ĐẠI NAM
UNIVERSITY

CHƯƠNG 2
TỔNG QUAN VỀ OOP

ThS. Phạm Văn Tiệp

Nội dung

- Kỹ thuật lập trình
- Kỹ thuật hướng đối tượng
- Các khái niệm cơ bản
- Các nguyên lý
- Phân tích thiết kế hướng đối tượng

ĐẠI NAM
UNIVERSITY

KỸ THUẬT LẬP TRÌNH

Kỹ thuật lập trình

- "**Lập trình hướng đối tượng**" là một kỹ thuật lập trình.
- Vậy "**kỹ thuật lập trình**" là gì?

Kỹ thuật lập trình: Kỹ thuật thực thi một giải pháp phần mềm (cấu trúc dữ liệu + giải thuật) dựa trên nền tảng một **phương pháp luận** (methodology) và một hoặc nhiều **ngôn ngữ lập trình** phù hợp với yêu cầu đặc thù của ứng dụng.

Kỹ thuật lập trình

"Phương pháp luận"

- Các mô thức lập trình
- Các ý tưởng, thuật toán để giải quyết vấn đề
- Phong cách trình bày trong lập trình
- Văn hóa lập trình

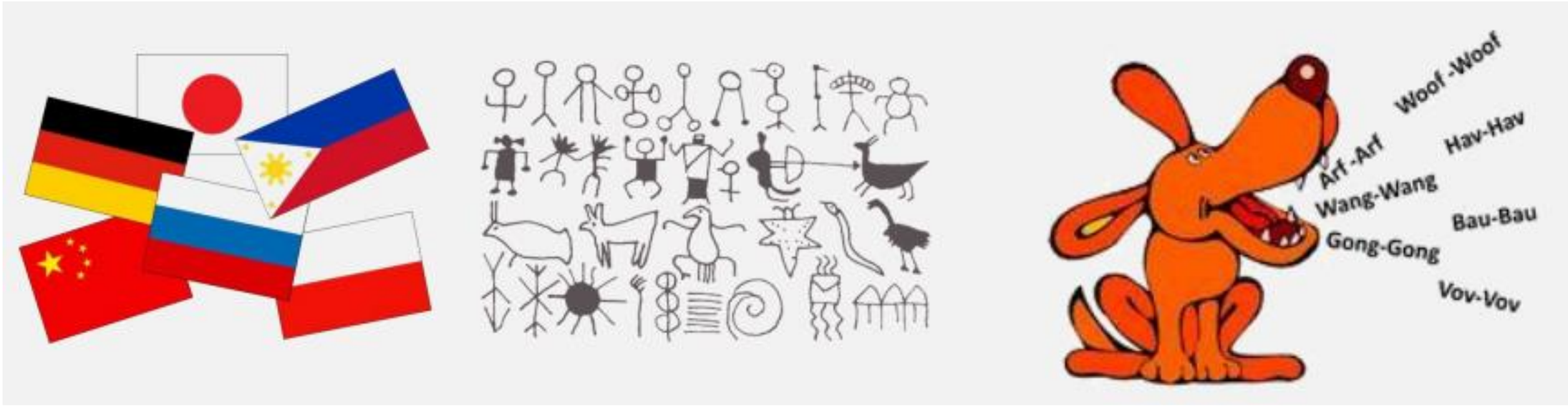
Kỹ thuật lập trình

"Ngôn ngữ lập trình"

- Mô thức - nguyên tắc chung cơ bản
- Cú pháp - xác định cái gì là hợp lệ trong mã nguồn
- Ngữ nghĩa - ngữ pháp của ngôn ngữ lập trình

Kỹ thuật lập trình

- Phương tiện để giao tiếp
- Hệ thống ký hiệu để diễn đạt



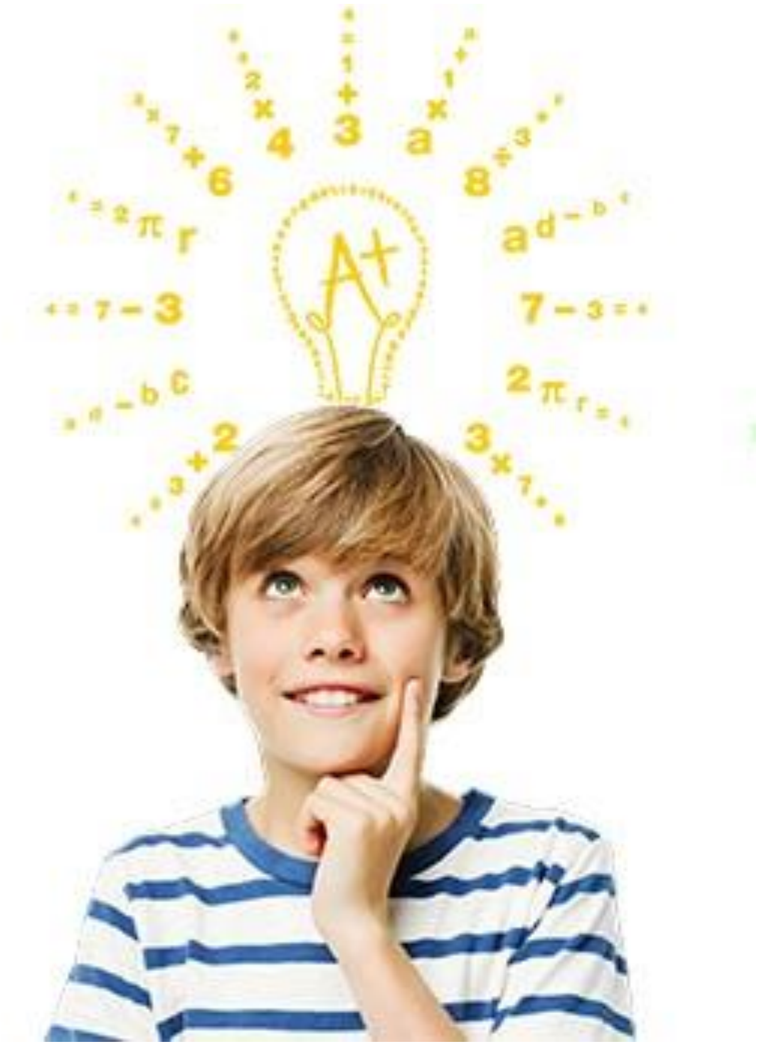
Mục tiêu của kỹ sư phần mềm

- Tạo ra sản phẩm tốt một cách có hiệu quả
- Nắm bắt được công nghệ
- Kiếm được nhiều tiền hơn

Phần mềm ngày càng lớn

- Một số hệ Unix chứa khoảng 4M dòng lệnh
- MS Windows chứa hàng chục triệu dòng lệnh
- Người dùng ngày càng đòi hỏi nhiều chức năng, đặc biệt là chức năng thông minh
- Phần mềm luôn cần được sửa đổi

Vì vậy



- Cần kiểm soát chi phí
  - ✓ Chi phí phát triển
  - ✓ Chi phí bảo trì
- Giải pháp chính là sử dụng lại
  - ✓ Giảm chi phí và thời gian phát triển
  - ✓ Nâng cao chất lượng

Để sử dụng lại (mã nguồn)

- Cần dễ hiểu
- Được coi là chính xác
- Có giao diện rõ ràng
- Không yêu cầu thay đổi khi sử dụng trong chương trình mới

Các phương pháp lập trình

- Lập trình tuần tự (lập trình tuyến tính)
- Lập trình cấu trúc (lập trình hướng thủ tục)
- Lập trình hàm
- Lập trình logic
- Lập trình hướng đối tượng

Lập trình tuần tự

- **Là phương pháp xuất hiện đầu tiên**
  - Các ngôn ngữ như Assembly, Basic
  - Sử dụng các biến tổng thể
  - Lạm dụng lệnh GOTO
- **Các nhược điểm**
  - Khó hiểu, khó bảo trì, hầu như không thể sử dụng lại
  - Chất lượng kém
  - Chi phí cao
  - Không thể phát triển các ứng dụng lớn

