Phân tích thiết kế hướng đối tượng

Bài 2: Một số khái niệm cơ bản trong OOP

TS. Nguyễn Hiếu Cường
Bộ môn CNPM, Khoa CNTT
Trường ĐH GTVT
cuonggt@gmail.com

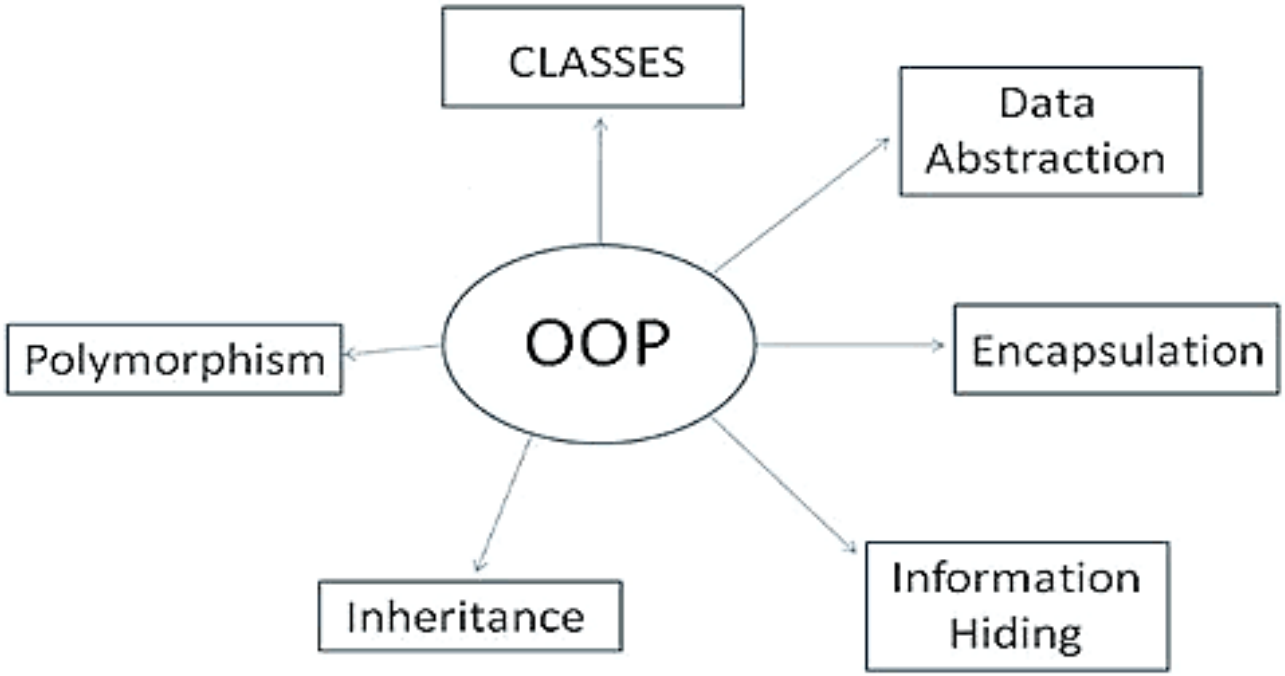
Hà Nội, 2019

Nội dung chính

- Giới thiệu về phát triển hệ thống, mô hình hóa, UML
- **Các khái niệm cơ bản về hướng đối tượng**
- Quy trình phát triển phần mềm
- Khảo sát, xác định yêu cầu
- Các loại mô hình hóa: tĩnh và động
- Các biểu đồ UML: lớp, trình tự, hành động, trạng thái…
- Thiết kế, các qui tắc và mẫu thiết kế
- . . .

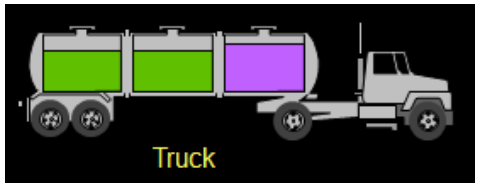
Các khái niệm cơ bản trong OOP

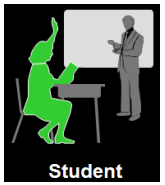
- Khái niệm trung tâm là **đối tượng** (object) và **lớp** (class)

