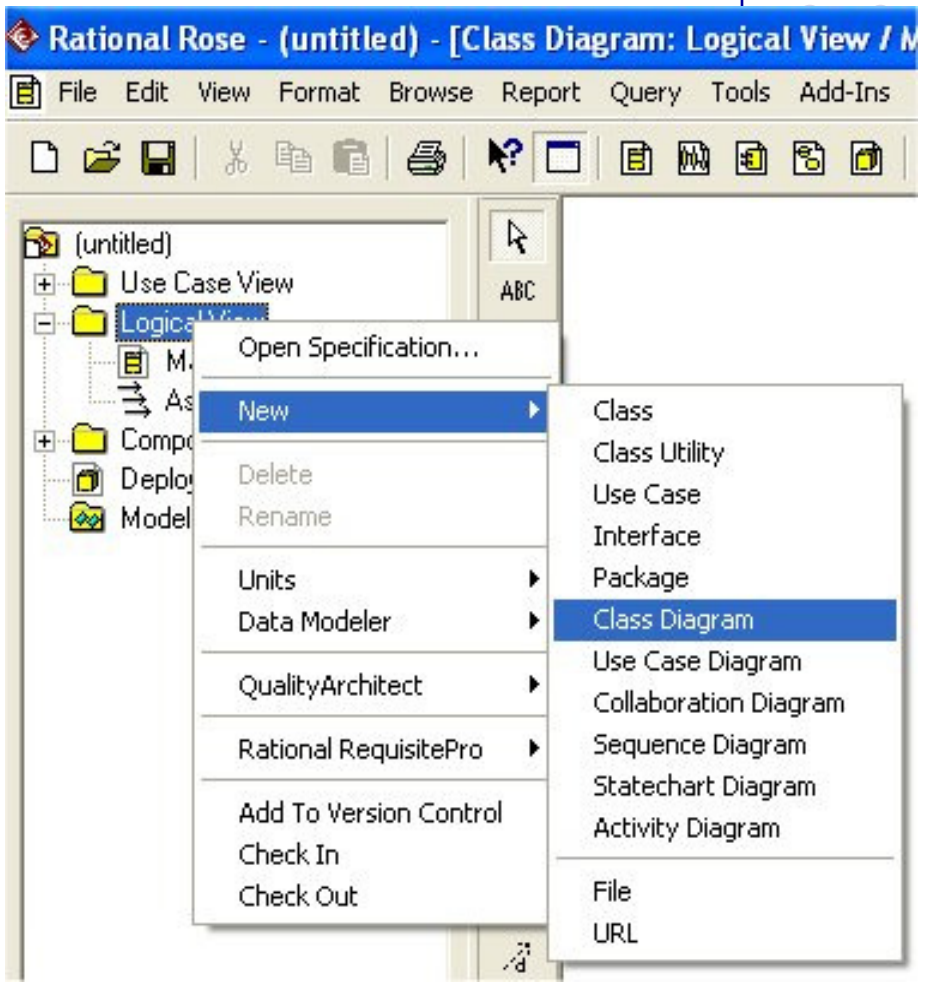
Tìm kiếm lớp như thế nào?