Ví dụ

```
10  k =1
20        gosub 100
30  if y > 120 goto 60
40  k = k+1
50        goto 20
60  print k, y
70  stop
100       y = 3*k*k + 7*k-3
110       return
```

Lập trình cấu trúc (hướng thủ tục)

- Sử dụng các lệnh có cấu trúc: for, do while, if then else...
- Các ngôn ngữ: Pascal, C, ...
- Chương trình là tập các hàm/thủ tục
- **Ưu điểm**
  - Chương trình được cục bộ hóa, do đó dễ hiểu, dễ bảo trì hơn
  - Dễ dàng tạo ra các thư viện phần mềm

Ví dụ

```
struct Date {
    int year, mon, day;
};
...
print_date(Date d) {
   printf(”%d / %d / %d\n”, d.day, d.mon, d.year);
}
```

Lập trình cấu trúc (hướng thủ tục)

❖ **Nhược điểm**

- Dữ liệu và mã xử lý (hàm) là tách rời nhau
- Người lập trình phải biết **cấu trúc dữ liệu** (vấn đề này một thời gian dài được coi là hiển nhiên)
- Khi thay đổi câu trúc dữ liệu thì mã xử lý (thuật toán) phải thay đổi theo.
- Khó đảm bảo tính đúng đắn của dữ liệu
- Không tự động khởi tạo hay giải phóng dữ liệu động

Tại sao phải thay đổi cấu trúc của dữ liệu?

- **Cấu trúc dữ liệu là mô hình của bài toán cần giải quyết**
  - Do thiếu kiến thức về bài toán, về miền ứng dụng..., không phải lúc nào cũng tạo được cấu trúc dữ liệu hoàn thiện ngay từ đầu.
  - Tạo ra một cấu trúc dữ liệu hợp lý luôn là vấn đề đau đầu của người lập trình.
- **Bản thân bài toán cũng không bất biến**
  - Cần phải thay đổi cấu trúc dữ liệu để phù hợp với các yêu cầu thay đổi.

Các vấn đề

- **Thay đổi cấu trúc**
  - Dẫn đến việc sửa lại mã chương trình (thuật toán) tương ứng và làm chi phí phát triển tăng cao.
  - Không tái sử dụng được các mã xử lý ứng với cấu trúc dữ liệu cũ.
- **Đảm bảo tính đúng đắn của dữ liệu**
  - Một trong những nguyên nhân chính gây ra lỗi phần mềm là gán các dữ liệu không hợp lệ.
  - Cần phải kiểm tra tính đúng đắn của dữ liệu mỗi khi thay đổi giá trị.

Giải pháp

- **Che dấu dữ liệu (che dấu cấu trúc)**
- **Truy cập dữ liệu thông qua giao diện xác định**

```
class MyDate {
   private int year, mon, day;
   public int getDay() {...}
   public boolean setDay(int) {...}
   ...
}
```

Đóng gói/ che giấu thông tin

- Đóng gói dữ liệu và các thao tác tác động lên dữ liệu thành một thể thống nhất (lớp đối tượng) thuận tiện cho sử dụng lại
- **Che giấu thông tin**
  - Thao tác với dữ liệu thông qua các giao diện xác định
  - Che giấu người lập trình khách (client programmer) cái có khả năng thay đổi (tách cái bất biến ra khỏi cái khả biến)

Sử dụng giao diện

```
MyDate d = new MyDate();

...

d.day = 32; // compile error

d.setDay(31);

d.setMonth(2); // should return False
```

ĐẠI NAM
UNIVERSITY

KỸ THUẬT HƯỚNG ĐỐI TƯỢNG

http://dainam.edu.vn

Lập trình hướng đối tượng là gì

- Mô hình hóa các đối tượng trong thế giới thực thành đối tượng phần mềm.
- **Chương trình = Đối tượng + Thông điệp**
- Chương trình được cấu thành bởi các đối tượng và tương tác giữa các đối tượng (qua thông điệp)
- **Thuộc tính**: các đặc điểm, trạng thái của đối tượng
- **Hành vi**: các hành vi của đối tượng

Lớp

- Lớp (Class): định nghĩa các thuộc tính và các phương thức chung của một nhóm đối tượng nào đó.
  - Lớp là trừu tượng, thuộc tính không mang giá trị cụ thể
  - Có thể liên tưởng đến kiểu dữ liệu
- Lớp là mô hình hóa rút gọn của thực thể trên thực tế
  - Chỉ mô tả những thuộc tính, phương thức quan tâm

Đối tượng

- Đối tượng (Object): là một thể hiện cụ thể của lớp, các thuộc tính có giá trị xác định
  - Có thể liên tưởng đến biến

ĐẠI NAM
UNIVERSITY

CÁC KHÁI NIỆM CƠ BẢN

Đối tượng (object)

- **Đối tượng** là chìa khóa để hiểu được kỹ thuật hướng đối tượng
- Trong hệ thống hướng đối tượng, mọi thứ đều là đối tượng.

Viết một chương trình hướng đối tượng nghĩa là đang xây dựng một mô hình của một vài bộ phận trong thế giới thực

Đối tượng là gì?

- **Đối tượng trong thế giới thực**
  - Ví dụ một chiếc ô tô
- **Liên quan đến chiếc ô tô:**
  - Các thông tin về chiếc xe như: màu sắc, tốc độ, số km đã đi được,...
  - Các hoạt động của chiếc xe như: tăng tốc khi nhấn ga, giảm tốc khi đạp phanh,...

Đối tượng trong thế giới thực

Một **đối tượng thế giới thực** là một thực thể cụ thể mà thông thường chúng ta có thể *sờ, nhìn thấy* hay *cảm nhận* được.

Tất cả có trạng thái (state) và hành động (behaviour)

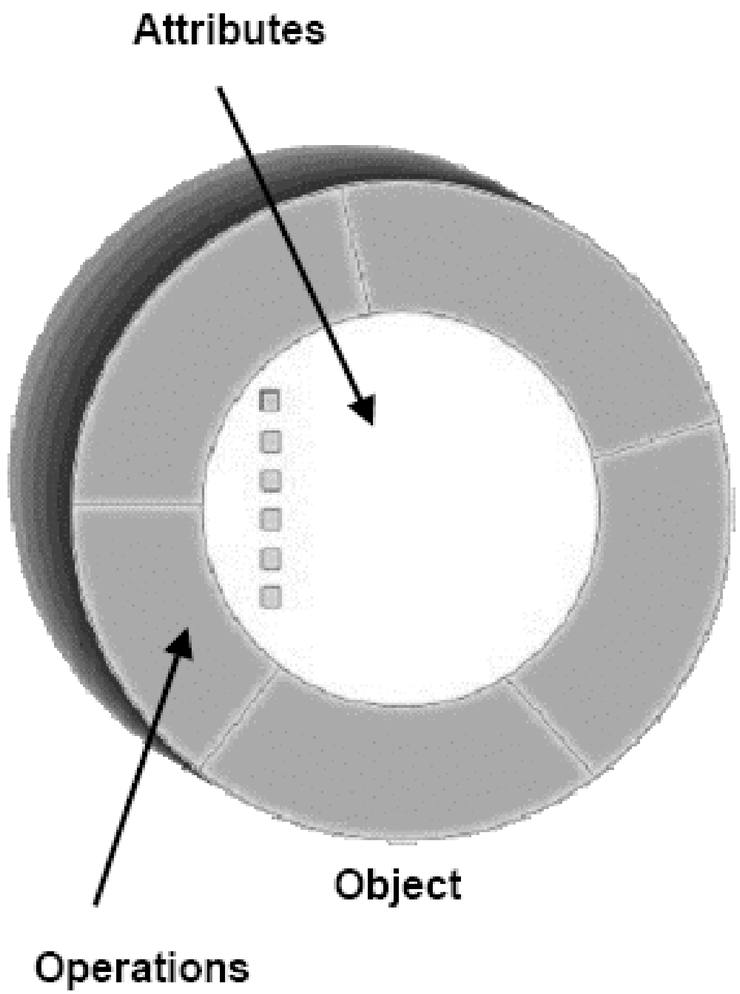
	Trạng thái	**Hành động**
Con chó	Tên Màu Giống Vui sướng	Sủa Vẫy tai Chạy Ăn
Xe đạp	Bánh răng Bàn đạp Dây xích Bánh xe	Tăng tốc Giảm tốc Chuyển bánh răng ...

