Các nguyên lý lập trình hướng đối tượng

Nguyễn Minh Huy
Bộ môn Công nghệ Phần mềm
Email: *nmhuy@fit.hcmuns.edu.vn*

Nội dung

- Phần mềm hướng đối tượng
- Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

Nội dung

- Phần mềm hướng đối tượng
- Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

Phần mềm hướng đối tượng

- Phần mềm là gì?
  - "A computer program, enable a computer to perform a specific task" (wikipedia).
  - ess a in a code the same way our words and sentences code our thoughts... The code is only the representation of the ideas, and the ideas are really the software" (*Hardware is from Mars; Software is from Venus*, Winn Rosch).

Software is something that is… soft!!

Phần mềm hướng đối tượng

- Thế nào là phần mềm hướng đối tượng?

Chưa đủ!

...ây??

Tuân thủ nguyên lý lập trình hướng đối tượng thông qua việc vận dụng các tính chất lập trình hướng đối tượng

Phần mềm hướng đối tượng

- Ba tín… ợng:
  - Tí…
  - Tí…
  - Tí…
- Nguyê… ng:
  - Nh… ền tảng.
  - Có …
  - Là… hướng đối tư…

Nguyên lý Open-Close
(The Open-Closed Principle)

Nguyên lý Nghịch đảo phụ thuộc
(The Dependency Inversion Principle)

Nguyên lý Thay thế Liskov
(The Liskov Substitution Principle)

Nguyên lý Phân tách Interface
(The Interface Segregation Principle)

Nội dung

- Phần mềm hướng đối tượng
  - Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

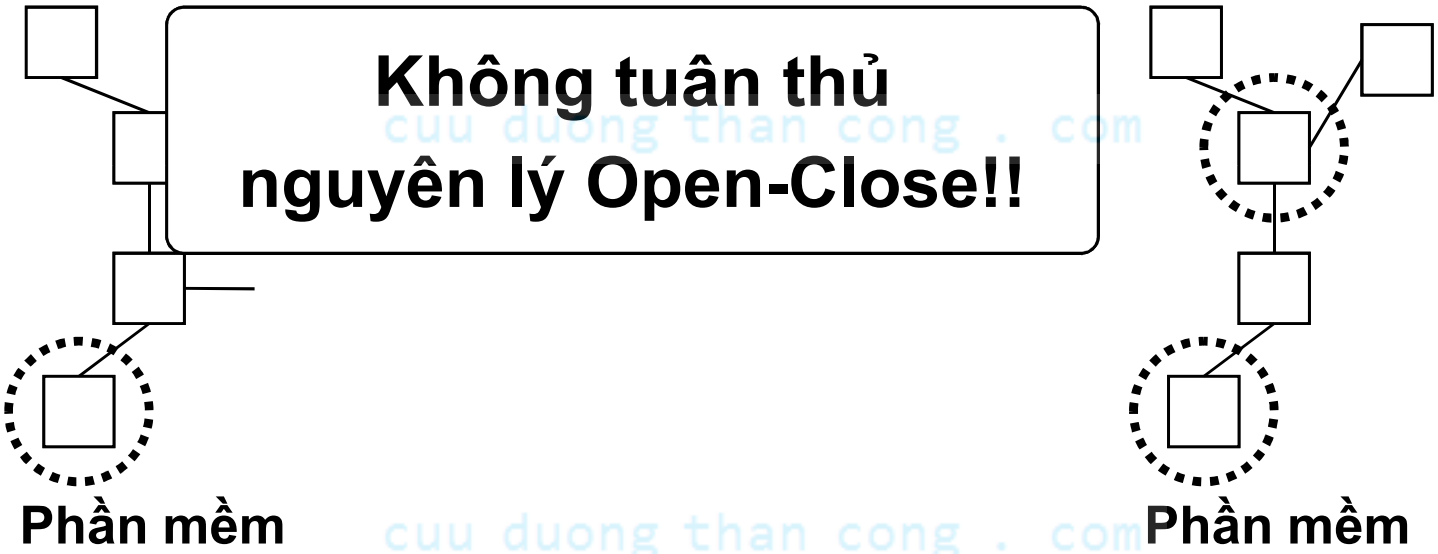
Nguyên lý Open-Close

- Betrand Meyers đề cập lần đầu tiên năm 1988 trong Object Oriented Software Construction.

- Phát biểu:

*"Các thực thể phần mềm (hàm, đơn thể, đối tượng, …) nên được xây dựng theo hướng mở cho việc mở rộng (**be opened for extension**) nhưng đóng đối với việc sửa đổi (**be closed for modification**)".*

Nguyên lý Open-Close



Nguyên lý Open-Close

- Ví dụ chương trình vẽ hình:

```
public enum ShapeType { LINE, RECTANGLE }

public abstract class Shape
{
    public abstract ShapeType getType();
}

public class Line: Shape
{
    public override ShapeType getType();
    public void drawLine();
}

public class Rectangle: Shape
{
    public override ShapeType getType();
    public void drawRectangle();
}
```