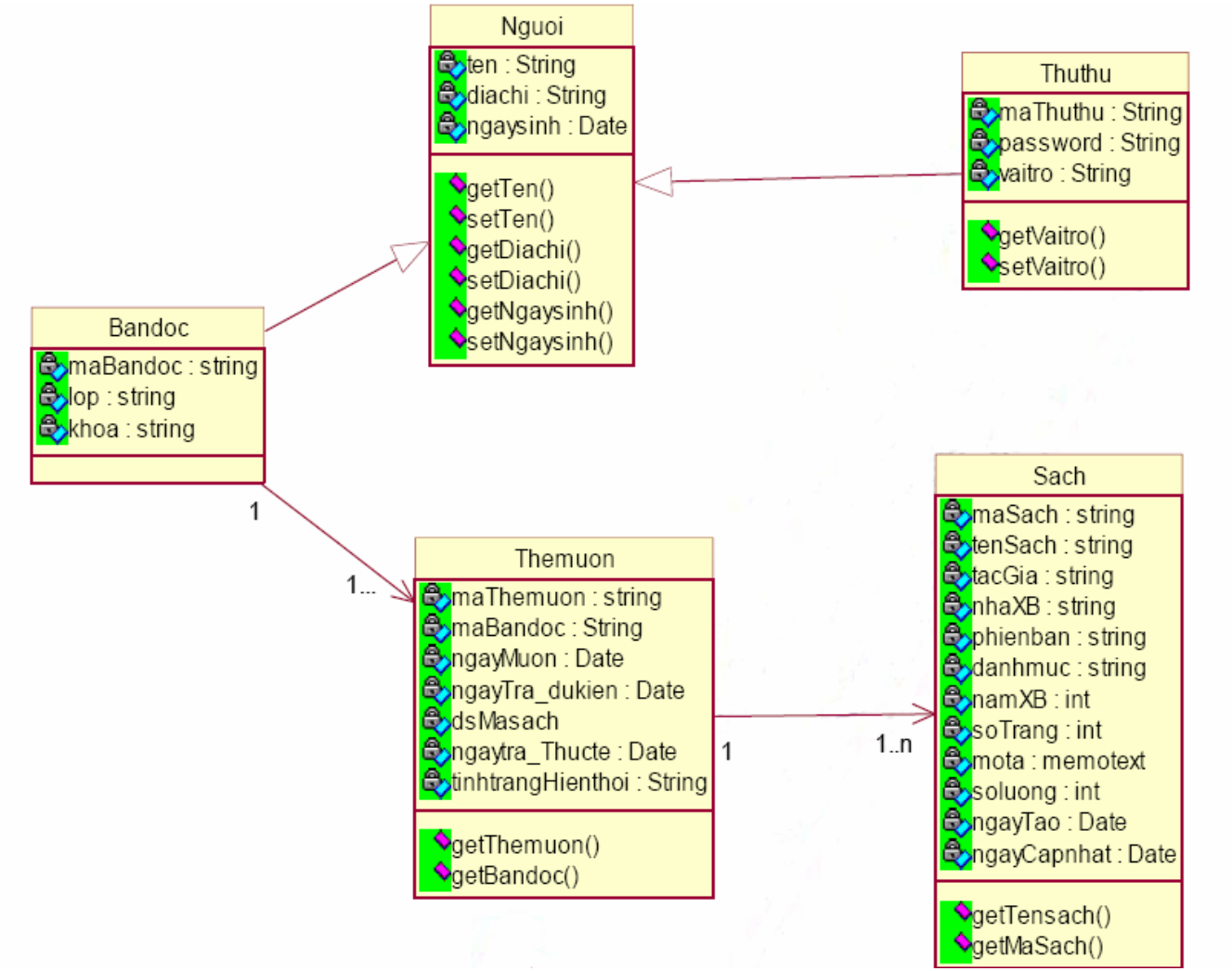
- Cùng với chuyên gia lĩnh vực vấn đề trả lời các câu hỏi sau đây để tìm ra lớp
  - Có thông tin nào cần lưu trữ hay phân tích? Nếu có, nó là lớp
  - Có hệ thống ngoài không? Nếu có thì nó được xem như những lớp chứa trong hệ thống của ta hay hệ thống của ta tương tác với chúng
  - Có mẫu, thư viện lớp, thành phần...? Nếu có, thông thường chúng chứa các ứng viên lớp
  - Hệ thống cần quản lý các thiết bị ngoại vi nào? Mọi thiết bị kỹ thuật nối với hệ thống đều là ứng viên lớp.
  - Tác nhân đóng vai trò tác nghiệp nào? Các nhiệm vụ này có thể là lớp; thí dụ người sử dụng, thao tác viên hệ thống, khách hàng...

Lập biểu đồ lớp

- Biểu đồ lớp cho biết hình ảnh tĩnh của bộ phận hệ thống
- Biểu đồ lớp bao gồm các lớp và quan hệ giữa chúng
- Thông thường mỗi hệ thống có vài biểu đồ lớp
  - Xây dựng vài biểu đồ lớp để mô tả đầy đủ hệ thống
- Biểu đồ lớp giúp người phát triển quan sát, lập kế hoạch cấu trúc hệ thống trước khi viết mã trình
- Rose
  - Biểu đồ lớp được hình thành trong Logical View

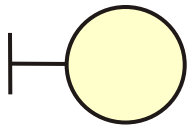


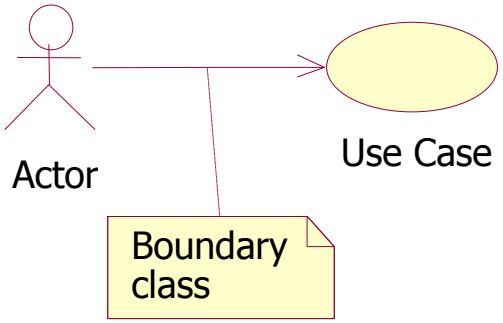
Biểu đồ lớp thực thể

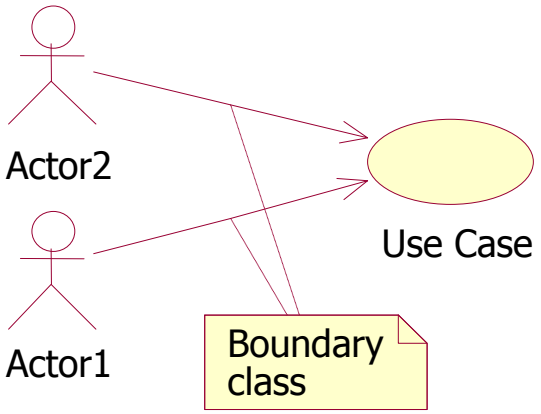


Các nhóm lớp

- Ba nhóm lớp cơ sở sử dụng trong pha phân tích là
  - Boundary: lớp biên (giao diện)
    - Dành cho lớp nằm trên biên hệ thống với thế giới còn lại
    - Chúng có thể là form, report, giao diện với phần cứng như máy in, scanner...
    - Khảo sát biểu đồ UC để tìm kiếm lớp biên