Đối tượng là gì?

• Là một thực thể được đóng gói thành trạng thái (state) và hành vi (behavior).

• **Trạng thái** được biểu diễn bởi các thuộc tính (attributes) và các mối quan hệ (relationships).

• **Hành vi** được biểu diễn bởi các thao tác (operations), phương thức (methods)



Trạng thái

Dave
Age: 32
Height: 1m84

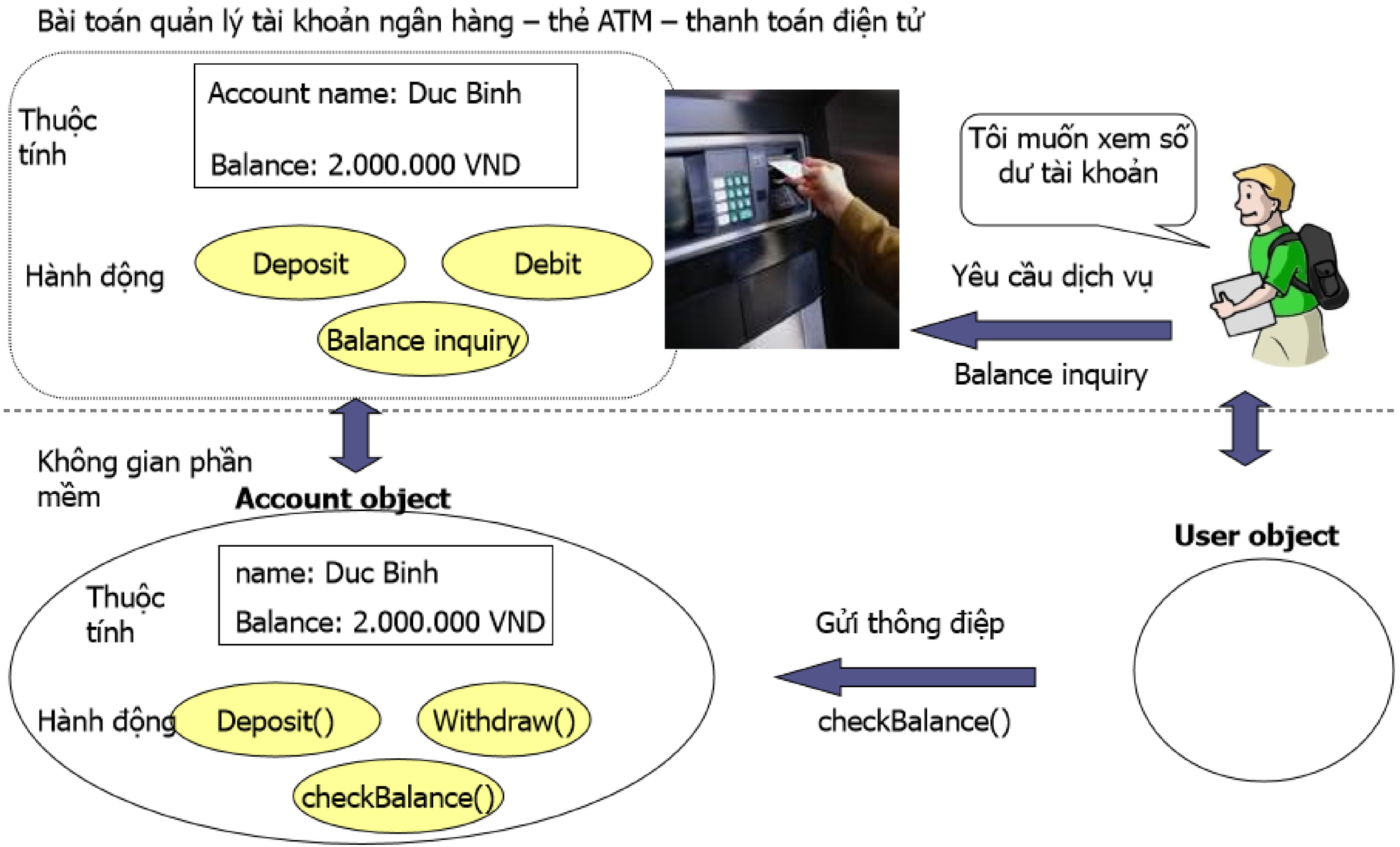
Brett
Age: 35
Height: 1m78

Gary
Age: 60
Height: 1m75

Hành vi

Get the mail.
Cook dinner.

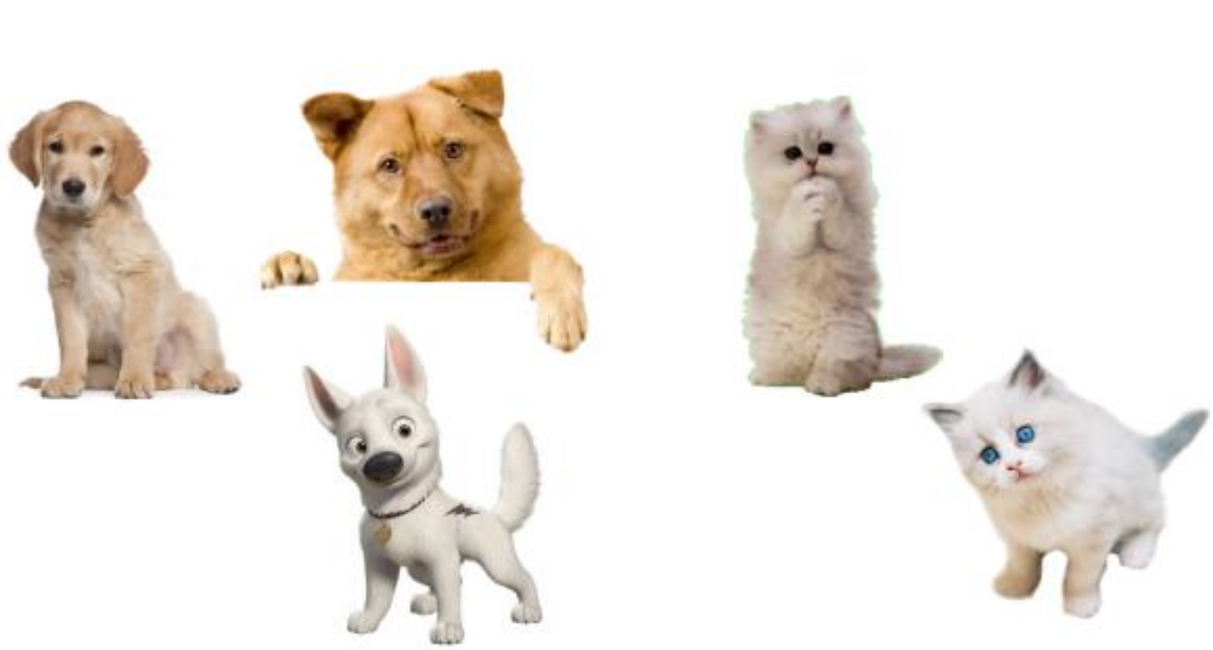
Đối tượng phần mềm và bài toán thực tiễn