Nguyên lý Open-Close

```
public void draw(ArrayList shapeList)
{
    Line        line;
    Rectangle   rectangle;

    foreach (Shape s in shapeList)
        switch (s.getType())
        {
            case ShapeType.LINE:
                line = (Line)s;
                line.drawLine();
                break;

            case ShapeType.RECTANGLE:
                rectangle = (Rectangle)s;
                rectangle.drawRectangle();
                break;
        }
}
```

Nguyên lý Open-Close

```
public abstract class Shape
{
    public abstract void draw();
}

public class Line: Shape
{
    public override void draw();
}

public class Rectangle: Shape
{
    public override void draw();
}
```

Nguyên lý Open-Close

```
public void draw(ArrayList shapeList)
{
    foreach (Shape s in shapeList)
        s.draw();
}
```


cuu duong than cong . com

Nguyên lý Open-Close

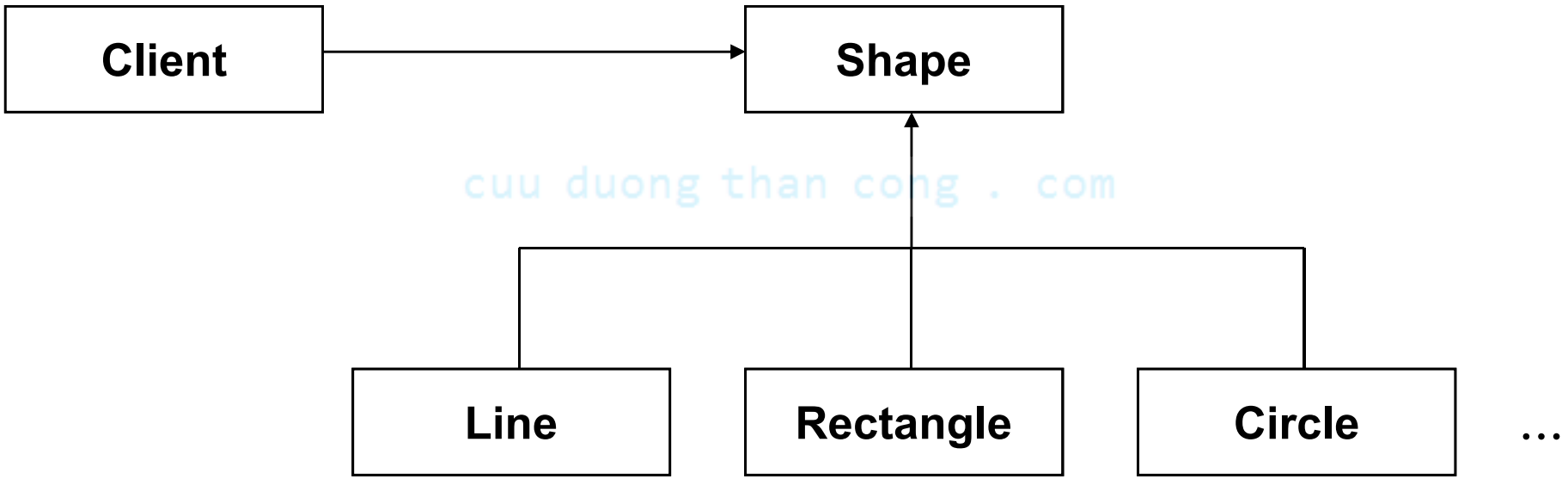
```
public abstract class Shape
{
    public abstract void draw();
}

public class Line: Shape
{
    public override void draw();
}

public class Rectangle: Shape
{
    public override void draw();
}

public class Circle: Shape
{
    public override void draw();
}
```

Nguyên lý Open-Close



Nguyên lý Open-Close

- Ghi chú:
  - Nguyên lý cốt lõi của phân tích thiết kế hướng đối tượng.
  - Ưu tiên áp dụng nguyên lý cho các thực thể phần mềm phải thường xuyên nâng cấp, mở rộng.
  - Việc tuân thủ nguyên lý mang tính tương đối, phụ thuộc ngữ cảnh.

Nguyên lý Open-Close

- Áp dụng:
  - Thuộc tính của đối tượng là private để hạn chế sự kết dính không cần thiết (coupling).

Đối tượng nắm giữ thông tin và
chịu trách nhiệm trên thông tin mình nắm giữ!!

  - Hạn chế ép kiểu động (runtime type-casting).

```
public void doSomething(Vehicle vehicle)
{
   Car     car = (Car)vehicle;
   car.run();
   car.stop();
}
```

Nội dung

- Phần mềm hướng đối tượng
- Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