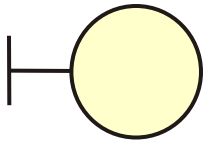




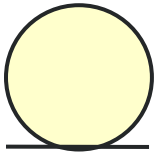


Các nhóm lớp

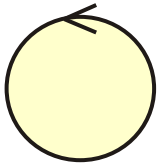
- Ba nhóm lớp cơ sở sử dụng trong pha phân tích là
  - Boundary
  - Entity: lớp thực thể
    - Lớp thực thể là lớp lưu trữ thông tin sẽ ghi vào bộ nhớ ngoài
    - Tìm chúng trong luồng sự kiện và biểu đồ tương tác
    - Thông thường phải tạo ra bảng CSDL cho lớp loại này
      - Mỗi thuộc tính của lớp thực thể sẽ là trường trong bảng CSDL
  - Control: lớp điều khiển
    - Có trách nhiệm điều phối hoạt động của các lớp khác
    - Thông thường mỗi UC có một lớp điều khiển
    - Nó không thực hiện chức năng nghiệp vụ nào
    - Các lớp điều khiển khác: điều khiển sự kiện liên quan đến an ninh và liên quan đến giao dịch CSDL



BoundaryClass

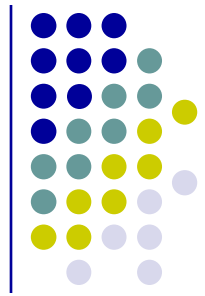


EntityClass



ControlClass

Đặc tả lớp trong biểu đồ



- Đặc tả lớp bao gồm
  - Tên lớp
    - Mỗi lớp trong mô hình có tên duy nhất
    - Thông thường sử dụng danh từ đơn, không nên có dấu cách
    - Thí dụ: Flight, Airplane
  - Phạm vi (Visibility)
    - Xác định khả năng nhìn thấy lớp từ ngoài gói
    - Các loại
      - Public: mọi lớp trong hệ thống có thể nhìn thấy
      - Private hay Protected: có thể nhìn thấy từ bên trong lớp hay từ lớp friend
      - Package hay Implementation: chỉ các lớp trong cùng gói mới nhìn thấy
  - Tính nhiều (Multiplicity)

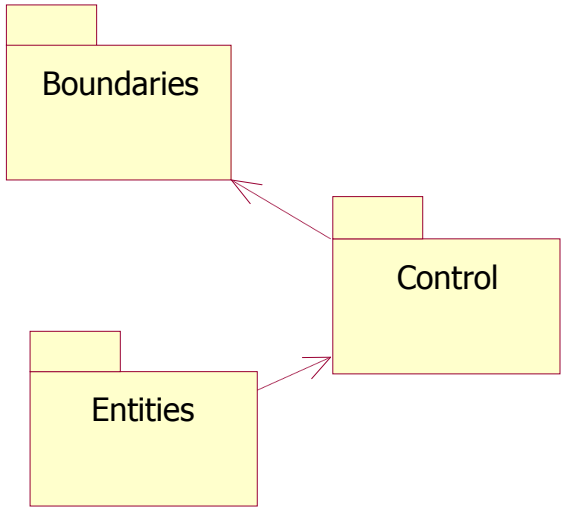
Đặc tả lớp trong biểu đồ

- Đặc tả lớp bao gồm
  - ...
  - Tính nhiều của lớp (Multiplicity)
    - Là số hiện thực mong đợi của lớp

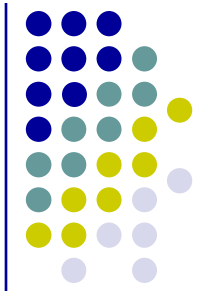
Multiplicity	Ý nghĩa
n (Mặc định)	Nhiều
0..0	Không
0..1	Không hoặc 1
0..n	Không hoặc nhiều
1..1	Chính xác 1
1..n	Một hoặc nhiều

Gói các lớp

- Gói (Packages) để nhóm các lớp có những cái chung
- Có nhiều quan điểm hình thành gói
  - Gói lớp theo prototype
    - Thí dụ có gói Boundaries, gói Control và gói Entities
  - Gói lớp theo chức năng
    - Thí dụ gói Security, gói Reporting, gói Error Handling...
  - Sử dụng tổ hợp hai loại tiếp cận trên để hình thành gói
- Có thể tổ chức gói bên trong gói khác
- Quan hệ giữa các gói hình thành trên cơ sở quan hệ giữa các lớp trong các gói.



