Trường Cao đẳng Công nghệ Thông tin TP.HCM

Khoa Công nghệ Thông tin – Điện tử

Chương 3:

CÁC KHÁI NIỆM VỀ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Giảng viên: Hà Mỹ Trinh

Email: trinhhm@itc.edu.vn

Nội dung

1. Phương pháp tiếp cận của lập trình truyền thống
2. Phương pháp tiếp cận của lập trình hướng đối tượng
3. Sơ lược về các khái niệm cơ bản
4. Phân tích thiết kế hướng đối tượng

1. Phương pháp tiếp cận của lập trình truyền thống

- **Lập trình truyền thống** đã trải qua hai giai đoạn:
  - Giai đoạn sơ khai, khi khái niệm lập trình mới ra đời, là ***lập trình tuyến tính***.
  - Giai đoạn tiếp theo, là ***lập trình hướng cấu trúc***.
- **Lập trình tuyến tính**
  - Đặc trưng cơ bản của lập trình tuyến tính là tư duy theo lối tuần tự. Chương trình sẽ được thực hiện tuần tự từ đầu đến cuối, lệnh này kế tiếp lệnh kia cho đến khi kết thúc chương trình.
  - Đặc trưng: có 2 đặc trưng
    - Đơn giản: chương trình được tiến hành đơn giản theo lối tuần tự, không phức tạp.
    - Đơn luồng: chỉ có một luồng (thread) công việc duy nhất, và các công việc được thực hiện tuần tự trong luồng đó.

1. Phương pháp tiếp cận của lập trình truyền thống

Lập trình cấu trúc

- Đặc trưng
  - Đặc trưng cơ bản nhất của lập trình cấu trúc thể hiện ở mối quan hệ:
  - Chương trình = CTDL + Giải thuật
- Trong đó
  - CTDL là cách tổ chức dữ liệu, cách mô tả bài toán dưới dạng ngôn ngữ lập trình
  - Giải thuật là một quy trình để thực hiện một công việc xác định
  - Trong chương trình, giải thuật có quan hệ phụ thuộc vào CTDL:
  - Một CTDL chỉ phù hợp với một số hạn chế các giải thuật.
  - Nếu thay đổi CTDL thì phải thay đổi giải thuật cho phù hợp.
  - Một giải thuật thường phải đi kèm với một CTDL nhất định.

1. Phương pháp tiếp cận của lập trình truyền thống

Lập trình cấu trúc

- Tính chất
  - Ưu điểm
    - Chương trình sáng sủa, dễ hiểu, dễ theo dõi.
    - Tư duy giải thuật rõ ràng.
  - Nhược điểm
    - ***Không*** hỗ trợ việc ***sử dụng lại mã nguồn: Giải thuật luôn phụ thuộc*** chặt chẽ vào ***CTDL***, do đó, khi thay đổi CTDL, phải thay đổi giải thuật, nghĩa là phải viết lại chương trình.
    - ***Không phù hợp với các phần mềm lớn***: tư duy cấu trúc với các giải thuật chỉ phù hợp với các bài toán nhỏ, nằm trong phạm vi một mô đun của chương trình. Với dự án phần mềm lớn, lập trình cấu trúc tỏ ra không hiệu quả trong việc giải quyết mối ***quan hệ vĩ mô giữa các mô đun*** của phần mềm.

2. Phương pháp tiếp cận của lập trình hướng đối tượng

- Xuất phát từ hai hạn chế chính của phương pháp lập trình cấu trúc
  - Không quản lý được sự thay đổi dữ liệu khi có nhiều chương trình cùng thay đổi một biến chung. Vấn đề này đặc biệt nghiêm trọng khi các ứng dụng ngày càng lớn, người ta ***không thể kiểm soát được sự truy nhập*** đến các biến dữ liệu chung.
  - ***Không tiết kiệm được tài nguyên con người***: Giải thuật gắn liền với CTDL, nếu thay đổi CTDL, sẽ phải thay đổi giải thuật, và do đó, phải viết lại mã chương trình từ đầu.
- Để khắc phục được hai hạn chế này khi giải quyết các bài toán lớn, người ta xây dựng một phương pháp tiếp cận mới, là **phương pháp lập trình hướng đối tượng**, với hai mục đích chính:

2. Phương pháp tiếp cận của lập trình hướng đối tượng

- Đóng gói dữ liệu để hạn chế sự truy nhập tự do vào dữ liệu, không quản lý được.
- Cho phép sử dụng lại mã nguồn, hạn chế việc phải viết lại mã từ đầu cho các chương trình.

