

## BÀI 15. THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

---

1

### Nội dung

- Mô hình MVC
- Giới thiệu một số biểu đồ UML

2

## 1. MÔ HÌNH MVC

---

3

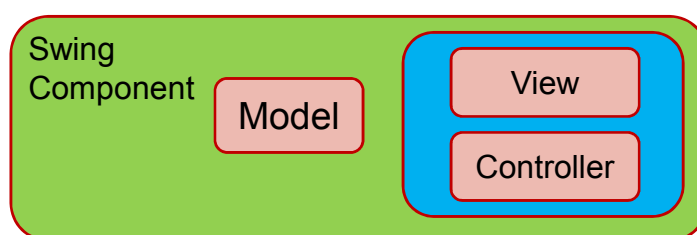
## MVC là gì?

- Mô hình thiết kế phần mềm 3 thành phần: Model – View – Control
- Model:
  - Mô hình hóa các đối tượng chứa dữ liệu cần xử lý
  - Cung cấp các phương thức để truy cập dữ liệu
  - Mô hình hóa các hoạt động nghiệp vụ
- View:
  - Cung cấp giao diện cho người dùng nhập/xuất dữ liệu
  - Kiểm tra tính hợp lệ của dữ liệu vào
  - Bắt các sự kiện trên giao diện
- Controller: nhận các sự kiện được truyền từ View → gọi đến các phương thức tương ứng của Model → hiển thị dữ liệu trả về trên View

4

## Java Swing và MVC

- Java Swing được xây dựng dựa trên mô hình MVC
- Mỗi đối tượng trong Java Swing đóng gói 3 thành phần:
  - Model: chứa dữ liệu và các phương thức thao tác trên dữ liệu đó
  - View: các phương thức để hiển thị đối tượng
  - Controller: bắt và xử lý sự kiện trên đối tượng
- Ví dụ: Xem đoạn mã tạo đối tượng JComboBox sau đây
- Mô hình thực sự của Java Swing Component



5

## Ví dụ

```

// Create JComboBox for setting the count step size
add(new JLabel("Step:"));
final Integer[] steps = {1, 2, 3, 4, 5};
final JComboBox<Integer> comboCount = new
    JComboBox<Integer>(steps);
comboCount.setPreferredSize(
    new Dimension(60, 20));
cp.add(comboCount);
comboCount.addItemListener(new ItemListener() {
    @Override
    public void itemStateChanged(ItemEvent e) {
        if (e.getStateChange() ==
            ItemEvent.SELECTED) {
            step = (Integer)
                comboCount.getSelectedItem();
        }
    }
});
  
```

Model

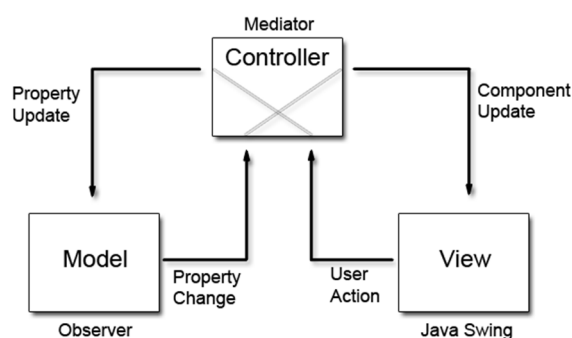
View

Controller

6

## Xây dựng phần mềm theo mô hình MVC

- Khi chương trình phát triển thêm nhiều tính năng, hoặc quá trình xử lý phức tạp hơn, mô hình MVC đóng gói trên đối tượng Swing không còn đáp ứng được.
- Xây dựng phần mềm theo mô hình MVC



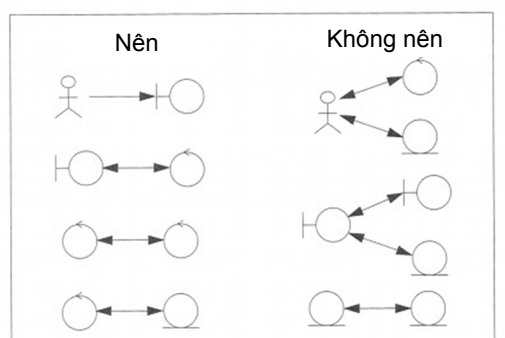
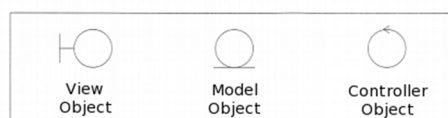
7