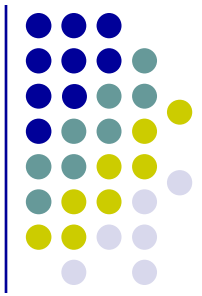
Thuộc tính lớp



- Thuộc tính là nhóm thông tin liên kết với lớp
- Có thể gắn một hay nhiều thuộc tính vào lớp
- Tìm kiếm thuộc tính?
  - Tìm trong tài liệu UC
  - Tìm các danh từ trong luồng sự kiện
    - Thí dụ: "Người sử dụng nhập tên, địa chỉ ngày sinh của Nhân viên"

      -> Tên, địa chỉ, ngày sinh là danh từ và là thuộc tính của lớp Nhân viên
  - Tìm trong tài liệu yêu cầu hệ thống
    - Thí dụ tài liệu yêu cầu hệ thống mô tả các thông tin cần thu thập
  - Tìm thuộc tính trong cấu trúc CSDL
    - Nếu đã xác định cấu trúc CSDL thì các trường trong bảng là thuộc tính của lớp

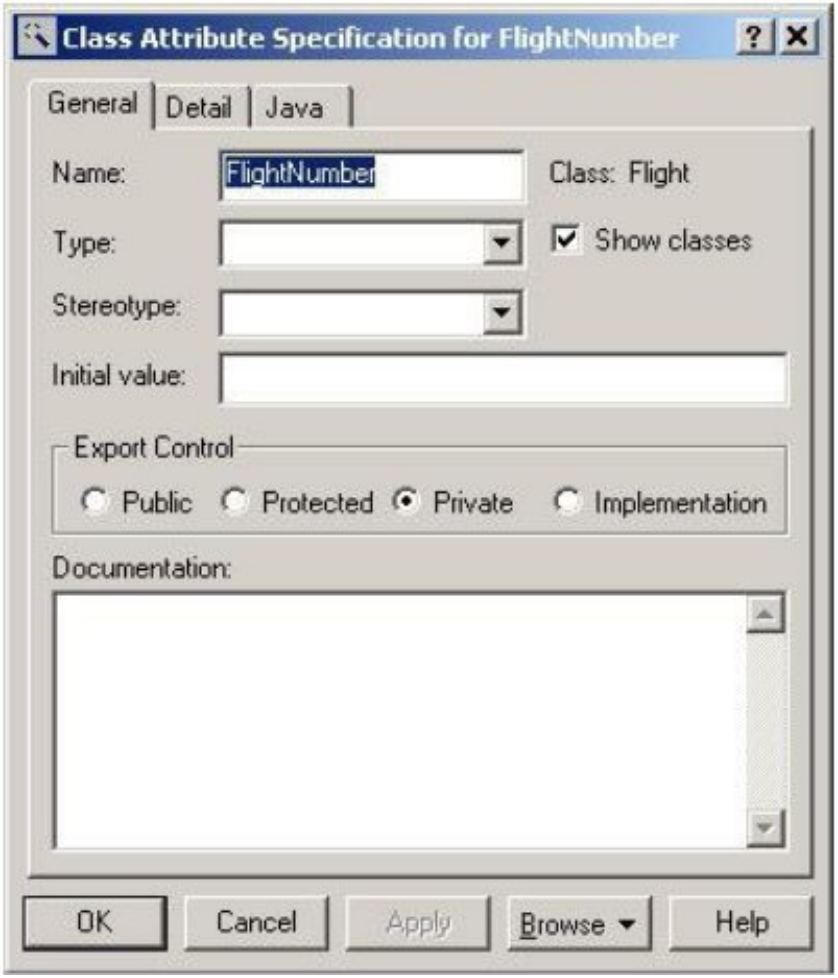
Thuộc tính lớp



- Trong trường hợp khó khăn quyết định danh từ tìm ra là thuộc tính hay là lớp
  - Thí dụ: Tên công ty là thuộc tính hay lớp?
  - Loại ứng dụng cụ thể quyết định việc này
  - Mặt khác cần quan sát nhóm thông tin có hành vi hay không
- Khi kết thúc tìm kiếm thuộc tính
  - Đảm bảo rằng các thuộc tính tìm ra phải có ích cho yêu cầu hệ thống
  - Gán thận trọng thuộc tính cho các lớp
  - Không nên hình thành lớp có quá nhiều hay quá ít thuộc tính (tốt nhất nên có lớp ít hơn 10 thuộc tính)

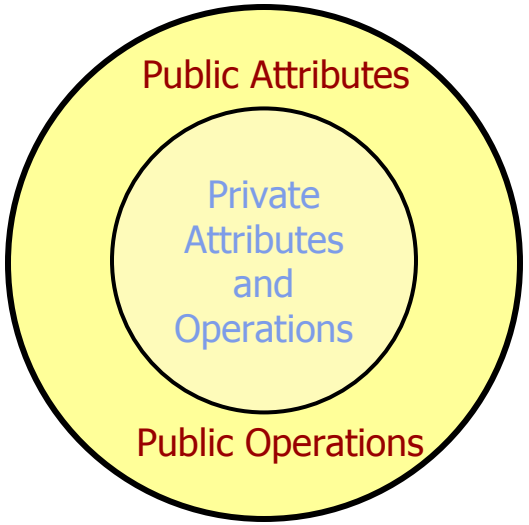
Đặc tả thuộc tính lớp

- Trong Rose: sử dụng cửa sổ đặc tả thuộc tính để gán đặc tính cho thuộc tính
- Với mỗi thuộc tính trong biểu đồ cần có
  - Tên thuộc tính
  - Kiểu dữ liệu thuộc tính lưu trữ. Phụ thuộc vào ngôn ngữ lập trình
    - Thí dụ, Add : String
  - Giá trị khởi đầu
    - Thí dụ, IDNumber: Interger=0
  - Stereotype
  - Phạm vi (visibility)
  - ....