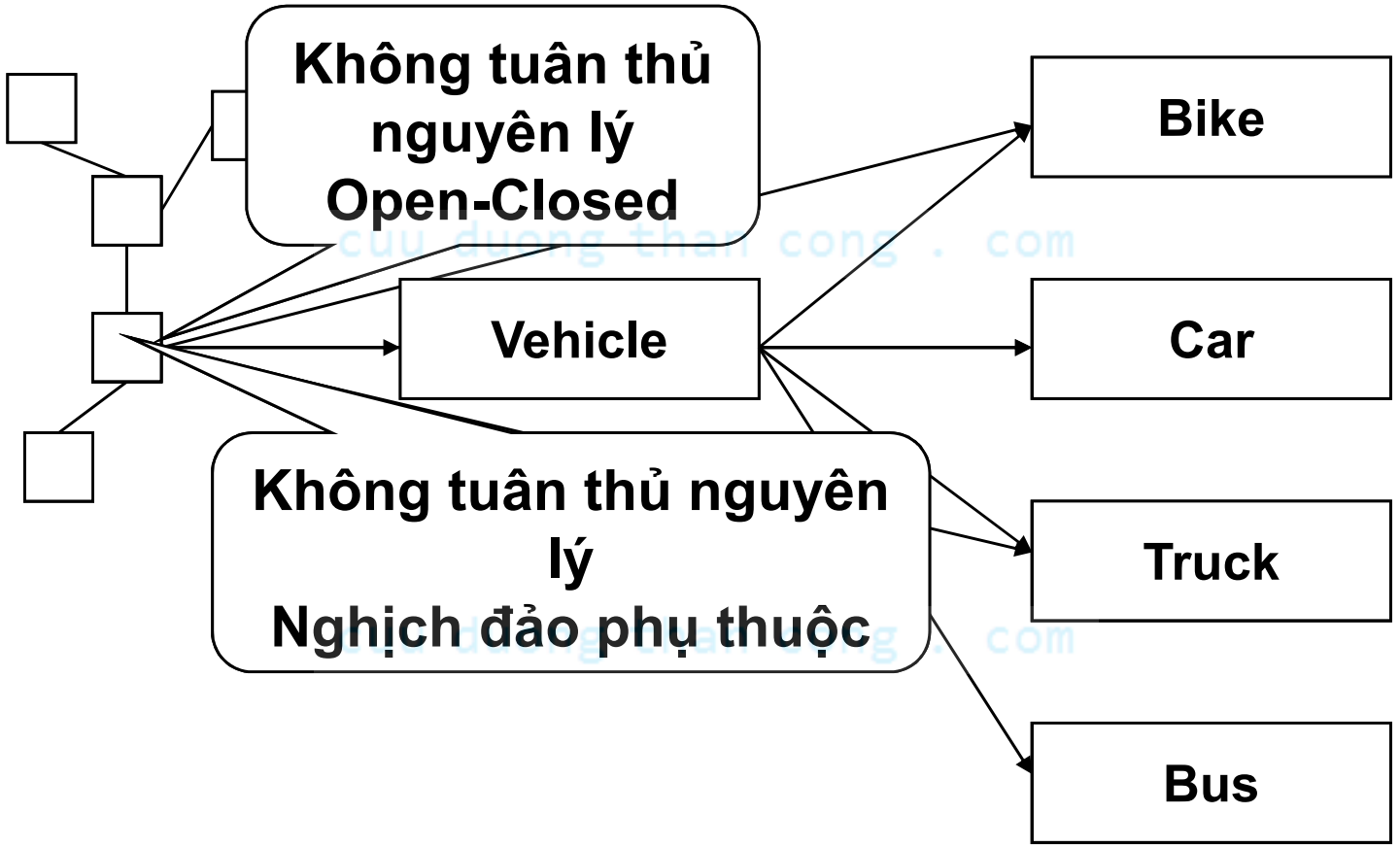
Nguyên lý Nghịch đảo phụ thuộc

- Phát biểu:

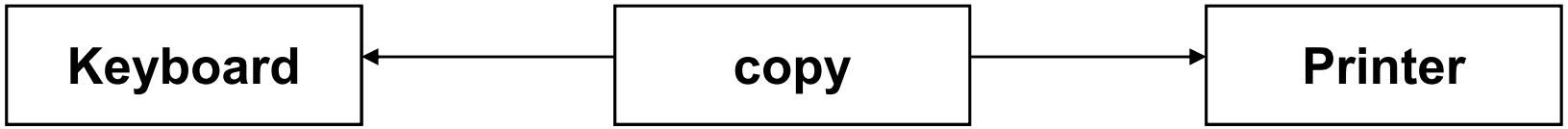
  "*Các thành phần trong phần mềm không nên phụ thuộc vào những cái riêng, cụ thể (**details**) mà ngược lại nên phụ thuộc vào những cái chung, tổng quát (**abstractions**) của những cái riêng, cụ thể đó.*

  Những cái chung, tổng quát không nên phụ vào những cái riêng, cụ thể. Sự phụ thuộc này nên được đảo ngược lại."

