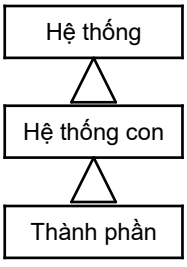
OOAD

Thiết kế hệ thống
(Phần II)

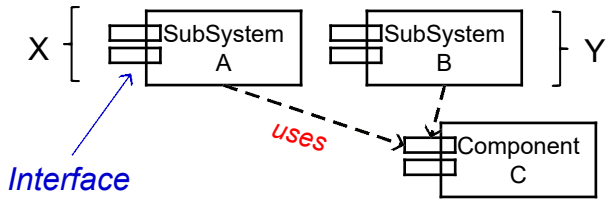
Nguyễn Anh Hào
0913609730 – nahao@ptithcm.edu.vn

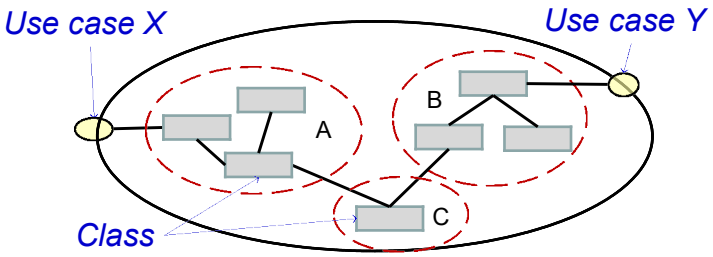
5) DESIGN SUBSYSTEM

- Hệ thống con: Là một nhóm các đối tượng hợp tác với nhau để thực hiện một số nhiệm vụ của hệ thống
  – Tham gia giải quyết một usecase
- Hệ thống con đại diện cho hệ thống cung cấp các dịch vụ (ra ngoài) thông qua giao diện của nó
- Thiết kế hệ thống con: không dành cho mục đích tái sử dụng (≠ Gói)



Hệ thống con





a) Từ phân tích đến thiết kế

1 Phân hoạch hệ thống thành các hệ thống con
- Mỗi hệ thống con sẽ xử lý một usecase trong lược đồ usecase
- Phân vùng (partitioning): Sử dụng các lược đồ giao tiếp cho usecase để chỉ ra các lớp ý niệm (được ánh xạ tới các lớp thiết kế) của hệ thống con.
- Phân tích nhân tử (factoring): Tách các đặc điểm tương tự của các lớp thành các lớp cơ sở (thư viện dùng chung)

Từ phân tích đến thiết kế

2 Thiết kế tương tác cho hệ thống con
- Chỉ có lớp biên có thể hiển thị ra bên ngoài. Tất cả các lớp bên trong của hệ thống con đều được ẩn (đối với bên ngoài).
- Kiểu dữ liệu của thông điệp cũng phải được khai báo với bên ngoài.

3 Ánh xạ các lớp vào ngôn ngữ để hiện thực (Java,..)
- Phương thức & thuộc tính (chứa dữ liệu cố định)

b) Interface : Vai trò

1 Bảo vệ hệ thống khỏi những tác động không mong muốn từ môi trường bên ngoài
- Kiểm tra an ninh
- Phát hiện lỗi

2 Lọc bỏ nội dung đầu vào/đầu ra không cần thiết
- Nguyên lý phân tách giao diện (4)

3 Mã hóa và giải mã nội dung đầu vào/đầu ra
- Tổng hợp và chuyển đổi dữ liệu sang các định dạng cần thiết cho hệ thống hoặc cho các đối tượng bên ngoài khác.

4 Lưu trữ tạm thời dữ liệu đầu vào/đầu ra bằng vùng đệm.

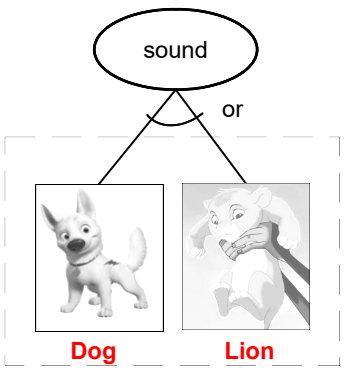
Interface : Thiết kế

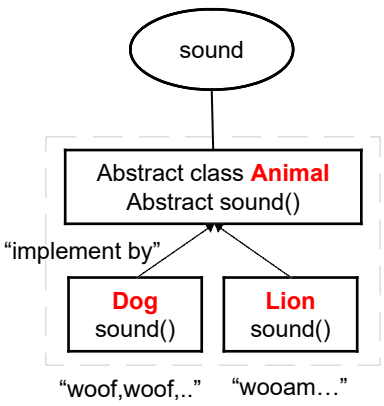
1 Chỉ có giao diện (lớp biên) là có thể được nhìn thấy từ các đối tượng bên ngoài; tất cả các lớp bên trong khác được định nghĩa là vô hình.
2 Lớp biên nên là lớp trừu tượng (abstract class). Hệ thống con có ít nhất 1 lớp cụ thể (concrete class) để hiện thực cho lớp biên.
3 Lớp biên chỉ cung cấp giao diện cố định.
4 Không có lớp bên trong nào phụ thuộc vào lớp biên
5 Kiểu dữ liệu của thông điệp được khai báo công khai cho các đối tượng bên ngoài.
6 Tuân thủ nguyên tắc SOLID .