Đặc tả thuộc tính lớp

- Với mỗi thuộc tính trong biểu đồ cần có
  - ...
  - Phạm vi (visibility)
    - Một tính chất quan trọng của lập trình hướng đối tượng là tính gói
    - Bốn lựa chọn phạm vi cho thuộc tính
      - Public: Mọi lớp đều nhìn thấy thuộc tính (+)
      - Private: Lớp khác không nhìn thấy thuộc tính (-)
      - Protected: Các lớp kế thừa có thể nhìn thấy (#)
      - Package và Implementation: Thuộc tính là public đối với các lớp trong cùng gói

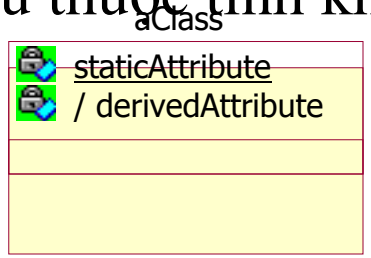


	Public	+	Public
	Private	-	Private
	Protected	#	Protected
	Package (Implementation)		

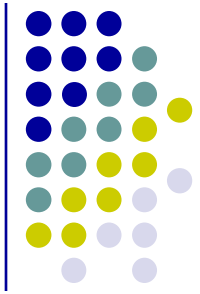
  - ...

Đặc tả thuộc tính lớp

- Với mỗi thuộc tính trong biểu đồ cần có
  - ...
  - Kiểu lưu trữ thuộc tính
    - By value: Lớp chứa thuộc tính
    - By reference: Thuộc tính đặt ngoài lớp, lớp có con trỏ đến thuộc tính
    - Unspecified: Không xác định
  - Thuộc tính tĩnh
    - Là thuộc tính chia sẻ cho mọi hiện thực lớp
    - Ký hiệu trong lớp là tên thuộc tính có gạch chân (phiên bản cũ: $)
  - Thuộc tính suy diễn
    - Là thuộc tính được tạo bởi 1 hay nhiều thuộc tính khác
    - Ký hiệu: dấu / trước tên thuộc tính
  - ...



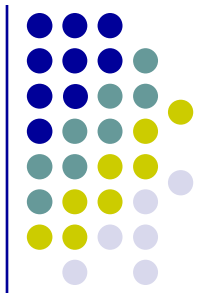
Thao tác lớp



- Thao tác là hành vi kết hợp với lớp, nó xác định trách nhiệm của lớp
- Mô tả thao tác bao gồm
  - Tên thao tác
  - Tham số thao tác
  - Kiểu giá trị cho lại
- Ký pháp trong UML

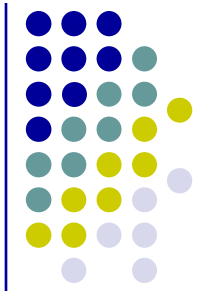
  Operation Name (arg1: arg1 data type, arg2: arg2 data type...): return type
- Chú ý khi bổ sung thao tác trong lớp
  - Không nên để lớp chỉ có 1 hay 2 thao tác
  - Nếu lớp không có thao tác thì mô hình hóa nó như thuộc tính
  - Nếu lớp có quá nhiều thao tác thì khó quản lý, nên chia sẻ chúng ra các lớp khác

Các loại thao tác



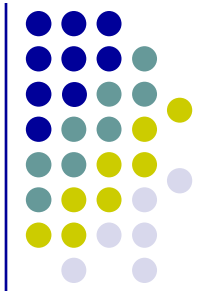
- Thao tác cài đặt (Implementor)
  - Cài đặt một vài chức năng nghiệp vụ
  - Hầu như mọi thông điệp trong biểu đồ tương tác ánh xạ vào thao tác cài đặt
- Thao tác quản lý (Manager)
  - Quản lý việc lập và hủy bỏ đối tượng
    - Thí dụ: các cấu tử, hủy tử của lớp
- Thao tác xâm nhập (Access)
  - Thao tác xâm nhập vào các thuộc tính private và protected
    - Thí dụ: các thao tác Get và Set cho mỗi thuộc tính trong lớp
- Thao tác trợ giúp (Helper)
  - Là các thao tác private và protected của lớp
  - Các thông điệp phản thân trong biểu đồ tương tác ánh xạ đến thao tác này

Quan hệ giữa các lớp



- Quan hệ là kết nối ngữ nghĩa giữa các lớp
  - Quan hệ cho một lớp biết thuộc tính, thao tác và quan hệ của lớp khác
- Các loại quan hệ chính
  - Kết hợp (Associations)
  - Thành phần (Aggregations)
  - Tổng quát hóa (Generalizations)
  - Phụ thuộc (Dependencies)