Nguyên lý Nghịch đảo phụ thuộc



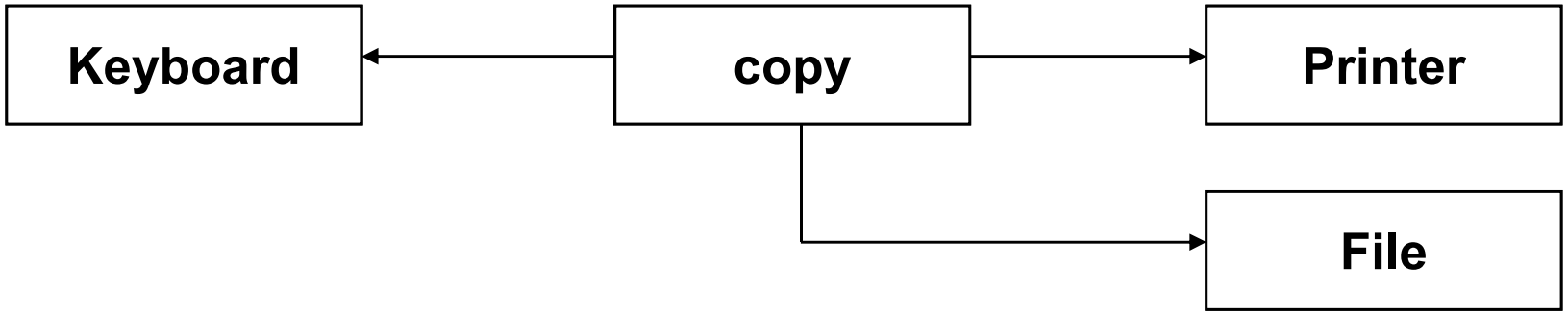
Nguyên lý Nghịch đảo phụ thuộc – Ví dụ




```
public void copy()
{
    Keyboard        keyboard = new Keyboard();
    Printer         printer = new Printer();
    char            c;

    while ((c = keyboard.read()) != ‘q’)
            printer.write(c);
}
```

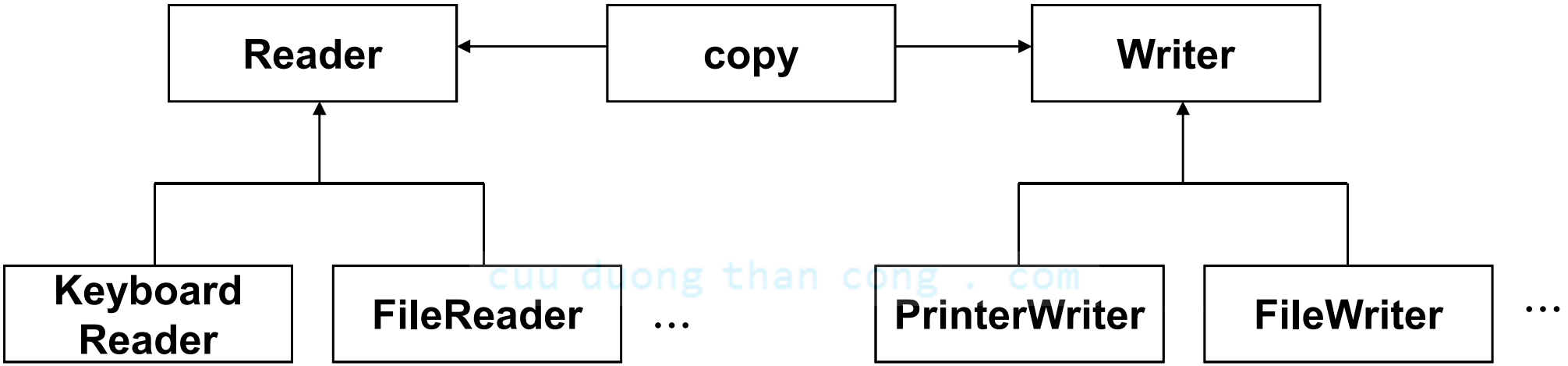
Nguyên lý Nghịch đảo phụ thuộc – Ví dụ




```
public void copy(OutputType type)
{
    Keyboard        keyboard = new Keyboard();
    Printer         printer = new Printer();
    File            file = new File();
    char            c;

    while ((c = keyboard.read()) != 'q')
        if (type == OutputType.PRINTER)
            printer.write(c);
        else if (type == OutputType.FILE)
            file.write(c);
}
```

Nguyên lý Nghịch đảo phụ thuộc – Ví dụ




```
public void copy(Reader reader, Writer writer)
{
    char            c;

    while ((c = reader.read()) != 'q')
        writer.write(c);
}
```

Nguyên lý Nghịch đảo phụ thuộc – Ghi chú

- Ghi chú:
  - Có mối liên hệ mật thiết với nguyên lý Open-Close.
  - "Chia để trị" theo hướng lập trình cấu trúc dễ vi phạm nguyên lý.
  - Thực thể phần mềm vi phạm nguyên lý có tính tái sử dụng không cao.
  - Allen Holub: *"The more abstraction you add, the greater the flexibility. In today's business environment, where requirements regularly change as program develops, this flexibility is essential."*

Nguyên lý Nghịch đảo phụ thuộc – Áp dụng

- Áp dụng:
  - Làm việc với lớp cơ sở thay vì lớp kế thừa cụ thể.

```
public void doSomething(Car car)
{
   car.run();
   car.stop();
}

public void doSomething(Vehicle vehicle)
{
   vehicle.run();
   vehicle.stop();
}
```

Nội dung

- Phần mềm hướng đối tượng
- Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