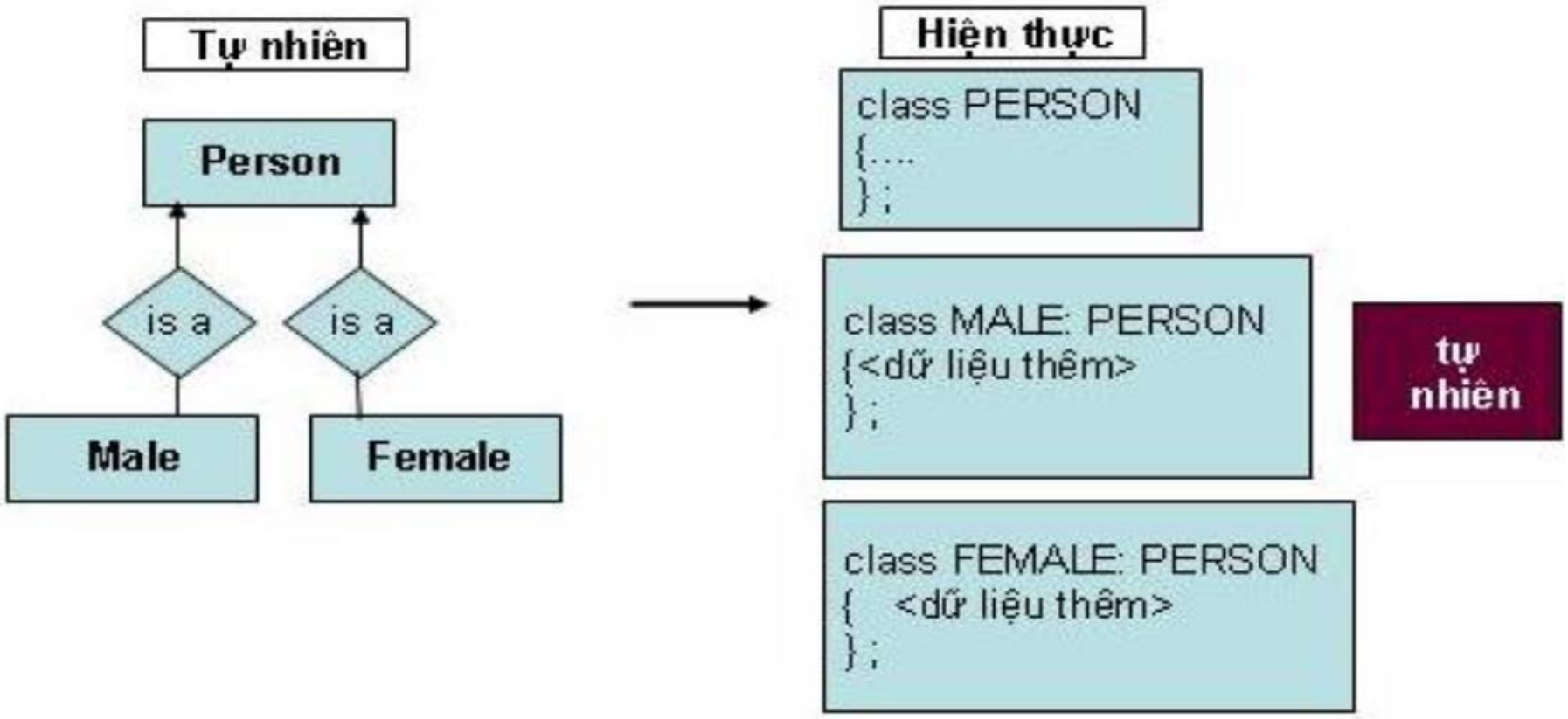
OOP – Object Oriented Programming

- Chương trình là sự hoạt động của các đối tượng →Giống tự nhiên
- Đối tượng thực thi một hoạt động tức là đối tượng thực hiện một hành vi mà đối tượng này có khả năng
- Một chương trình là một trật tự các lời yêu cầu đối tượng thực hiện hành vi của mình

→ Chương trình là một kịch bản (Script)