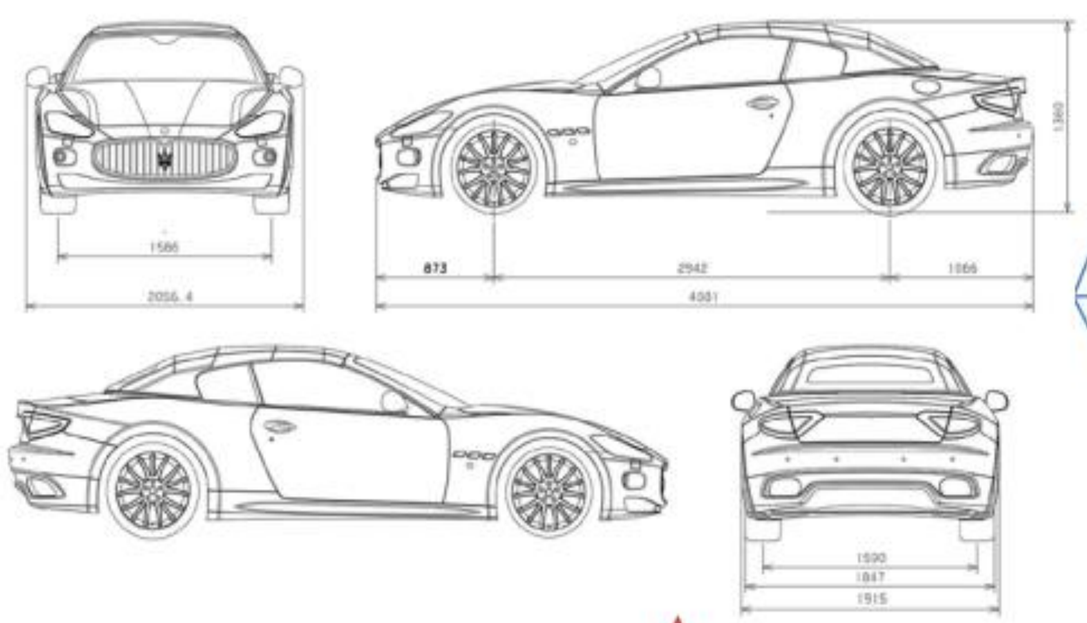
Lớp

Một lớp là một thiết kế (blueprint) hay mẫu (prototype) cho các đối tượng cùng kiểu

- Lớp định nghĩa các thuộc tính và các phương thức chung cho tất cả các đối tượng của cùng một loại nào đó
- Một đối tượng là một thể hiện cụ thể của một lớp.
- Mỗi thể hiện có thể có những thuộc tính thể hiện khác nhau