Tìm kiếm quan hệ

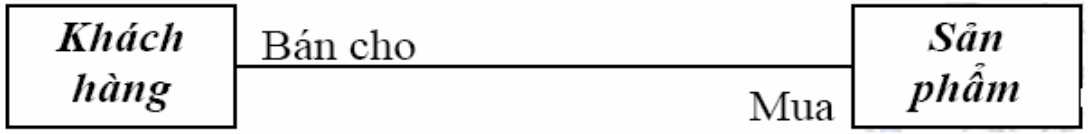


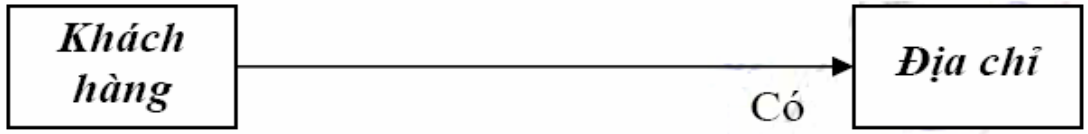
- Khảo sát biểu đồ trình tự và biểu đồ cộng tác
  - Nếu lớp A gửi thông điệp đến lớp B thì giữa chúng có quan hệ
    - Thông thường là quan hệ kết hợp hay phụ thuộc
- Khảo sát các lớp để tìm ra các quan hệ
  - Quan hệ tổng thể - thành phần
    - Bất kỳ lớp nào được hình thành từ lớp khác thì chúng có quan hệ tập hợp
  - Quan hệ tổng quát hóa
    - Nếu nhiều lớp kế thừa từ lớp thứ ba thì giữa chúng với lớp thứ ba có quan hệ khái quát hóa

Quan hệ kết hợp

Kết hợp là quan hệ cấu trúc để mô tả tập liên kết (một liên kết là kết nối giữa các đối tượng). Khi đối tượng của lớp này gửi/nhận thông điệp đến/từ đối tượng của lớp kia thì ta gọi chúng là có quan hệ kết hợp.

Ký pháp đồ họa của kết hợp được mô tả trên hình sau, chúng có thể chứa tên nhiệm vụ và tính nhiều (multiplicity).

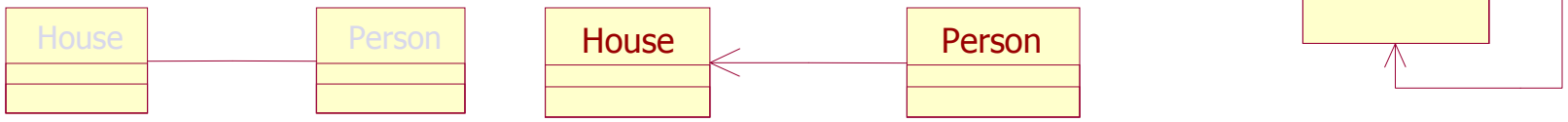




Quan hệ kết hợp cũng có thể có dạng một chiều.

Quan hệ kết hợp

- Association là kết nối ngữ nghĩa giữa các lớp
  - Kết hợp cho một lớp biết về thuộc tính và thao tác public của lớp khác
  - Quan hệ kết hợp hai chiều, một chiều
  - Quan hệ kết hợp phản thân



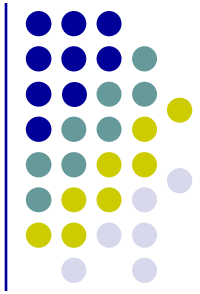

```
class Person
{
public:
Person();
~Person();
private:
House *the_House;
};
```

```
class House
{
public:
House();
~House();
private:
Person *the_Person;
};
```

```
class Person
{
public:
Person();
~Person();
private:
House *the_House;
};
```

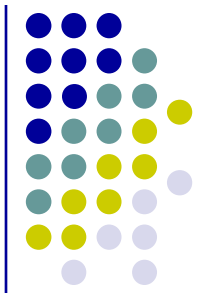
```
class House
{
public:
House();
~House();
private:
};
```

Quan hệ thành phần



- Aggregation là quan hệ giữa tổng thể và bộ phận (Whole-Parts)
  - Trong quan hệ này, một lớp biểu diễn cái lớn hơn còn lớp kia biểu diễn cái nhỏ hơn
  - Biểu diễn quan hệ has-a
    - Một đối tượng của lớp tổng thể có nhiều đối tượng của lớp thành phần
  - Có 2 loại quan hệ thành phần:
    - Tụ hợp
    - Gộp