## Lợi ích của MVC

- Cho phép phân tách hệ thống lớn thành 3 nhóm thành phần → dễ dàng hơn trong thiết kế, phát triển và bảo trì
- Các thành phần có thể phát triển đồng thời
- Từ một Model có thể hiển thị trên các View khác nhau. Ví dụ: cùng một tập số liệu có thể hiển thị dưới dạng bảng, biểu đồ cột, biểu đồ tròn...
- Để tăng đảm bảo tính cộng tác khi phát triển đồng thời, các lớp cần phải được triển khai từ các giao diện

8

## Giao tiếp giữa các thành phần

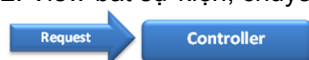


9

## Các bước xử lý yêu cầu người dùng

1. Người dùng thực hiện một hành vi trên View

2. View bắt sự kiện, chuyển yêu cầu cho Controller xử lý



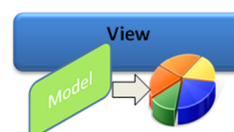
3. Controller gọi phương thức tương ứng mà Model cung cấp



4. Controller nhận kết quả trả về (có thể là một Model chứa dữ liệu) và chuyển cho View để hiển thị



5. View thay đổi khung nhìn và hiển thị kết quả



10

## Một ví dụ

```
public class Student {  
    private String rollNo;  
    private String name;  
    public String getRollNo() {  
        return rollNo;  
    }  
    public void setRollNo(String rollNo) {  
        this.rollNo = rollNo;  
    }  
    public String getName() {  
        return name;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

11

## Một ví dụ (tiếp)

```
public class StudentView {  
    public void printStudentDetails(String studentName,  
                                    String studentRollNo){  
        System.out.println("Student: ");  
        System.out.println("Name: " + studentName);  
        System.out.println("Roll No: " + studentRollNo);  
    }  
}
```

12

## Một ví dụ (tiếp)

```
public class StudentController {
    private Student model;
    private StudentView view;

    public StudentController(Student model,
                               StudentView view){
        this.model = model;
        this.view = view;
    }

    public void setStudentName(String name){
        model.setName(name);
    }
}
```

13

## Một ví dụ (tiếp)

```
    public String getStudentName() {
        return model.getName();
    }

    public void setStudentRollNo(String rollNo){
        model.setRollNo(rollNo);
    }

    public String getStudentRollNo(){
        return model.getRollNo();
    }

    public void updateView(){
        view.printStudentDetails(model.getName(),
                                  model.getRollNo());
    }
}
```

14

## Một ví dụ (tiếp)