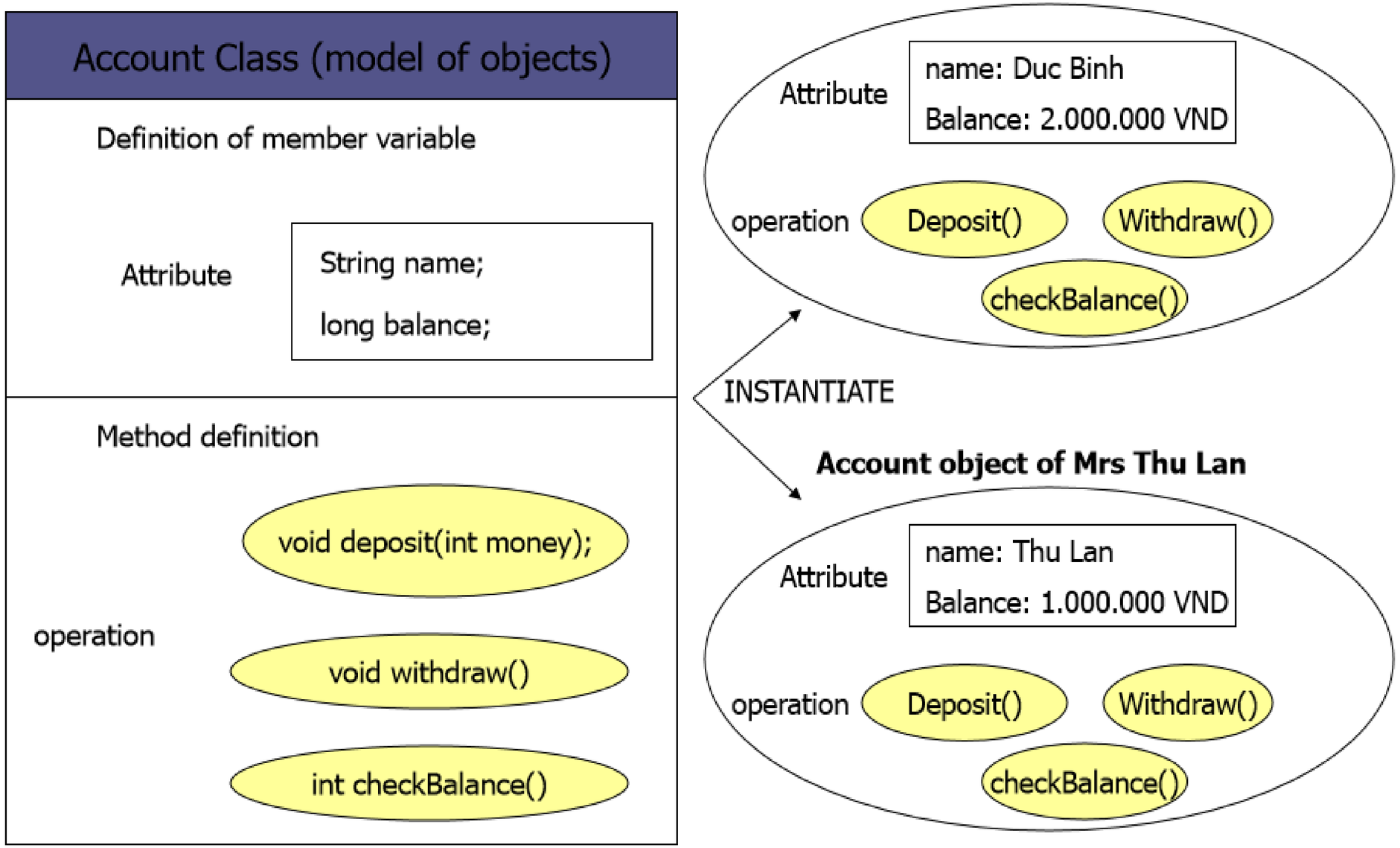


Lớp và đối tượng



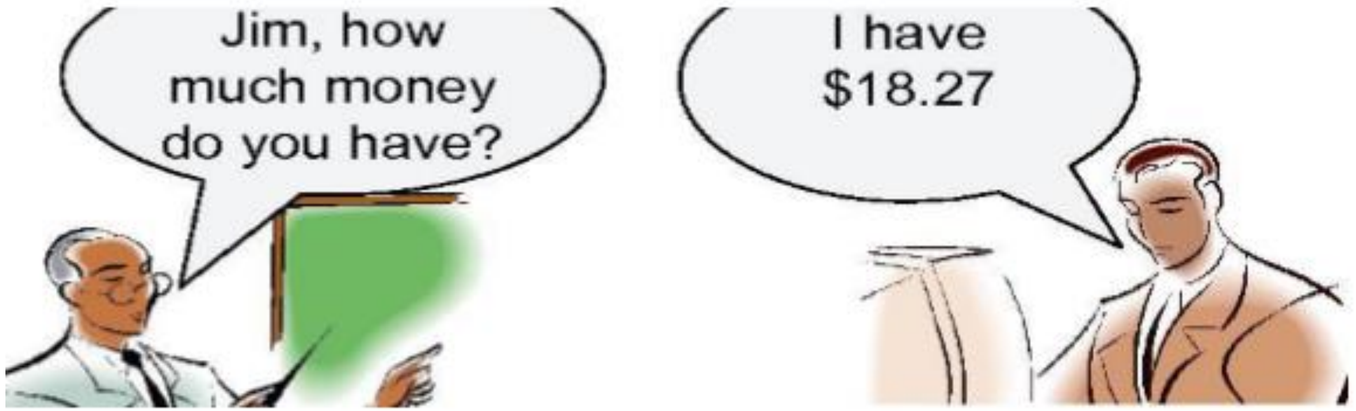
Class

Lớp và đối tượng

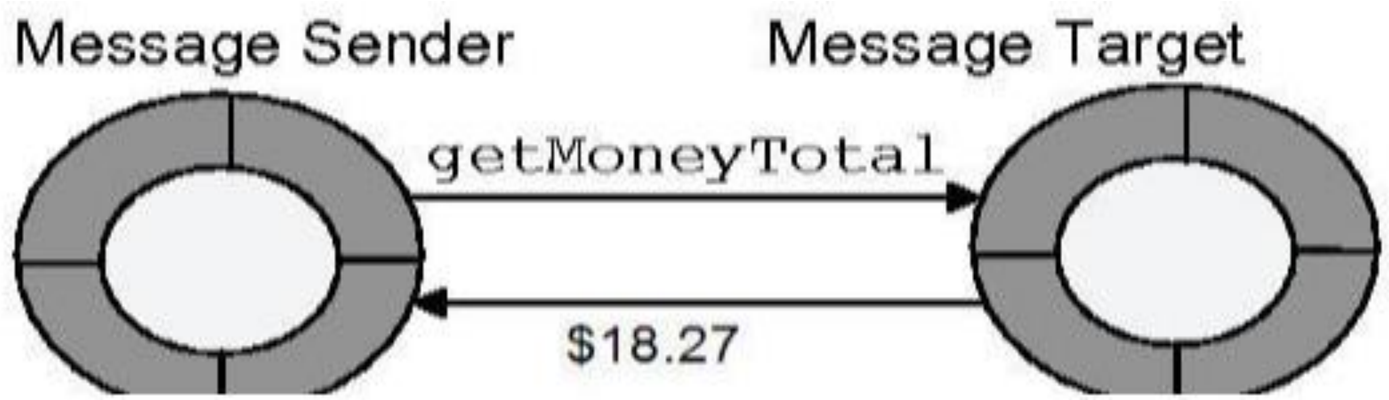


Tương tác giữa các đối tượng

- Sự giao tiếp giữa các đối tượng trong thế giới thực:

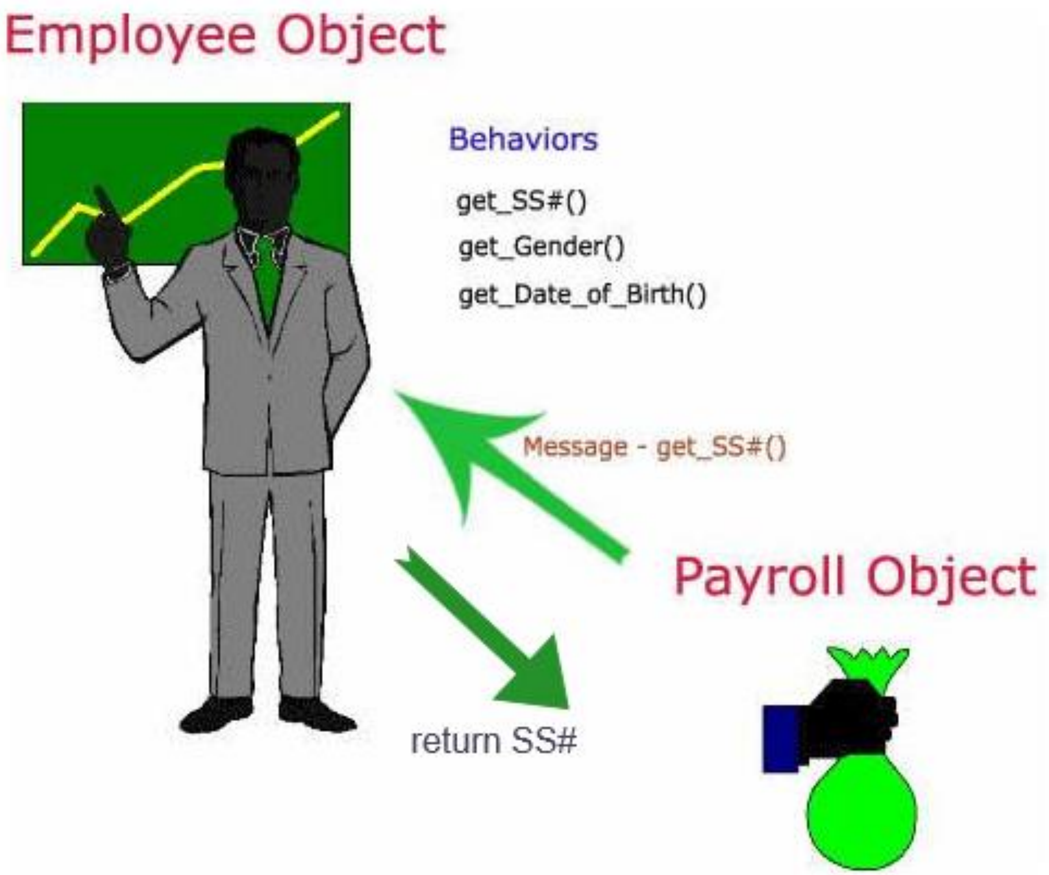


- Các đối tượng và sự tương tác giữa chúng trong lập trình
  - Các đối tượng giao tiếp với nhau bằng cách gửi thông điệp (message)



Trao đổi thông điệp

- Một chương trình (xây dựng theo tiếp cận HĐT) là tập các đối tượng trao đổi thông điệp với nhau.

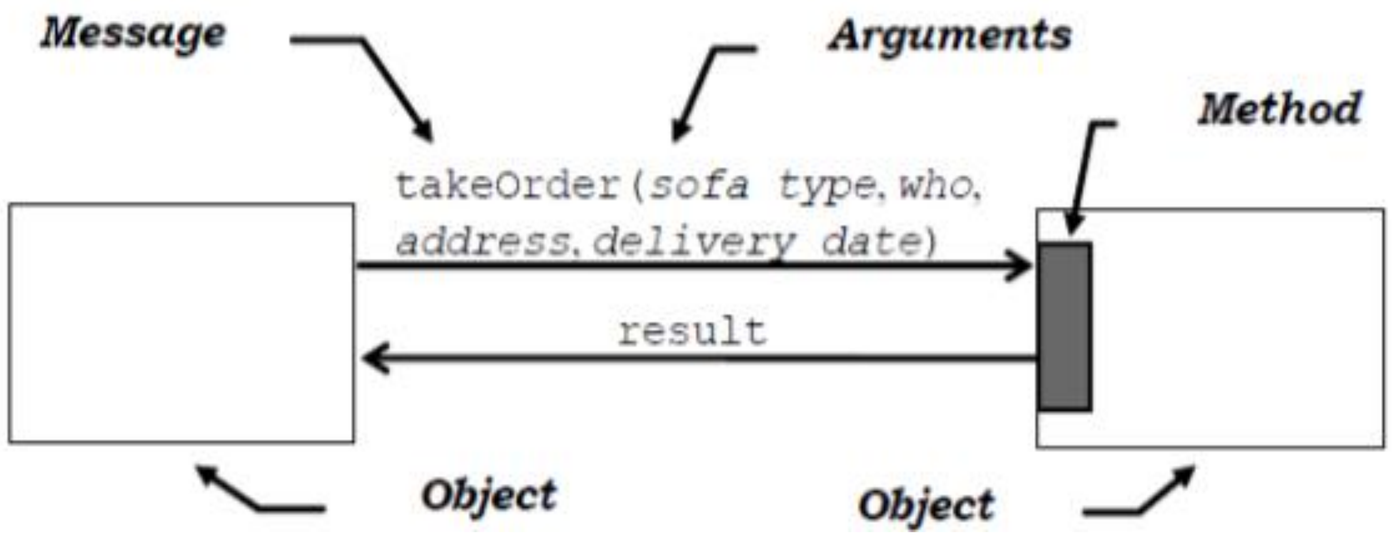


Gọi hàm với gửi thông điệp

- **Gọi hàm (Call function)**
  - Chỉ ra chính xác đoạn mã nào sẽ được thực hiện.
  - Chỉ có duy nhất một sự thực thi của một hàm với một tên nào đó.
  - Không có hai hàm trùng tên
- **Gửi thông điệp**
  - Yêu cầu một dịch vụ từ một đối tượng và đối tượng sẽ quyết định cần phải làm gì
  - Các đối tượng khác nhau sẽ có các cách thực thi các thông điệp theo cách khác nhau