2. Phương pháp tiếp cận của lập trình hướng đối tượng

- Ưu điểm của OOP
  - Dễ mô tả các quan hệ phân cấp trong thế giới tự nhiên
  - Có tính bảo mật cao: Bên ngoài không thể tùy tiện truy cập một dữ liệu thuộc tính



2. Phương pháp tiếp cận của lập trình hướng đối tượng

- Ưu điểm của OOP
  - Dễ tái sử dụng code

```java
class Nguoi {
    private String ten;
    private int tuoi;

    public Nguoi(String ten, int tuoi) {
        this.ten = ten;
        this.tuoi = tuoi;
    }

    public void xuatTenTuoi() {
        System.out.println(ten + " - " + tuoi);
    }
}
```

```java
class HocSinh extends Nguoi {
    private int diem;

    public HocSinh(String ten, int tuoi, int diem) {
        super(ten, tuoi); // gọi constructor lớp cha
        this.diem = diem;
    }

    public void xuatDiem() {
        System.out.println(diem);
    }
}

public class Main {
    public static void main(String[] args) {
        HocSinh hs = new HocSinh("Hoa", 19, 5);
        hs.xuatTenTuoi();
        hs.xuatDiem();
    }
}
```

3. Sơ lược về các khái niệm cơ bản

Các đặc trưng của lập trình OOP

- Tính đóng gói
  - Dữ liệu luôn được tổ chức thành các thuộc tính của lớp đối tượng. Việc truy nhập đến dữ liệu phải thông qua các phương thức của đối tượng lớp.
  - Cơ chế đóng gói (encapsulation) là phương thức tốt để thực hiện cơ chế che dấu thông tin so với các ngôn ngữ lập trình cấu trúc
- Tính kế thừa:
  - Cho phép sử dụng lại mã nguồn. Các lớp đối tượng có thể kế thừa (inheritance) từ các lớp đối tượng khác. Khi đó, trong các lớp kế thừa, có thể sử dụng các phương thức hoạt động của các lớp bị kế thừa, mà không cần phải định nghĩa lại.
- Tính đa hình: Kỹ thuật cho phép có khác biệt giữa code của cùng một hành vi trong lớp cha và trong lớp con

3. Sơ lược về các khái niệm cơ bản

- Các nguyên lý cơ bản

Lập trình hướng đối tượng (OOP)			
Trừu tượng hóa (Abstraction)	Đóng gói (Encapsulation)	Module hóa (Modularization)	Kế thừa (Inheritance)
- Chỉ giữ lại những đặc điểm quan trọng - VD: class Nguoi với tên, tuổi → Tập trung vào tính năng chính.	- Giấu dữ liệu bên trong đối tượng và chỉ cho phép truy cập qua các phương thức (get/set). → Bảo vệ dữ liệu; kiểm soát dữ liệu được nhập	- Chia chương trình lớn thành nhiều phần nhỏ, mỗi phần làm một việc rõ ràng. - VD: Nguoi.java HocSinh.java GiaoVien.java	- Cho phép một lớp kế thừa thuộc tính và phương thức của lớp khác - VD: class HocSinh extends Nguoi → Tái sử dụng; mở rộng chức năng dễ dàng; hệ thống Mở rộng chức năng dễ dàng