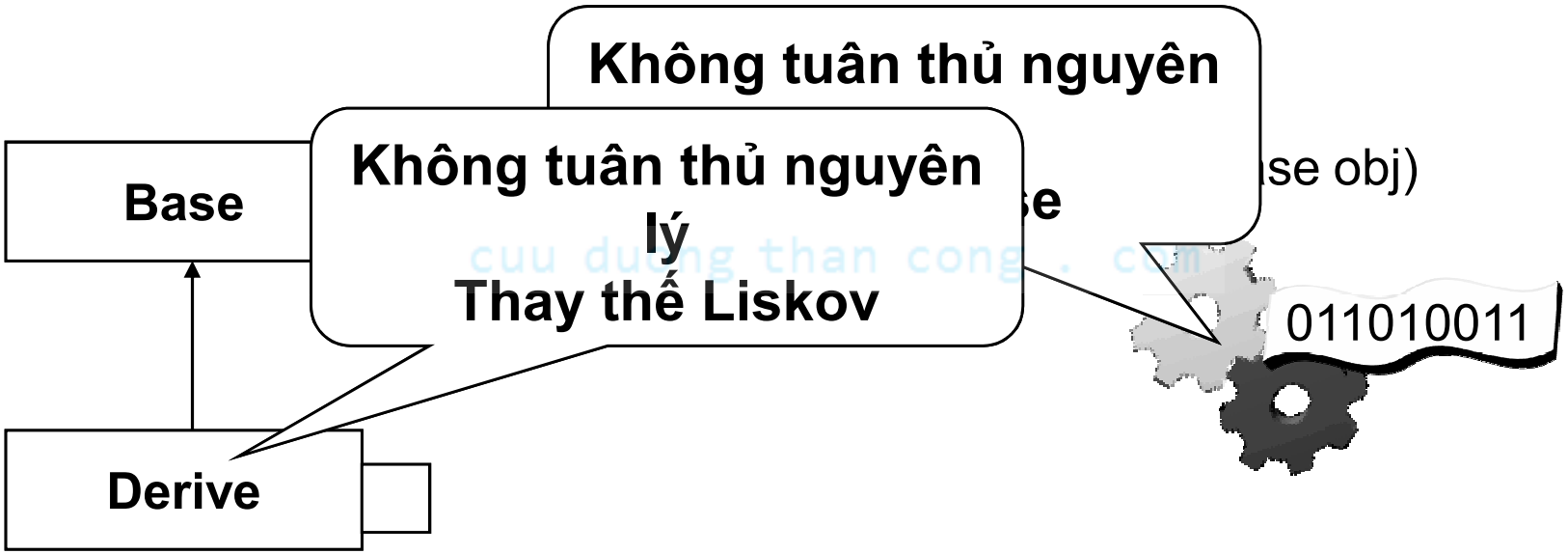
Nguyên lý Thay thế Liskov

- Barbara Liskov đề cập lần đầu tiên tại OOPSLA 87.

- Phát biểu:

*"Lớp B chỉ nên kế thừa từ lớp A khi và chỉ khi với mọi hàm F thao tác trên các đối tượng của A, cách cư xử (**behaviors**) của F không thay đổi khi ta thay thế (**substitute**) các đối tượng của A bằng các đối tượng của B."*

Nguyên lý Thay thế Liskov



Nguyên lý Thay thế Liskov

- Ví dụ về hậu quả kế thừa chỉ để tái sử dụng code:

```
class Stack
{
    private ArrayList           data;
    // Khai báo thêm dữ liệu của Stack.

    public virtual void push(int n);
    public virtual int pop();
}

class Queue: Stack
{
    // Khai báo dữ liệu của Queue.

    public override void push(int n);
    public override int pop();
}
```

Nguyên lý Thay thế Liskov

```
int myFunc(Stack s)
{
    s.push(5);
    s.push(6);
    s.push(7);

    int     a = s.pop();
    int     b = s.pop();

    if (a == 7 && b == 6)
            return a * b;

    throw new ArgumentException();
}
```

Nguyên lý Thay thế Liskov

- Ghi chú:
  - Có mối liên hệ mật thiết với nguyên lý Open-Close.
  - Ưu tiên áp dụng nguyên lý cho các thực thể phần mềm phải thường xuyên nâng cấp, mở rộng.
  - Việc tuân thủ nguyên lý mang tính tương đối, phụ thuộc ngữ cảnh.

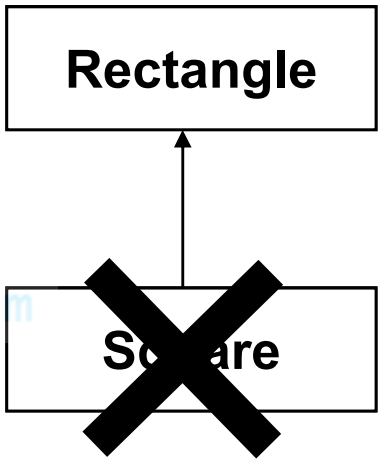
When you inherit from one class, you sign a contract with that class!!

Nguyên lý Thay thế Liskov

- Áp dụng:
  - Quan hệ IS-A đôi khi không chính xác trong việc xác định kế thừa.


```
public double doSomething(Rectangle obj)
{
    obj.setWidth(5);
    obj.setHeight(6);

    if (obj.Area == 30)
        return obj.Area;
    throw new ArgumentException();
}
```



Nguyên lý Thay thế Liskov

- Áp dụng:
  - Sử dụng subtype và composition thay vì dùng subclass.