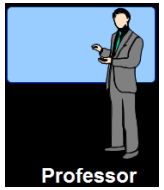


Đối tượng

- Thế giới thực bao gồm các *đối tượng* (object)!
  - Đối tượng vật lý

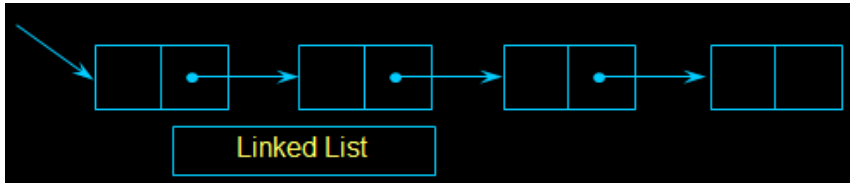






  - Đối tượng khái niệm

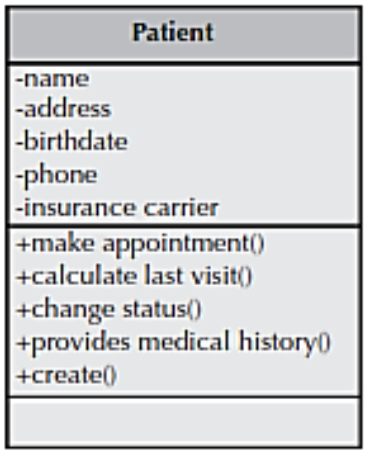
  - Đối tượng phần mềm



- Mỗi đối tượng gồm các *thuộc tính* và các *thao tác*.

Lớp

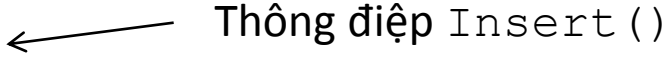
- Lớp (Class)
  - Định nghĩa trừu tượng của các đối tượng có cùng những đặc tính chung
  - Đối tượng (object): thể hiện cụ thể (instance) của một lớp
- Tác dụng của lớp?
  - Trừu tượng hoá dữ liệu
  - Bao gói
  - Che giấu thông tin
  - ...



Thông điệp và truyền thông điệp

- Thông điệp (message)
  - Kích hoạt một hành động của của đối tượng

```
Patient aPatient;
aPatient.Insert();
```

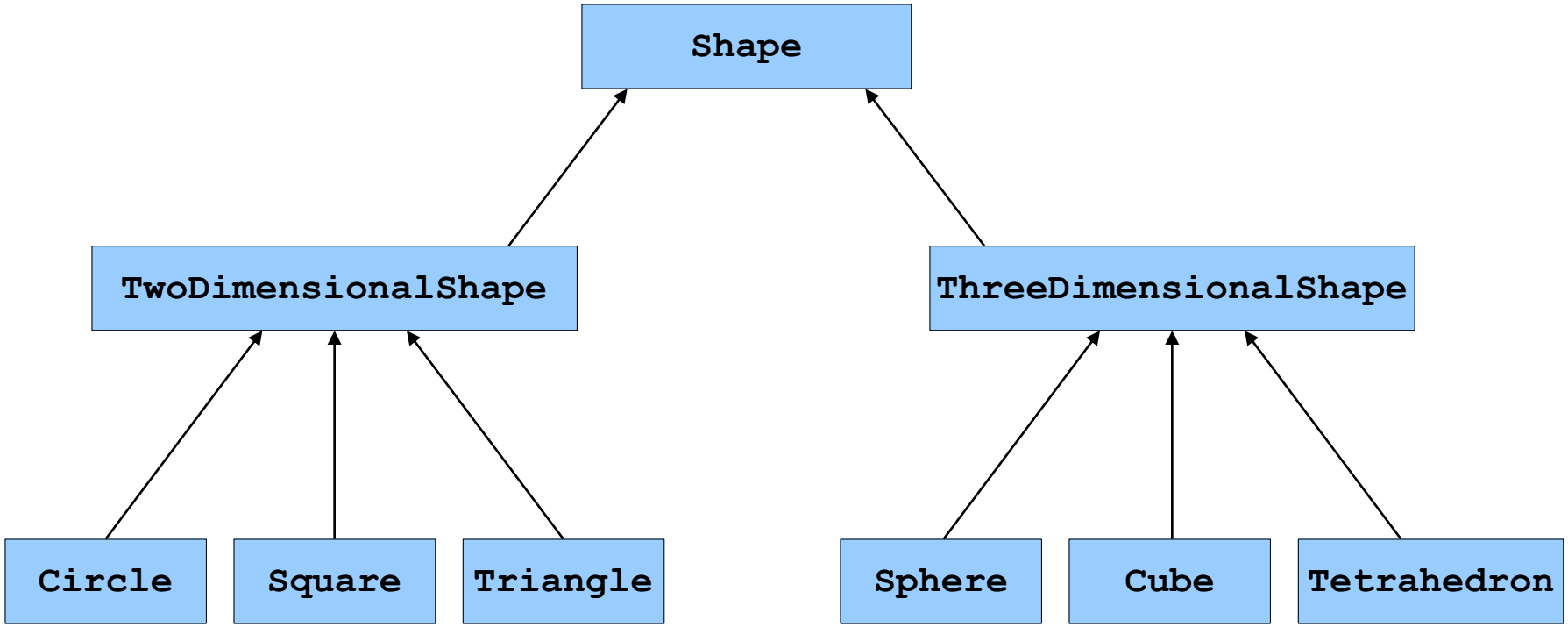


- Phương thức ≠ thông điệp?
  - Phương thức (method) là hàm thành phần của lớp (member function)
  - Thông điệp là lời gọi hàm của một đối tượng của lớp đó