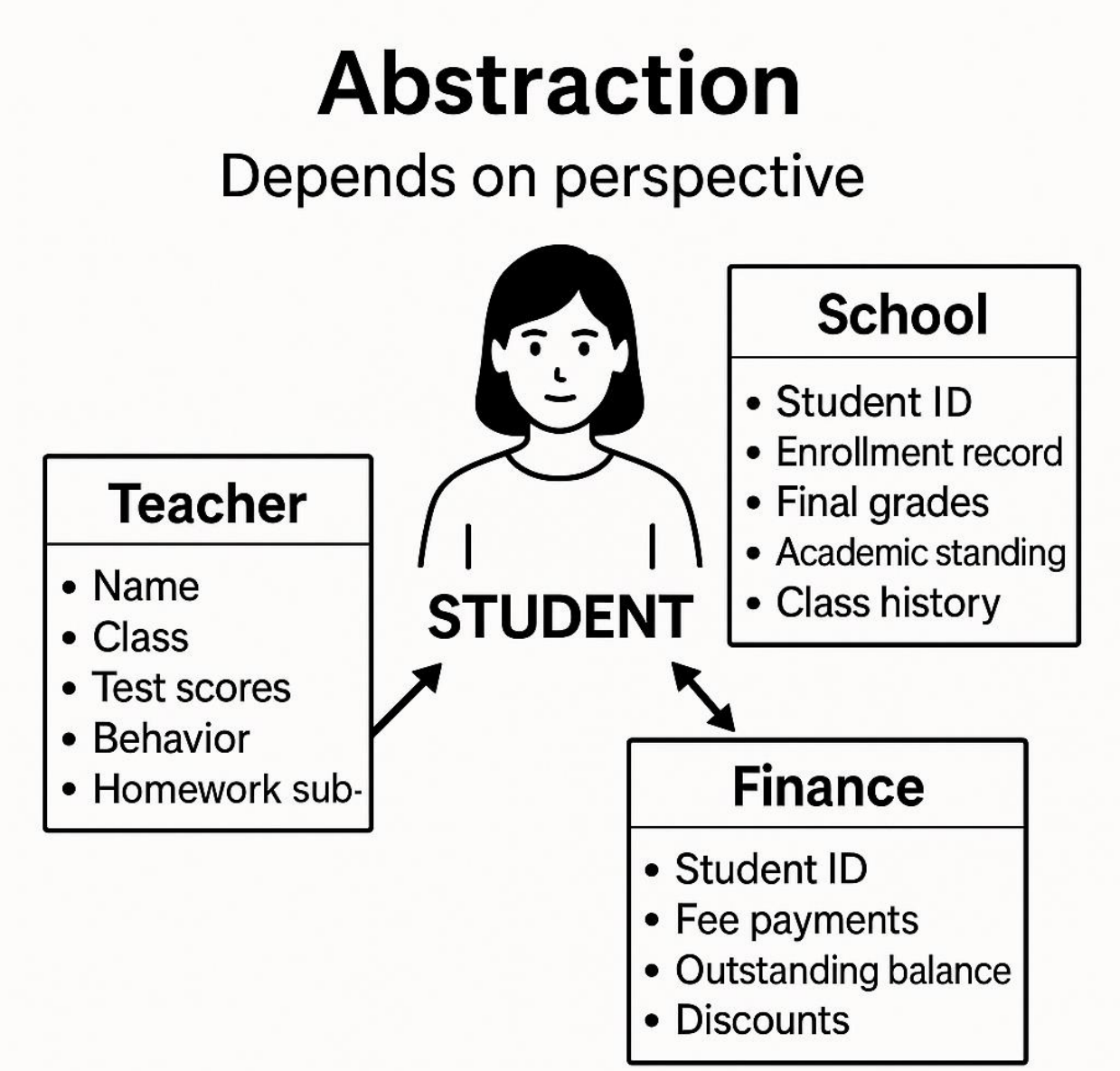
3. Sơ lược về các khái niệm cơ bản

Các nguyên lý cơ bản

- Trừu tượng hóa
  - Là quá trình loại bỏ đi các thông tin cụ thể và giữ lại những thông tin chung.
  - Tập trung vào các đặc điểm cơ bản của thực thể, các đặc điểm phân biệt nó với các loại thực thể khác.
  - Phụ thuộc vào góc nhìn
  - Quan trọng trong ngữ cảnh này nhưng lại không có ý nghĩa nhiều trong ngữ cảnh khác.