Nội dung

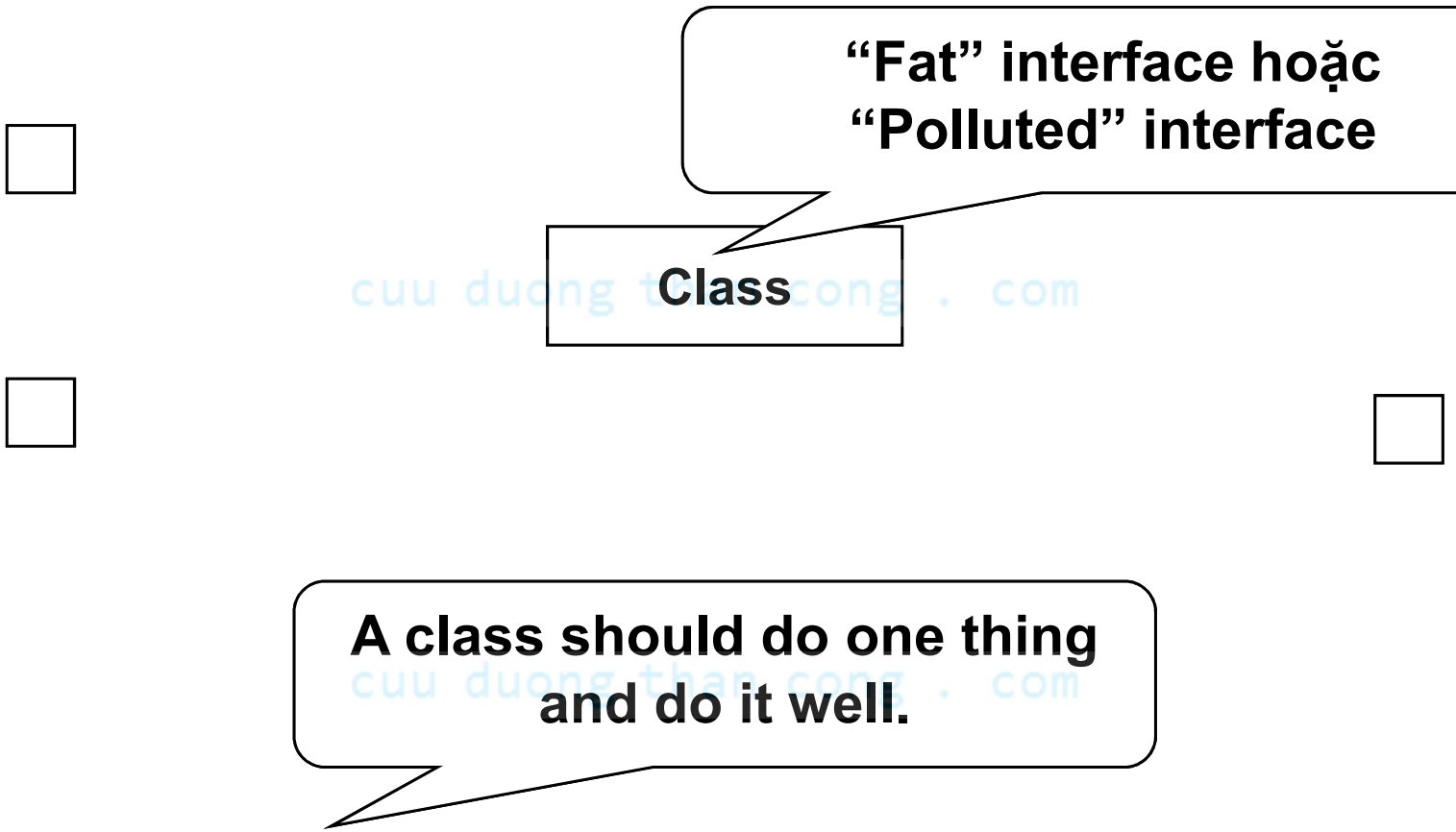
- Phần mềm hướng đối tượng
- Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