User Interface Modelling with UML.pdf

Abstract interface

Giao diện trừu tượng: không chứa bất kỳ phương thức cụ thể nào (phương thức có mã). Nó chỉ chứa các phương thức trừu tượng: chúng không có phần thân; chúng chỉ có tên, tham số và kiểu trả về.





Lợi ích của giao diện trừu tượng

1. Tách biệt "cái gì" và "như thế nào"
   – Giao diện trừu tượng không chứa logic cài đặt, nó sẽ cho phép nhiều cách cài đặt (implement) khác nhau
2. Nhất quán trong cách ứng xử
   – Nếu một lớp có một giao diện trừu tượng, tất cả các cài đặt cho giao diện phải nhất quán — trình biên dịch sẽ đảm bảo điều này.
3. Cho phép đa hình
   – Các lớp khác nhau có thể hiện thực cho một phương thức bằng nhiều cách
4. Hỗ trợ dependency inversion (D trong SOLID)
   – Bằng cách cung cấp giao diện trừu tượng ra ngoài (không phụ thuộc vào một kiểu dữ liệu)

c) Messages

Thông điệp là nội dung dữ liệu được truyền giữa các đối tượng, do đó nội dung của nó cần phải được nơi gửi và nơi nhận hiểu rõ.

- Nội dung của thông điệp thường bị phụ thuộc vào cấu trúc dữ liệu của nó (kiểu, độ dài,...).
- Việc gửi và nhận thông điệp được nhiều lập trình viên xử lý và mã lệnh của họ phụ thuộc lẫn nhau do cấu trúc này.
- Để tránh sự phụ thuộc vào cấu trúc dữ liệu, thông điệp phải là một đối tượng chỉ cung cấp các phương thức (cố định) để xử lý cấu trúc dữ liệu ẩn bên trong của thông điệp.

Ví dụ về cách cài đặt thông điệp

Một cách cài đặt địa chỉ trong C++ như sau:

```
public char Addr[40];
```

Những khó khăn nào sẽ xảy ra khi phát triển phần mềm?

Các lập trình viên sử dụng cấu trúc này phải lưu ý:
1 Addr là chuỗi C, kết thúc bằng '\0', strlen ≤ 39 ký tự
2 Quy ước chung giữa các lập trình viên về cấu trúc:
"<số> <đường> <phường> <quận> <thành phố>" …

Làm thế nào để thay đổi cấu trúc của Addr mà không cần phải tuân thủ quá nhiều quy ước cho các chương trình liên quan?

Đối tượng thông điệp


```
class Address
{   private: // dữ liệu ẩn
    string num, street, district, city;
    public : // truy cập thông qua giao diện public
    string GetAddr (); string Getnum (), string GetStreet (); …
    string SetNum (chuỗi sn); string SetStreet (chuỗi st);
….
};
```

1 Các phương thức công khai và các thuộc tính riêng tư chỉ được quyết định bởi người lập trình viên tạo ra thông điệp này.
2 Các chương trình khác truy cập dữ liệu của thông điệp bằng các phương thức công khai

Abstract Data Type

- Kiểu dữ liệu trừu tượng là kiểu dữ liệu mà cấu trúc cài đặt của nó bị ẩn, nó chỉ định nghĩa những hoạt động nào cần thực hiện chứ không đề cập đến cách thức thực hiện những hoạt động này.
  - Nó định nghĩa một giao diện chung cho các lớp cụ thể của nó.
  - Nó không chỉ rõ dữ liệu sẽ được sắp xếp như thế nào và thuật toán nào sẽ được sử dụng để thực hiện các hoạt động.

- Nó cung cấp một cái nhìn độc lập về việc thực hiện, để
  - Cho phép tái sử dụng mã
  - Tạo nhiều phiên bản cài đặt khác nhau