Thông điệp với phương thức

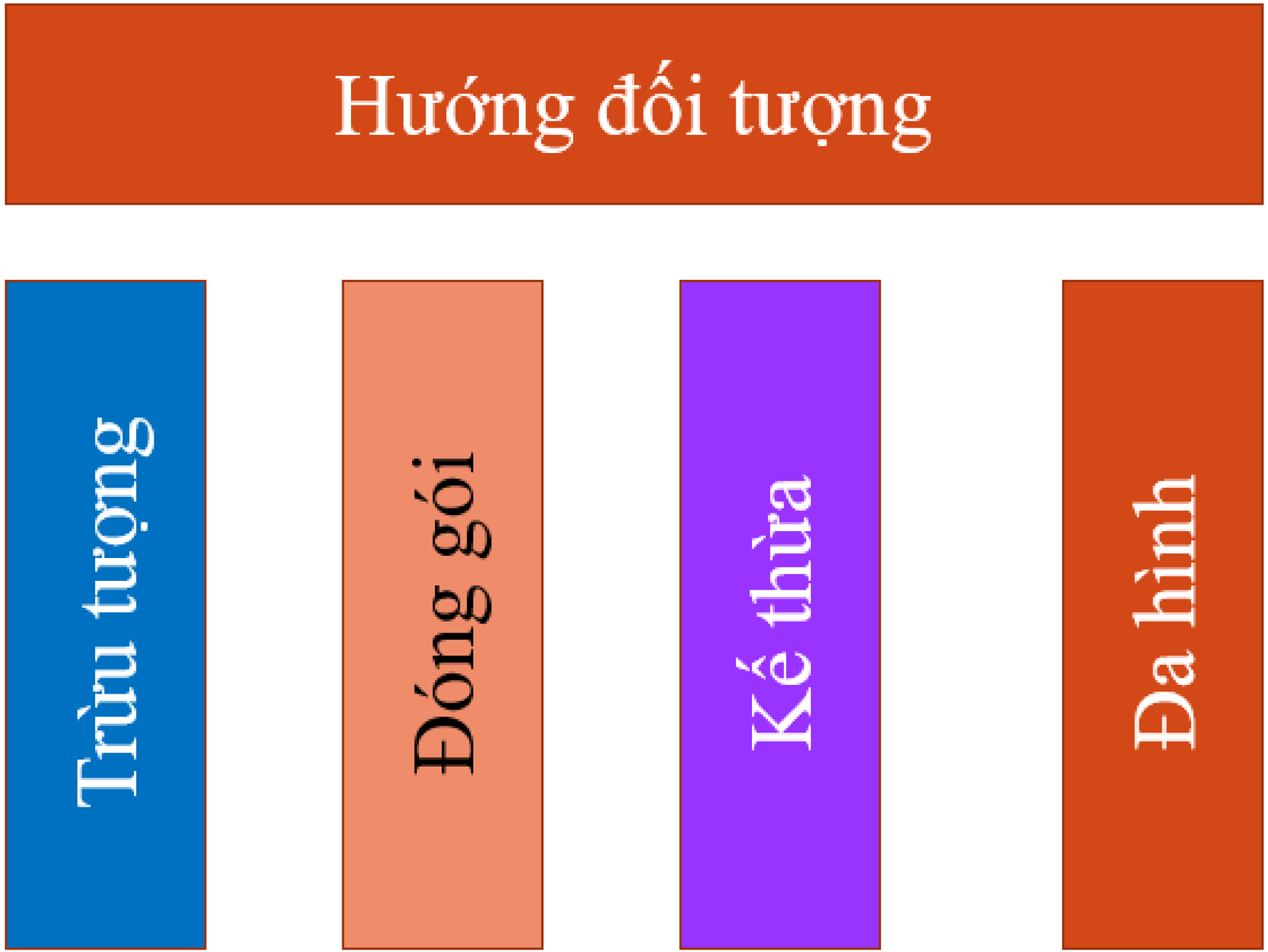
- **Thông điệp**
  - Được gửi từ đối tượng này đến đối tượng kia, không bao gồm đoạn mã thực sự sẽ được thực thi
- **Phương thức**
  - Thủ tục/hàm trong ngôn ngữ lập trình cấu trúc
  - Là sự thực thi dịch vụ được yêu cầu bởi thông điệp
  - Là đoạn mã sẽ được thực thi để đáp ứng thông điệp được gửi đến cho đối tượng



ĐẠI NAM
UNIVERSITY

CÁC NGUYÊN LÝ CƠ BẢN

Các nguyên lý cơ bản của OOP



Tính trừu tượng

- "Sự bỏ qua có chọn lựa"
  - Quyết định cái gì là quan trọng và không.
  - Tập trung và dựa trên những gì là quan trọng
  - Bỏ qua và không phụ thuộc vào những gì là không quan trọng

Tính trừu tượng

- Loại bỏ đi các thông tin cụ thể, giữ lại các thông tin chung
- Tập trung vào các đặc điểm của thực thể, làm cho nó khác biệt với những thực thể khác
- Phụ thuộc góc nhìn
  - Quan trọng trong ngữ cảnh này nhưng lại không có ý nghĩa nhiêu trong ngữ cảnh khác.

Ví dụ: Trừu tượng

Car

- Start
- Stop
- Accelerate
- Brake

- Engine
- Water pump
- Radiator
- Aircon system

→ Khách hàng biết

→ Khách hàng không biết

Trừu tượng hoá – Góc nhìn

Registration

- Vehicle Identification Number
- License plate
- Current Owner
- Tax due, date

Owner

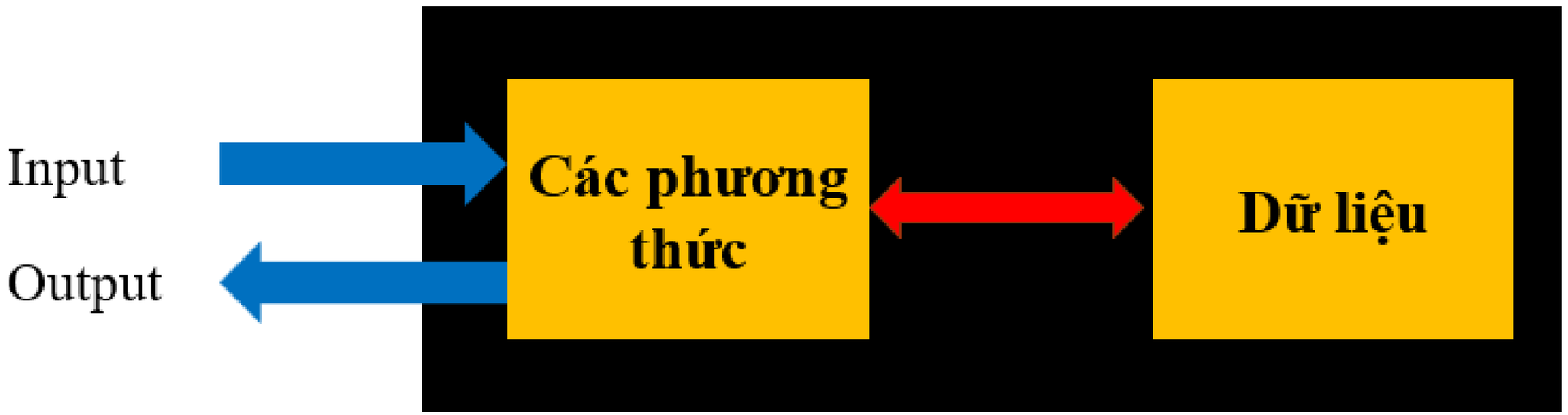
- Car description
- Service history
- Petrol mileage history