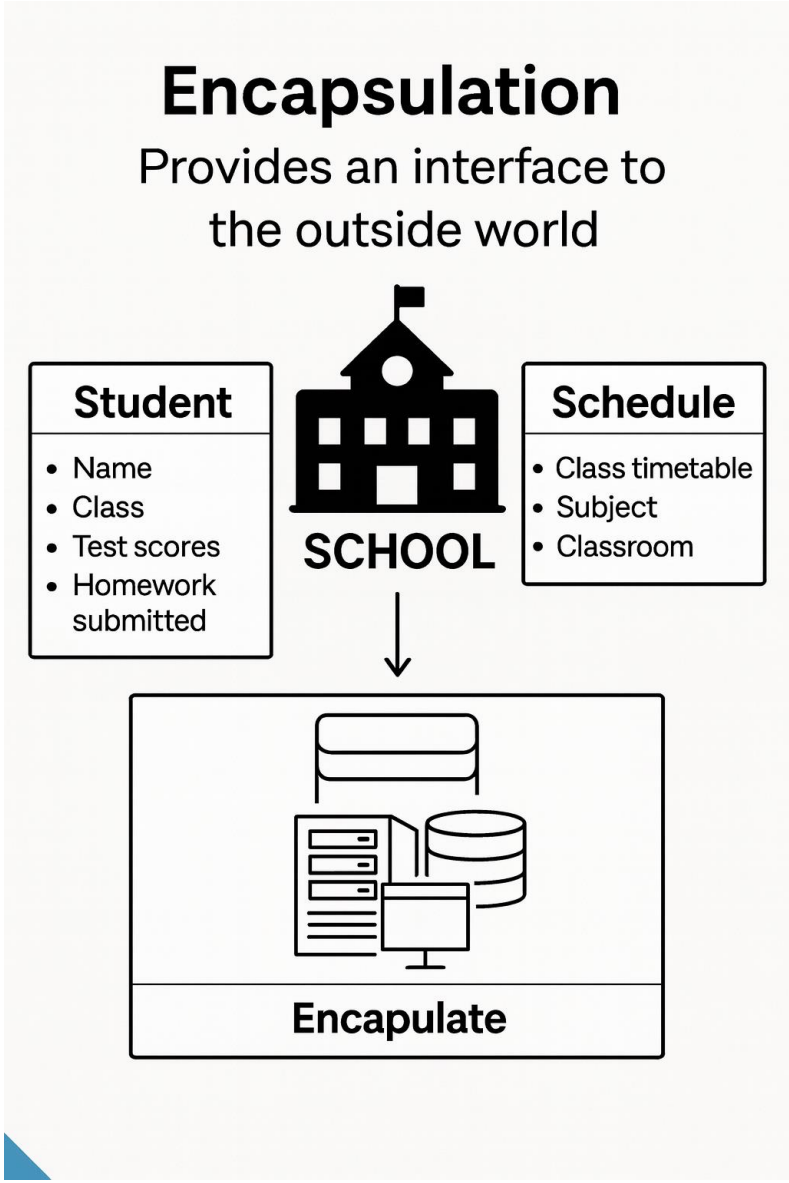
3. Sơ lược về các khái niệm cơ bản

- Trừu tượng hóa
  - Phụ thuộc vào góc nhìn
- VD: Đối tượng trung tâm: Học sinh



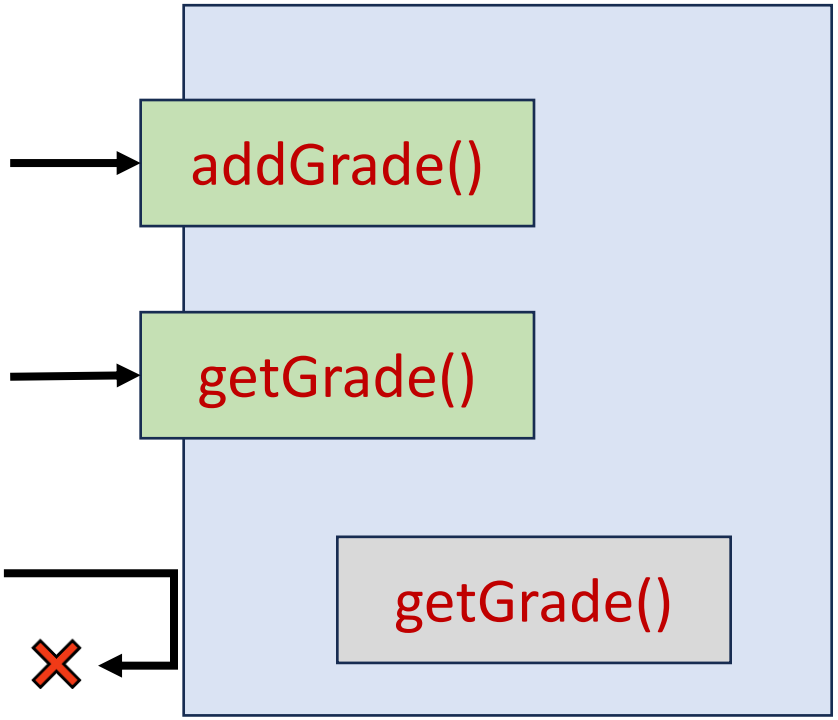
3. Sơ lược về các khái niệm cơ bản

- Đóng gói
  - Che giấu, ẩn đi chi tiết thực hiện bên trong
  - Cung cấp cho thế giới bên ngoài một giao diện
  - Người dùng không phụ thuộc vào việc sửa đổi sự thực thi bên trong