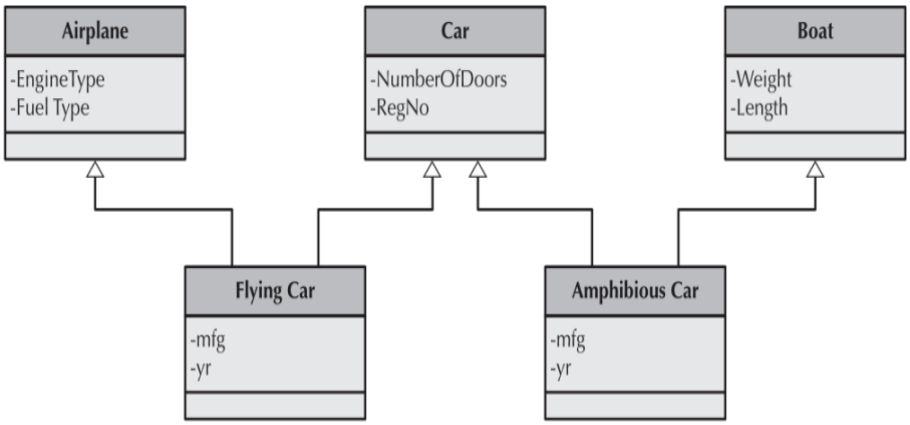
d) Lưu dữ liệu vào cơ sở dữ liệu

CÁC LOẠI CƠ SỞ DỮ LIỆU

- Hệ thống quản lý CSDL quan hệ (RDBMS)
  - Chỉ hỗ trợ tính toàn vẹn tham chiếu trên dữ liệu đơn giản
  - Không thể xử lý dữ liệu phức tạp, không hỗ trợ OO
- Hệ thống quản lý CSDL quan hệ đối tượng (ORDBMS)
  - ORDBMS là RDBMS có phần mở rộng để xử lý việc lưu trữ các đối tượng trong cấu trúc bảng quan hệ.
  - Các ORDBMS trên thị trường vẫn chưa hỗ trợ tất cả các tính năng hướng đối tượng (ví dụ: **kế thừa**)
- Hệ thống quản lý CSDL hướng đối tượng (OODBMS)
  - Có khả năng xử lý dữ liệu phức tạp, hỗ trợ trực tiếp OO
  - Công nghệ vẫn đang phát triển
- Kho dữ liệu NoSQL (đa dạng hóa kiểu dữ liệu)
  - Có khả năng xử lý dữ liệu phức tạp
  - Công nghệ là vẫn đang trưởng thành, không theo RDBMS

Ví dụ: Lưu quan hệ đa kế thừa

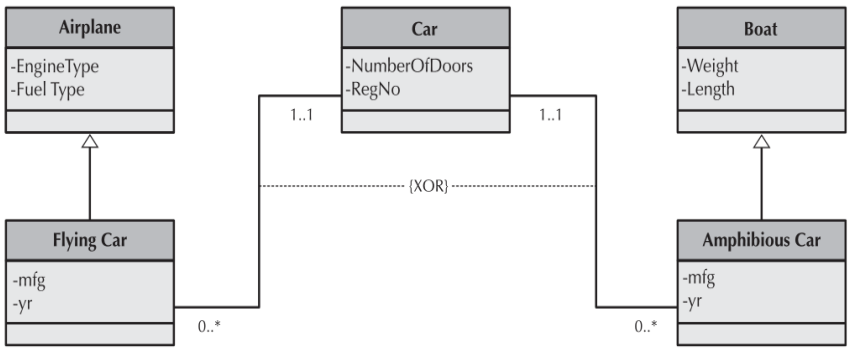
Ví dụ đơn giản về đa kế thừa



(a)

ORDBMS đa kế thừa: QUY TẮC 1a

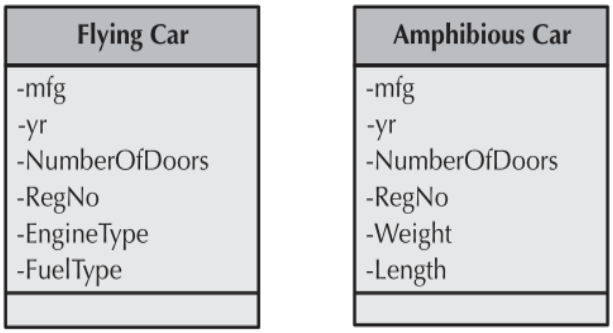
Chuyển đổi các mối quan hệ kế thừa thành các mối quan hệ liên kết . Liên kết từ lớp con đến lớp cha phải là 1..1, hoặc 1..0



(b)

RDBMS đa kế thừa: QUY TẮC 1b

Làm phẳng hệ thống kế thừa bằng cách sao chép các thuộc tính và phương thức của lớp cơ sở bổ sung xuống tất cả các lớp con và loại bỏ lớp cơ sở ra khỏi thiết kế (cần chuẩn hóa lại)



(c)

e) Domain Objects --> RDBMS

1 Lớp cụ thể được ánh xạ vào bảng trong RDBMS.
2 Thuộc tính đơn trị được ánh xạ vào cột trong bảng RDBMS.
3 Đảm bảo rằng khóa chính của lớp con giống như khóa chính của lớp cha.
4 Các phương thức được ánh xạ tới các stored procedure hoặc các mô-đun chương trình
5 Kết tập và liên kết có giá trị đơn (một-một) được ánh xạ tới các cột có thể lưu trữ khóa ngoại của các bảng liên quan.
6 Thuộc tính đa trị được ánh xạ vào các bảng mới và tạo ra mối quan hệ một-nhiều
7 Các mối quan hệ kết tập và liên kết đa giá trị được ánh xạ vào các bảng mới và tạo ra một bảng liên kết

Systems Analysis and Design An Object-Oriented Approach with UML (p.337)