Garage

- License plate
- Work description
- Billing info
- Owner

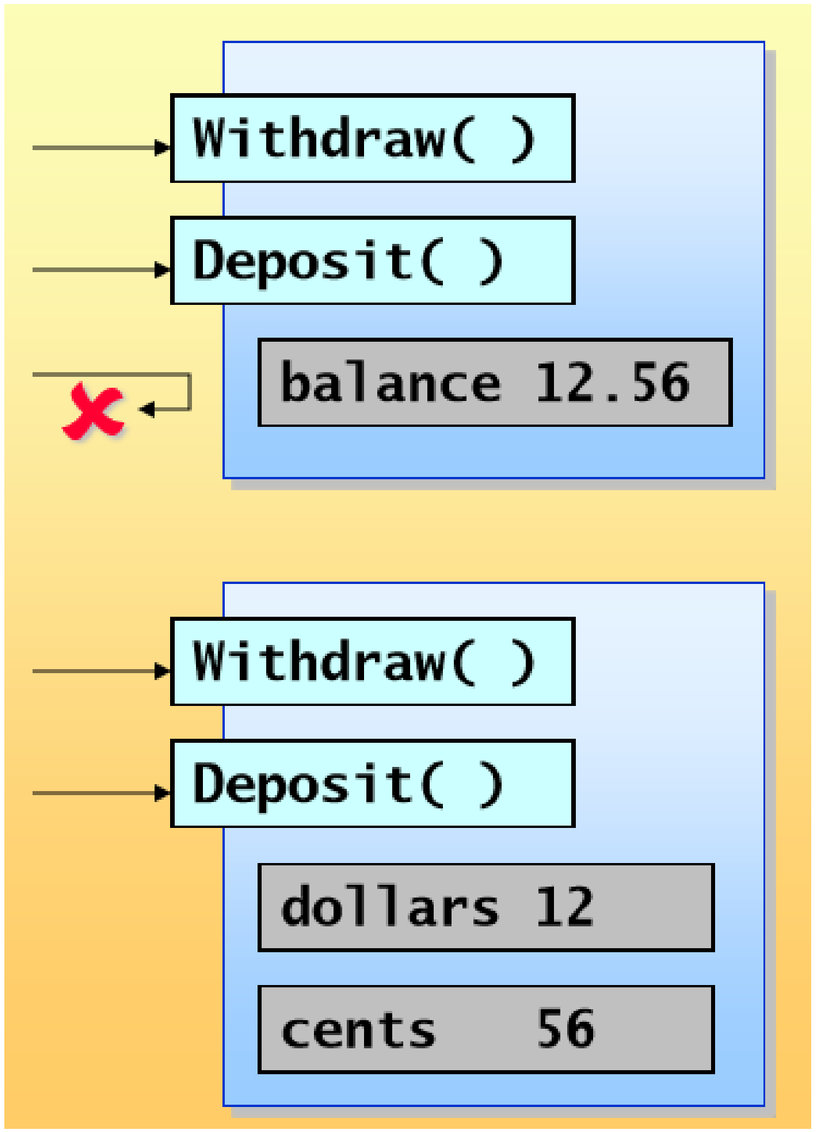
Đóng gói (Encapsulation)

- **Đóng gói**: là tính chất không cho phép đối tượng khác thay đổi dữ liệu thành viên của đối tượng nội tại.
- Chỉ có các phương thức của đối tượng đó mới có quyền thay đổi trạng thái nội tại của nó mà thôi.
- Các đối tượng khác muốn thay đổi thuộc tính của đối tượng nội tại, thì chúng cần truyền thông điệp cho đối tượng, và việc quyết định thay đổi hay không vẫn do đối tượng nội tại quyết định.



Tính đóng gói đem lại lợi ích gì?

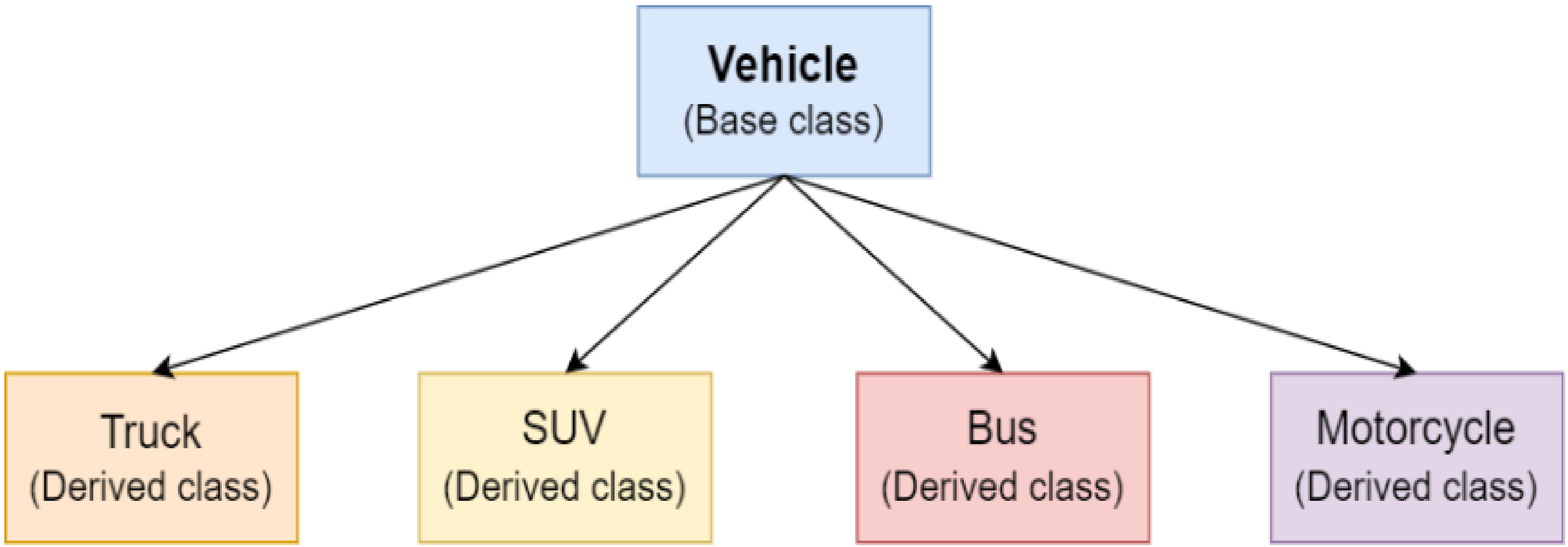
- **Cho phép điều khiển**
  - Việc sử dụng đối tượng được kiểm soát thông qua các method public



- **Hỗ trợ sự thay đổi**
  - Việc sử dụng đối tượng không bị ảnh hưởng nếu dữ liệu nội tại (private) bị thay đổi

Tính kế thừa

- Tính kế thừa là cơ chế cho phép xây dựng một lớp mới dựa trên lớp đã có.



Tính đa hình

- **Đa hình**: nhiều hình thức thực hiện một hành vi, nhiều kiểu tồn tại của một đối tượng.

ĐẠI NAM
UNIVERSITY

PHÂN TÍCH THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

Phân tích thiết kế hướng đối tượng

- Phương pháp luận (methodology) trong phân tích và thiết kế phần mềm thông thường được định nghĩa như là một tập các quá trình và thao tác để tìm và khám phá cách có thể giải quyết được bài toán phần mềm.
- Một trong các phương pháp hiệu quả nhất để phát triển phần mềm.

Phát triển phần mềm

- **Sáu giai đoạn**
  - Giai đoạn 0: Lập kế hoạch (make a plan)
  - Giai đoạn 1: Xác định mục tiêu - làm gì (what are we making)
  - Giai đoạn 2: Xác định cách làm thế nào (how to build it)
  - Giai đoạn 3: Xây dựng phần lõi - Building the core
  - Giai đoạn 4: Lặp lại (hiệu chỉnh) các trường hợp sử dụng
  - Giai đoạn 5: Phát triển (evolution)