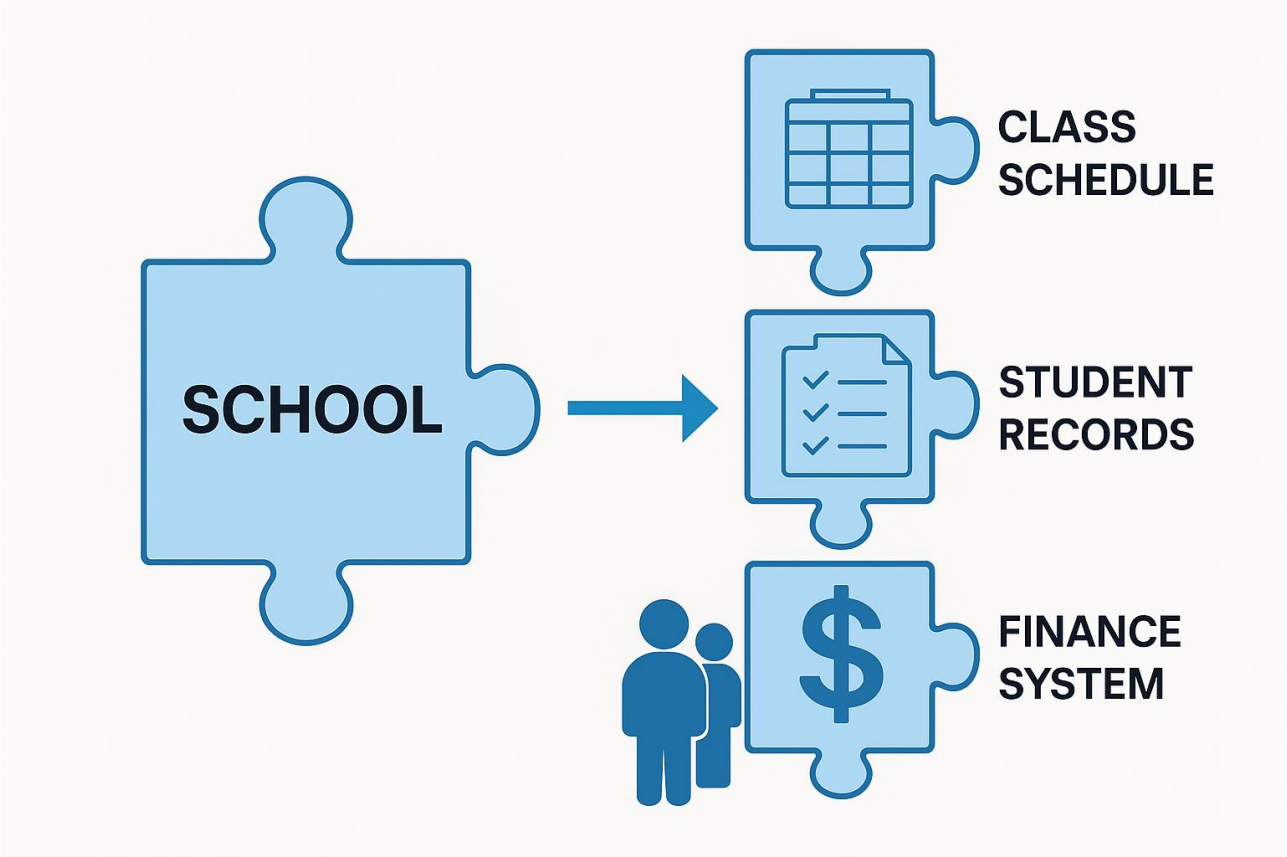
3. Sơ lược về các khái niệm cơ bản

- Đóng gói
  - 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



3. Sơ lược về các khái niệm cơ bản

- Module hóa
  - Chia nhỏ hệ thống phức tạp thành những thành phần nhỏ có thể quản lý được.
  - Cho phép người dùng hiểu được về hệ thống.
  - Chia nhỏ một hệ thống phức tạp thành các mô đun nhỏ hơn.