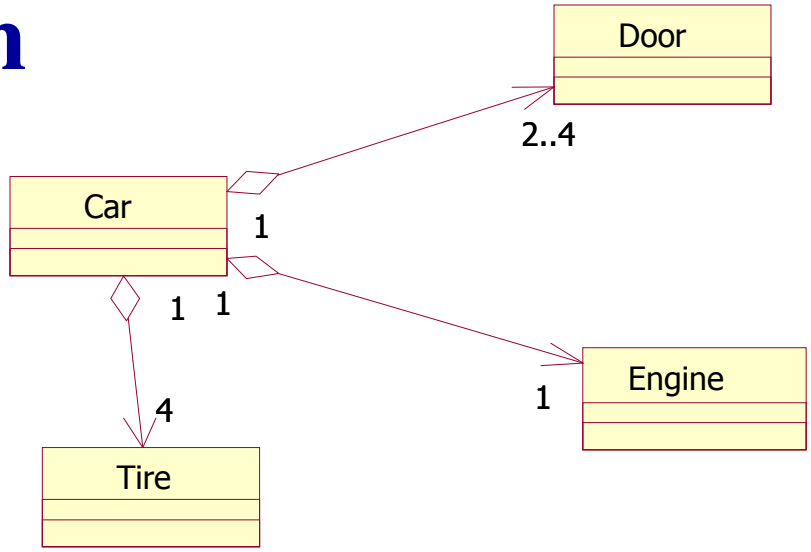
Quan hệ thành phần



- Tụ hợp:
  - Tổng thể và bộ phận có thể hủy bỏ vào thời điểm khác nhau
  - Tên khác: quan hệ tụ hợp bởi tham chiếu (by reference)
- Ví dụ: Một lớp học phần được mở (trong học tín chỉ) có nhiều sinh viên theo học, lớp học phần được mở là class tổng thể, còn sinh viên là class thành phần. Nếu không có lớp học phần được mở thì sinh viên vẫn tồn tại, hoặc khi hủy lớp học phần được mở đi thì sinh viên vẫn không bị hủy.

Quan hệ thành phần

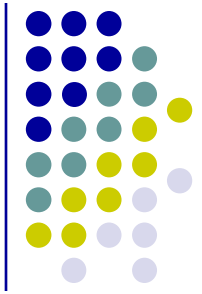
- Ví dụ:




```
#include "car.h"
class Door {
.....
private:
Car *the_car;
};

#include "door.h"
class Car {
...
private:
Door *the_Door;
};
```

Quan hệ thành phần



- Gộp: là dạng đặc biệt (mạnh hơn) của quan hệ tụ hợp
  - Tổng thể và thành phần được hình thành hay hủy bỏ vào cùng thời điểm
  - Tên khác: quan hệ tụ hợp bởi giá trị (by value)
- Ví dụ: Một công ty (Company) có nhiều phòng ban(Department). Như thế khi công ty bị hủy đi thì phòng ban (Department) không còn tồn tại, hoặc Một phòng học(Class-room) có nhiều bàn ghế (table), thực tế là tất cả các phòng học đều có bàn ghế riêng cho từng phòng và bàn của phòng nào đều có mã số phòng đi kèm với mã số bàn, như vậy nếu phòng học bị hủy thì bàn ghể của phòng đó bị hủy theo.

Quan hệ thành phần

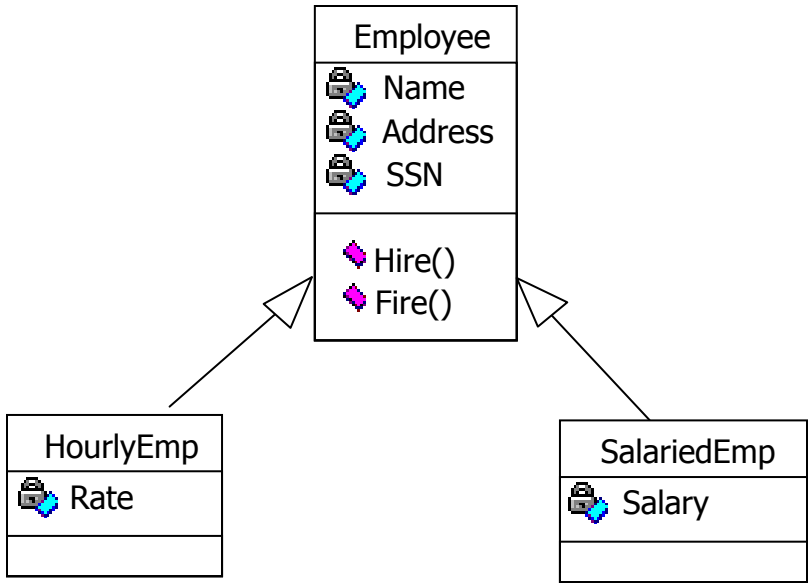
- Ví dụ:


```
#include "Window.h"
class Frame
{
...
private:
}
```

```
#include "Frame.h"
class Window
{
...
private:
Frame the_Frame;
}
```

Quan hệ tổng quát hóa

- Generalization là quan hệ kế thừa của hai phần tử mô hình như lớp, tác nhân, Use case và gói
  - Cho phép một lớp kế thừa các thuộc tính, thao tác public và protected của lớp khác



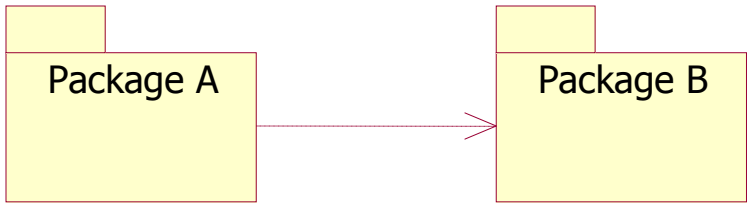

```
#include "Employee.h"
class HourlyEmp: public
Employee
{
private:
   float Rate;
.....
};
```