Kế thừa

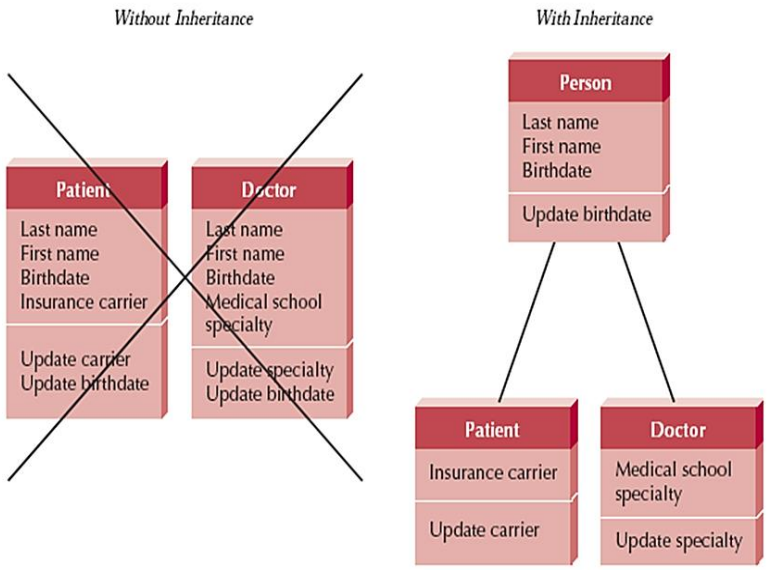
- Mục đích
  - Xây dựng một lớp mới bằng cách kế thừa các lớp đã có
- Một lớp có thể
  - Được thừa kế từ một hoặc nhiều lớp khác
  - Là cơ sở của một hoặc nhiều lớp khác
- Những gì được kế thừa?
  - Dữ liệu
  - Các phương thức

Ví dụ


Shape
TwoDimensionalShape
ThreeDimensionalShape
Circle
Square
Triangle
Sphere
Cube
Tetrahedron

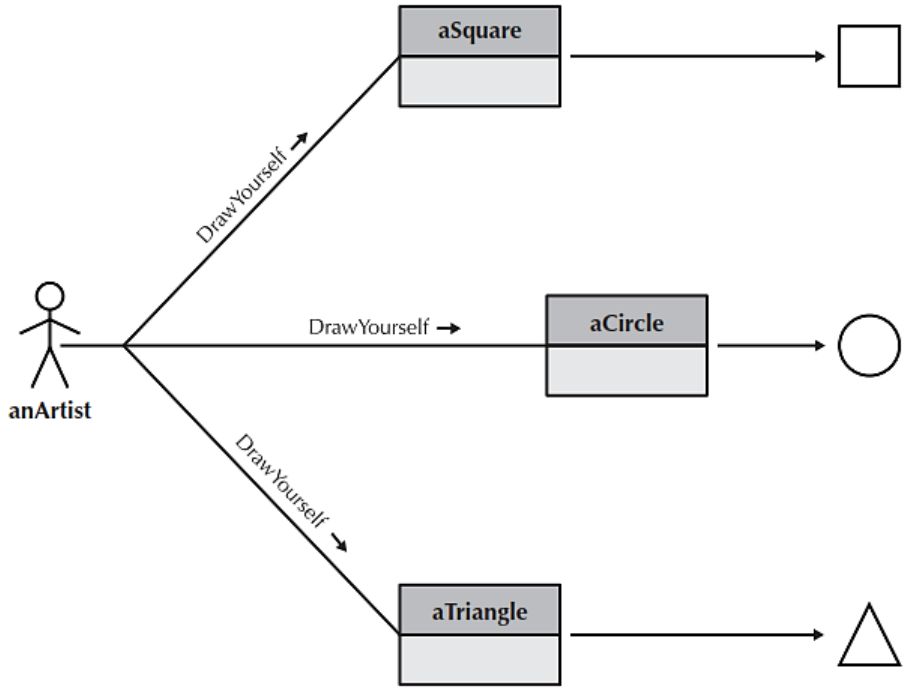
Tác dụng của kế thừa

- Sử dụng lại
- Phát triển chương trình
- Thiết kế hiệu quả hơn
  - ***So sánh 2 thiết kế bên***