3. Sơ lược về các khái niệm cơ bản

- Phân cấp
  - Xếp hạng hay xếp thứ tự các mức trừu tượng vào một cấu trúc cây
  - Tổ chức để phân loại. Sử dụng phân cấp rất dễ dàng nhận ra sự giống và khác nhau giữa các đối tượng

4. Phân tích thiết kế hướng đối tượng

- Phương pháp luận (methodology) trong PT&TK 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)

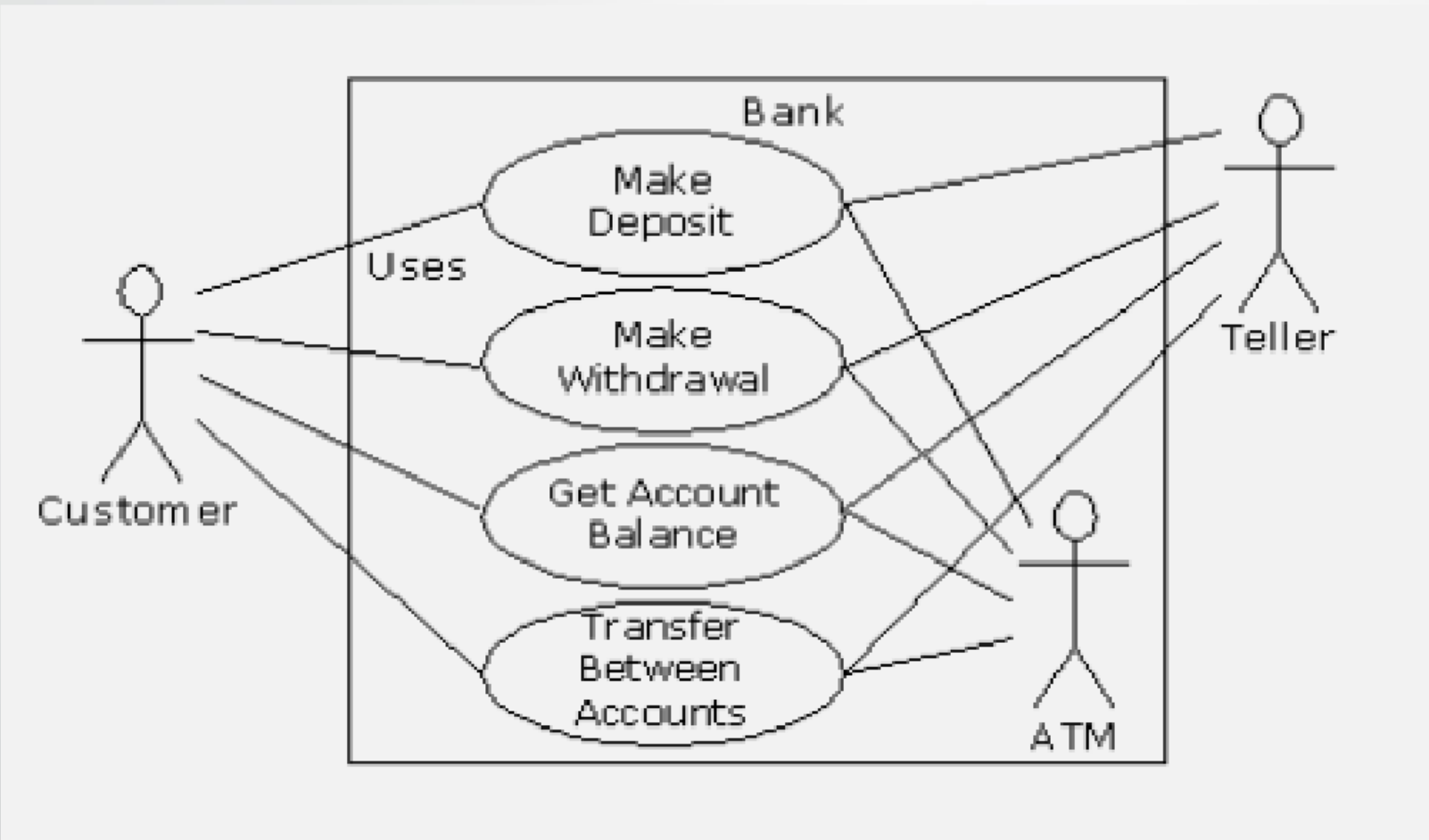
4. Phân tích thiết kế hướng đối tượng

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 PT&TK hướng đối tượng người ta sử dụng các ký pháp và kỹ thuật Use case để mô tả các công việc này