Nguyên lý Phân tách Interface

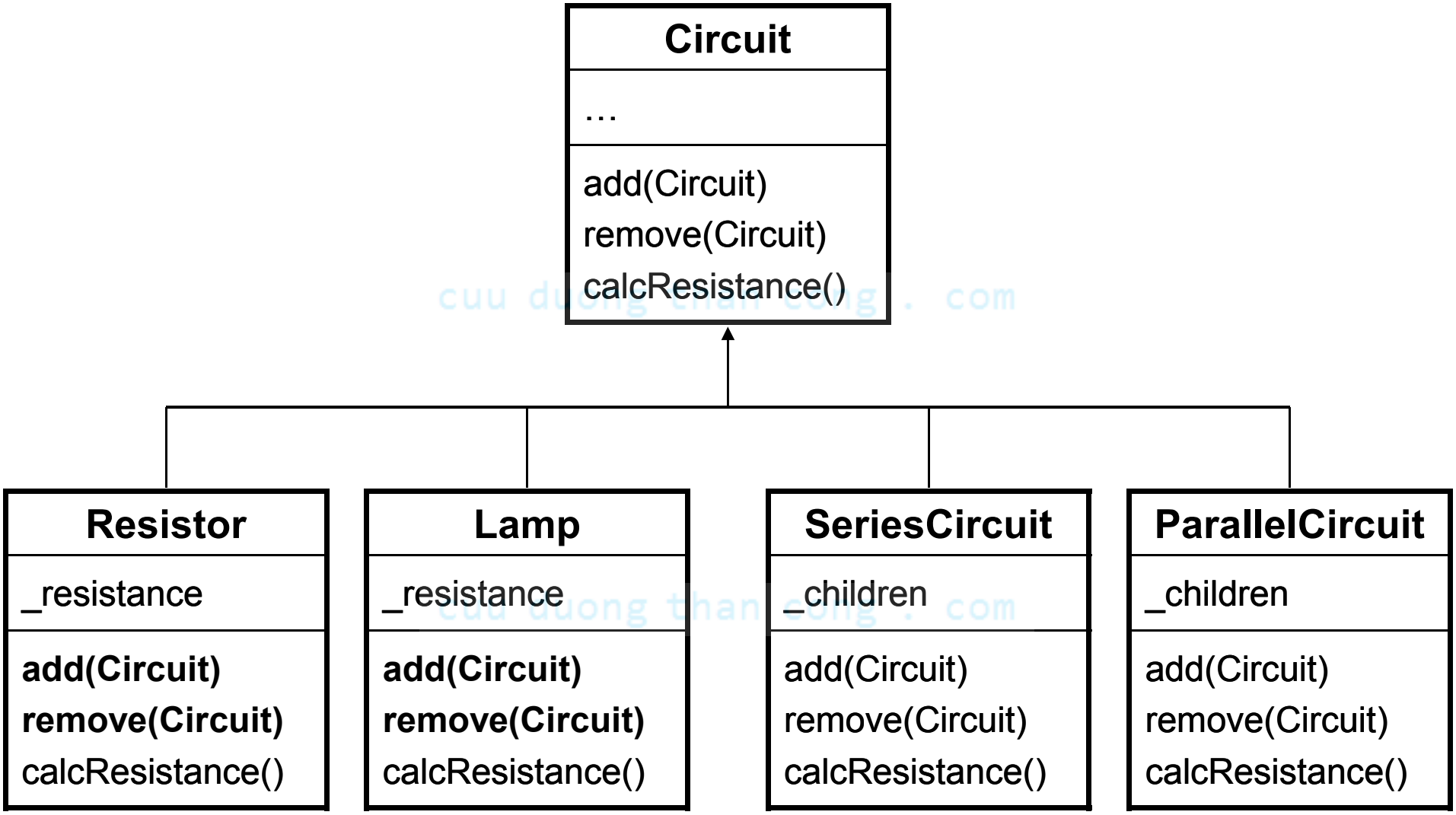
- Phát biểu:

  “Không nên buộc các thực thể phần mềm phụ thuộc vào những interface mà chúng không sử dụng đến.”