Quan hệ phụ thuộc

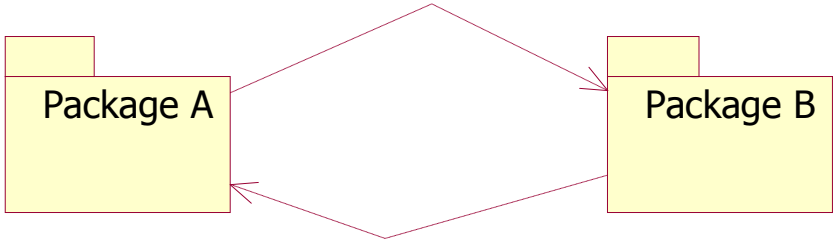
- Dependency là quan hệ chỉ ra một lớp tham chiếu lớp khác. Phụ thuộc là mối quan hệ giữa hai lớp đối tượng: một lớp đối tượng A có tính độc lập và một lớp đối tượng B phụ thuộc vào A; một sự thay đổi của A sẽ ảnh hưởng đến lớp phụ thuộc B.
- Khi thay đổi đặc tả lớp tham chiếu thì lớp sử dụng nó bị ảnh hưởng

Quan hệ phụ thuộc gói

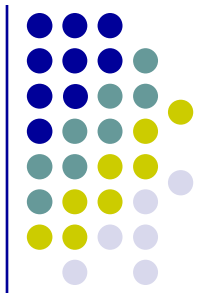
- Có thể vẽ quan hệ phụ thuộc giữa các gói như giữa các lớp
- Phụ thuộc gói từ gói A đến gói B có nghĩa rằng vài gói trong lớp A có quan hệ một chiều với các lớp trong gói B



- Tránh phụ thuộc vòng giữa các gói

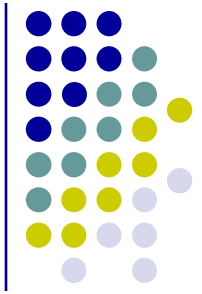


Đặc tả quan hệ giữa các lớp



- Đặc tả chi tiết quan hệ bao gồm
  - Multiplicity
  - Tên quan hệ
  - Tên nhiệm vụ
  - Phần tử liên kết

Đặc tả quan hệ giữa các lớp



- Đặc tả chi tiết quan hệ bao gồm
  - Multiplicity
    - Cho biết bao nhiêu hiện thực của một lớp liên kết với một hiện thực của lớp khác vào cùng thời điểm

  - Tên quan hệ
    - Tên quan hệ là động từ mô tả tại sao lại tồn tại quan hệ

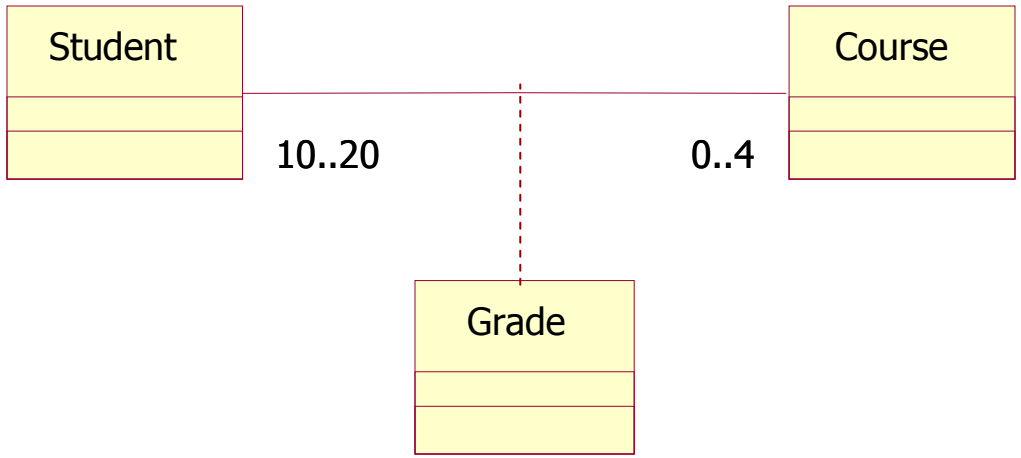
  - ...

Đặc tả quan hệ giữa các lớp

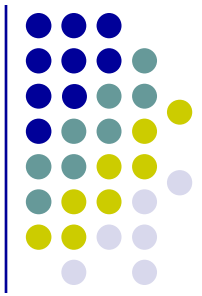
- Đặc tả chi tiết quan hệ bao gồm
  - .....
  - Tên nhiệm vụ
    - Sử dụng tên nhiệm vụ thay thế cho tên quan hệ trong quan hệ kết hợp hay tụ hợp để chỉ ra tại sao quan hệ tồn tại

Đặc tả quan hệ giữa các lớp

- Đặc tả chi tiết quan hệ bao gồm
  - .....
  - Phạm vi kết hợp (Qualifier)
  - Phần tử liên kết
    - Còn gọi là lớp kết hợp, nơi lưu trữ thuộc tính liên quan đến kết hợp



Tóm tắt



- Bài này đã xem xét các vấn đề sau
  - Tìm kiếm lớp
  - Tìm kiếm thuộc tính, thao tác lớp
  - Tìm kiếm các loại quan hệ giữa các lớp
  - Biểu diễn biểu đồ lớp và gói
  - Biểu diễn đồ họa các thuộc tính của thuộc tính, thao tác trong lớp
  - Biểu diễn các thuộc tính cho quan hệ giữa các lớp