Xác định mục tiêu

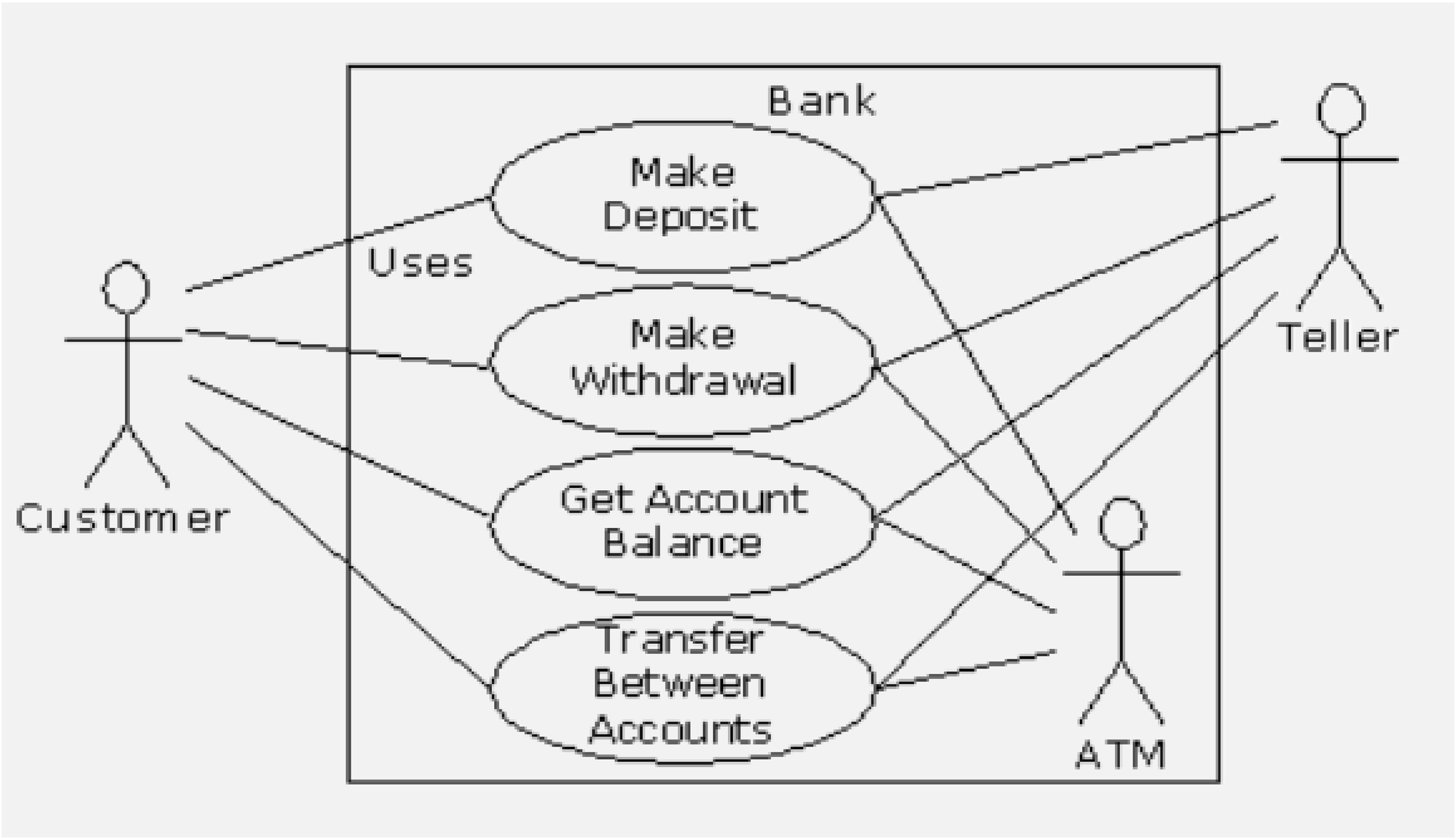
• Giai đoạn 1: Xác định mục tiêu - làm gì (what are we making)

• Trong giai đoạn này chúng ta có nhiệm vụ xác định cụ thể các mục tiêu, chức năng và nhiệm vụ mà phần mềm chúng ta cần xây dựng phải đáp ứng.

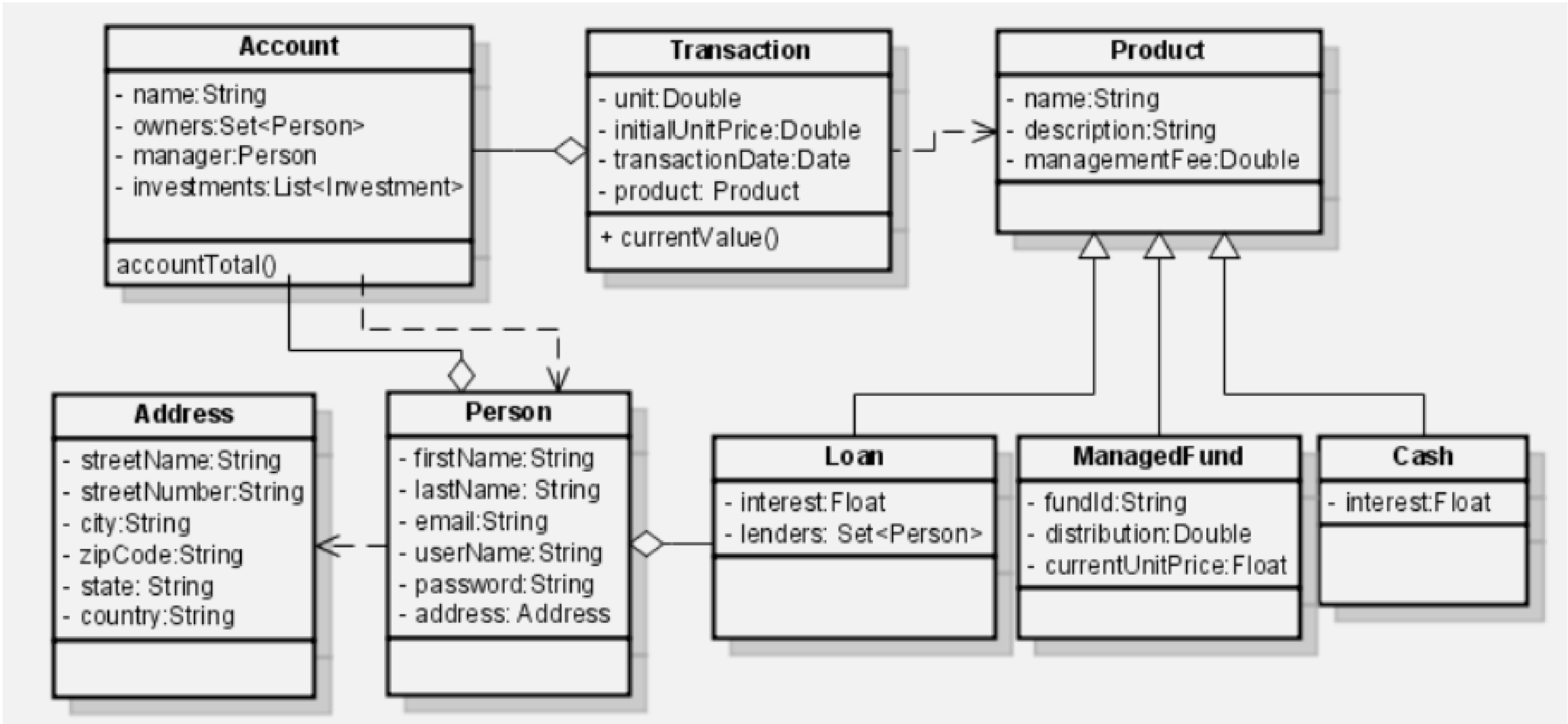
• Trong phương pháp lập trình cổ điển hướng thủ tục người ta gọi giai đoạn này là giai đoạn tạo ra "phân tích yêu cầu và mô tả hệ thống" (requirements analysis and system specification).

• Trong phân tích và thiết kế hướng đối tượng người ta sử dụng các ký pháp và kỹ thuật Use case để mô tả các công việc này.

Biểu đồ use case



Biểu đồ lớp


Account
- name:String
- owners:Set<Person>
- manager:Person
- investments:List<Investment>
accountTotal()
Transaction
- unit:Double
- initialUnitPrice:Double
- transactionDate:Date
- product: Product
+ currentValue()
Product
- name:String
- description:String
- managementFee:Double
Address
- streetName:String
- streetNumber:String
- city:String
- zipCode:String
- state: String
- country:String
Person
- firstName:String
- lastName: String
- email:String
- userName:String
- password:String
- address: Address
Loan
- interest:Float
- lenders: Set<Person>
ManagedFund
- fundId:String
- distribution:Double
- currentUnitPrice:Float
Cash
- interest:Float