Tương ứng bội

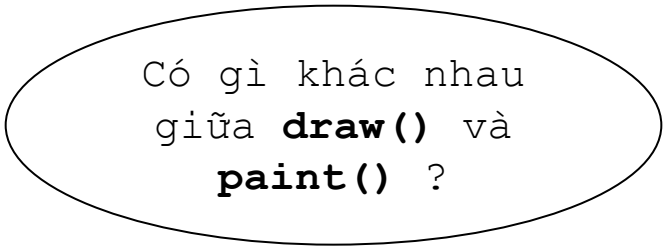
- Tương ứng bội hoặc tính đa hình (polymorphism)
  - Một thông điệp có thể được diễn giải theo các cách khác nhau
  - Có thể ẩn các chi tiết cài đặt dưới một giao diện chung



Ví dụ về hàm ảo

```
class Shape {
public:
  virtual void draw()
  {
    cout<<"draw shape \n";
  }
  void paint()
    {
    cout<<"paint shape \n";
  }
};
class Square: public Shape {
public:
  void draw(){cout<<"draw square \n";}
  void paint(){cout<<"paint square"; }
};
class Circle: public Shape {
public:
  void draw() {cout<<"draw circle";}
  void paint() {cout<<"paint circle";}
};
```

```
void main()
{
  Shape *ptr;
  Shape p; //OK
  Circle c;
  Square s;
  ptr= &c;
  ptr->draw();    // draw circle
  ptr->paint();   // paint shape
  ptr= &s;
  ptr->draw();    // draw square
  ptr->paint();   // paint shape
}
```



Ví dụ về hàm ảo

```
#include <iostream>
using namespace std;
class A {
public:
     A() { cout << "A constructor\n"; }
     void m1() { cout << "A.m1\n"; m2();  }
     virtual void m2() { cout << "A.m2\n"; }
 };
 class B : public A     {
 public:
     B() { cout << "B constructor\n";}
     void m1() { cout << "B.m1\n"; }
     void m2() { cout << "B.m2\n"; }
  };
  void func(A &a) { a.m1(); }
  int main() {
     B b;
     func(b);
  }
```

```
A constructor
B constructor
A.m1
B.m2
```

Question

Object technology is . . . ?

A. A set of principles guiding software construction.
B. A new theory striving to gain acceptance.
C. A dynamic new language by Grady Booch.
D. Based on the principles of abstraction and modularity.

Question

A model . . .?

A. Is not necessary when team members understand their job.
B. Has to be structural AND behavioral.
C. Is a simplification of reality.
D. Is an excuse for building an elaborate plan.

Question

Why do we model?

A. Helps to visualize a system
B. Gives us a template for constructing a system
C. Documents our decisions
D. All of the above

Question

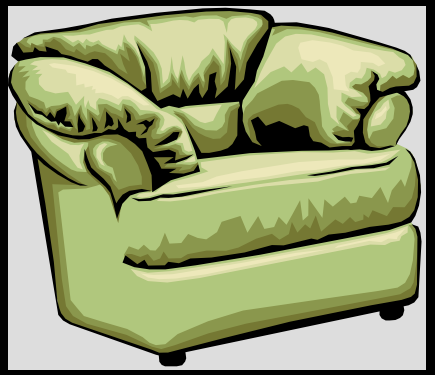
The best models are connected to . . .?

A. Java-script code
B. Reality
C. C ++
D. Issues that tie it to an object-oriented developer

Question

Which project would be least likely to require a model?

A. 

B. 

C. 

D. 

Question

Which principles of modeling are correct?

A. The model you create, influences how the problem is attacked.
B. The best kinds of models are those that let you chose your degree of detail.
C. The best models are connected to reality.
D. Create models that are built and studied separately.

Question

Encapsulation . . . ?

A. Allows direct manipulation of things that have been encapsulated.
B. Is often referred to as information hiding.
C. Causes costly and extensive maintenance.
D. Causes changes to affect clients during implementation.

Question

What happens when you incorporate modularity into your plan?

A. It reduces something complex into manageable pieces.
B. It builds modules that talk to each other.
C. Creates systems too large to understand.
D. Parts of your system cannot be independently developed.

Question

A class . . . ?

A. Is an encapsulation of an object.
B. Represents the hierarchy of an object.
C. Is an instance of an object.
D. Is an abstract definition of an object.

Question

Polymorphism can be described as?

A. Hiding many different implementations behind one interface
B. Inheritance
C. Information placing
D. Generalization