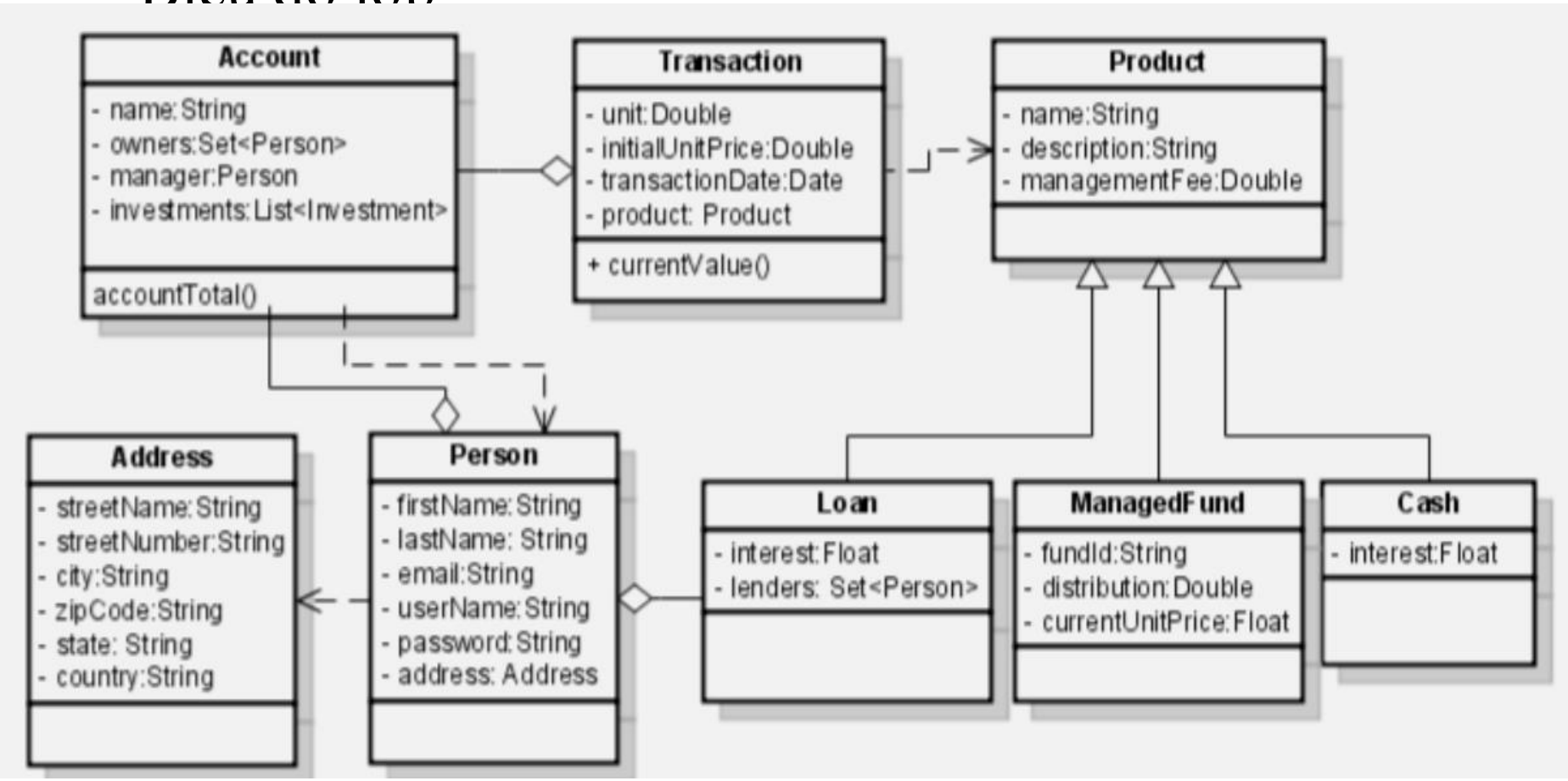
4. Phân tích thiết kế hướng đối tượng

- Biểu đồ Use case



4. Phân tích thiết kế hướng đối tượng

- Biểu đồ lớp




Q
&
A