Nguyên lý Phân tách Interface



Nguyên lý Phân tách Interface

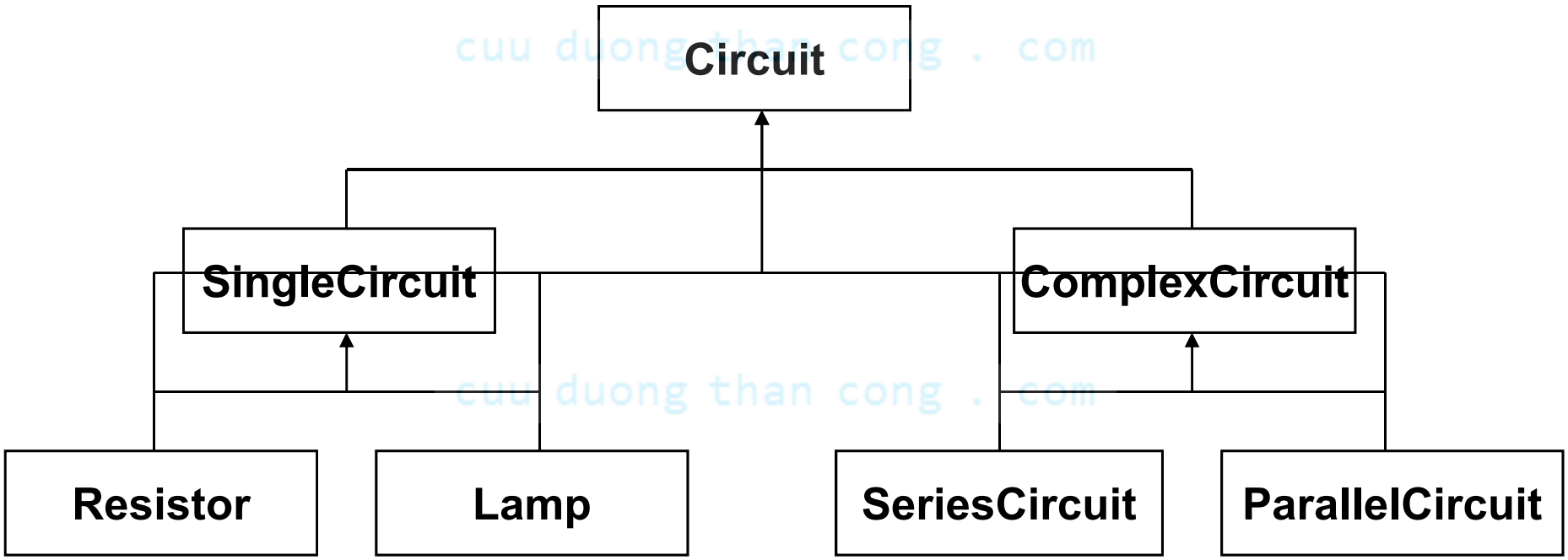


Nguyên lý Phân tách Interface

- Ghi chú:
  - Có mối liên hệ với nguyên lý Open-Close.
  - Thiết kế lớp đối tượng đơn giản, gọn nhẹ.

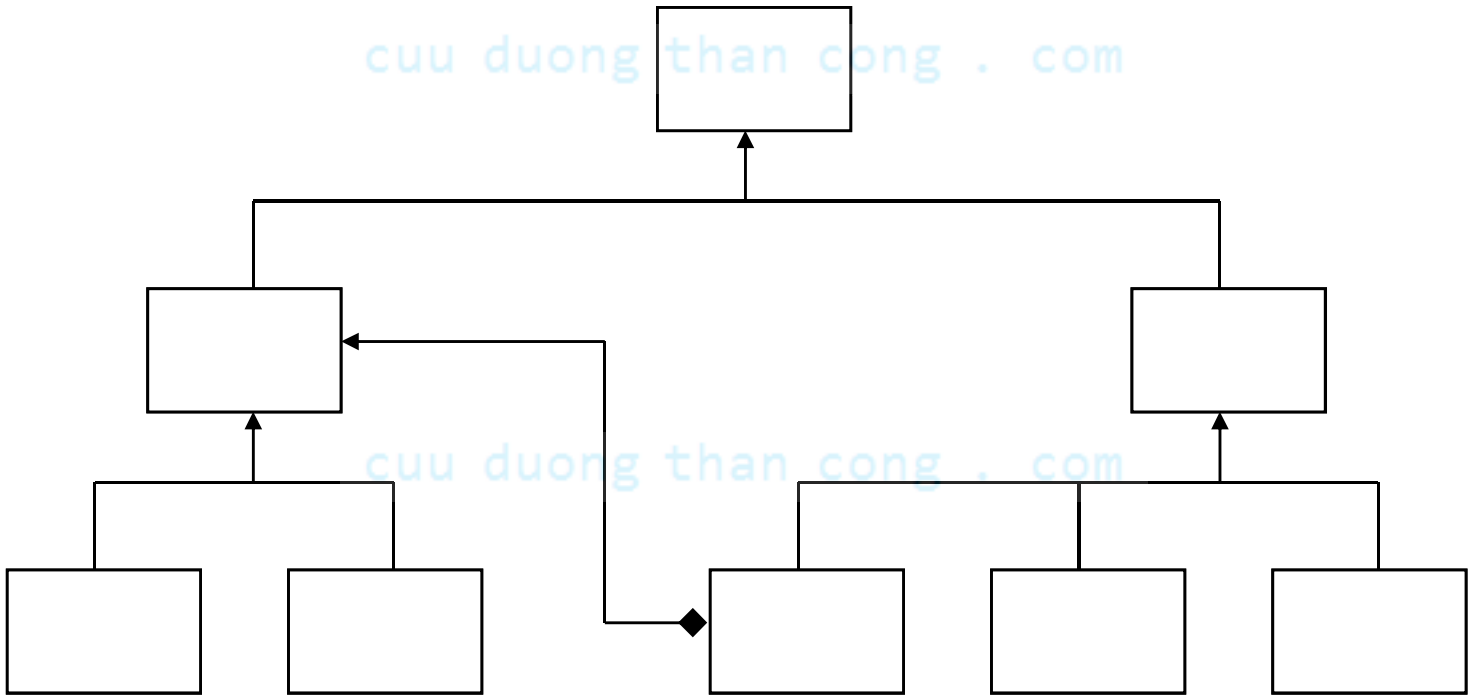
Nguyên lý Phân tách Interface

- Áp dụng:
  - Tăng mức độ trừu tượng trong cây kế thừa.



Nguyên lý Phân tách Interface

- Áp dụng:
  - Sử dụng đa kế thừa hay composition khi cần thiết.



Nội dung

- Phần mềm hướng đối tượng
- Nguyên lý Open-Close
- Nguyên lý Nghịch đảo phụ thuộc
- Nguyên lý Thay thế Liskov
- Nguyên lý Phân tách Interface
- Thảo luận

Thảo luận

Thảo luận

Hỗn độn và đầy may rủi…

There is no Silver Bullet!!

We are building hard software!!

Đang ở đâu?

Từ đâu đến?

Ngành kỹ thuật non trẻ

Mục tiêu

Chất lượng | Hiệu quả

High Level Language

Đi về đâu???

Phụ thuộc vào chính các bạn!