```
public class MVCPatternDemo {  
    public static void main(String[] args) {  
        Student model = retrieveStudentFromDatabase();  
        StudentView view = new StudentView();  
        StudentController controller = new  
            StudentController(model, view);  
        controller.updateView();  
        controller.setStudentName("John");  
        controller.updateView();  
    }  
    private static Student retrieveStudentFromDatabase() {  
        Student student = new Student();  
        student.setName("Robert");  
        student.setRollNo("10");  
        return student;  
    }  
}
```

15

## 2. GIỚI THIỆU VỀ UML

---

16



## Unified Modeling Language

- Là một hệ thống các ký pháp để mô hình hóa hướng đối tượng
- Phiên bản hiện tại: UML 2.0
- UML mô hình hóa hệ thống theo các góc nhìn khác nhau:
  - Góc nhìn sử dụng: góc nhìn từ ngoài vào, phản ánh các chức năng hệ thống cần có
  - Góc nhìn thiết kế: góc nhìn cấu trúc logic bên trong hệ thống, phản ánh các nhiệm vụ hệ thống cần thực hiện
  - Góc nhìn quá trình: phản ánh quá trình thực hiện của các thành phần trong hệ thống
  - Góc nhìn cài đặt: phản ánh cách thức lắp ráp các thành phần để hệ thống hoạt động
  - Góc nhìn triển khai: phản ánh vị trí của các thành phần của hệ thống trên các thiết bị vật lý

17

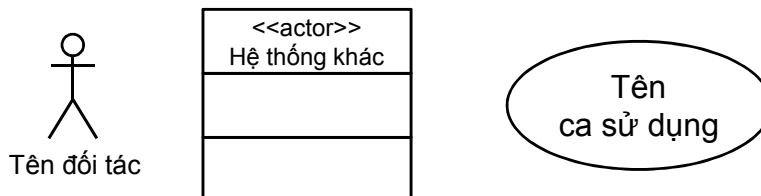
## Các biểu đồ UML chính

- Các biểu đồ cấu trúc:
  - Biểu đồ lớp (Class diagram)
  - Biểu đồ đối tượng (Object diagram)
  - Biểu đồ gói (Package diagram)
  - Biểu đồ thành phần (Component diagram)
  - Biểu đồ triển khai (Deployment diagram)
- Các biểu đồ hành vi:
  - Biểu đồ ca sử dụng (Use-case diagram)
  - Biểu đồ hoạt động (Activity diagram)
  - Biểu đồ máy trạng thái (State machine diagram)
  - Biểu đồ trình tự (Sequence diagram)
  - Biểu đồ giao tiếp (Communication diagram)

18

## Biểu đồ Ca sử dụng(Use-case diagram)

- Ca sử dụng(use case) là một biểu diễn của một tập hợp các chuỗi hành động của hệ thống nhằm cung cấp một kết quả cho một đối tác (actor)
  - Tên ca sử dụng: động từ
- Đối tác: một đối tượng bên ngoài tương tác với hệ thống
  - Tên đối tác: danh từ



19

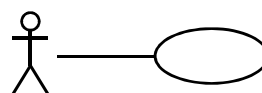
## Biểu đồ Ca sử dụng

- Đặc điểm của ca sử dụng:
  - Phải liên kết với một hoặc một số đối tác
  - Phải dẫn tới một kết quả cụ thể
  - Phải là tập hợp của nhiều chuỗi hành động
- Đặc tả ca sử dụng:
  - Sử dụng biểu đồ Ca sử dụng
  - Sử dụng ngôn ngữ tự nhiên:
    - Mã và tên ca sử dụng
    - Đối tác
    - Điều kiện đầu vào
    - Luồng chính
    - Luồng phụ ( xử lý ngoại lệ)

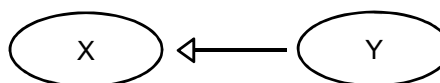
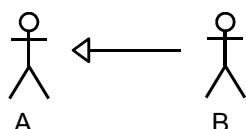
20

## Các liên kết trong ca sử dụng

- Liên kết đối tác và ca sử dụng: \_\_\_\_\_
  - Đối tác và ca sử dụng có trao đổi dữ liệu với nhau
  - Nếu trao đổi là 1 chiều: →



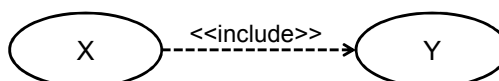
- Khái quát hóa: →
  - Đối tác A là khái quát hóa của B nếu B thừa kế mọi đặc điểm của A
  - Ca sử dụng X là khái quát hóa của Y nếu Y thừa kế mọi đặc điểm của X



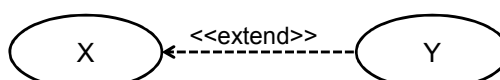
21

## Các liên kết trong ca sử dụng

- Bao hàm (include): Ca sử dụng X bao hàm Y nếu trong quá trình thực hiện X phải thực hiện đầy đủ Y
  - X: ca sử dụng cơ sở



- Mở rộng (extend): Ca sử dụng X thêm vào các bước thực hiện và kết quả của ca sử dụng Y trong điều kiện nào đó
  - X: ca sử dụng cơ sở



- Lưu ý: Ca sử dụng không chỉ ra trình tự các bước thực hiện

22

## Các liên kết trong ca sử dụng

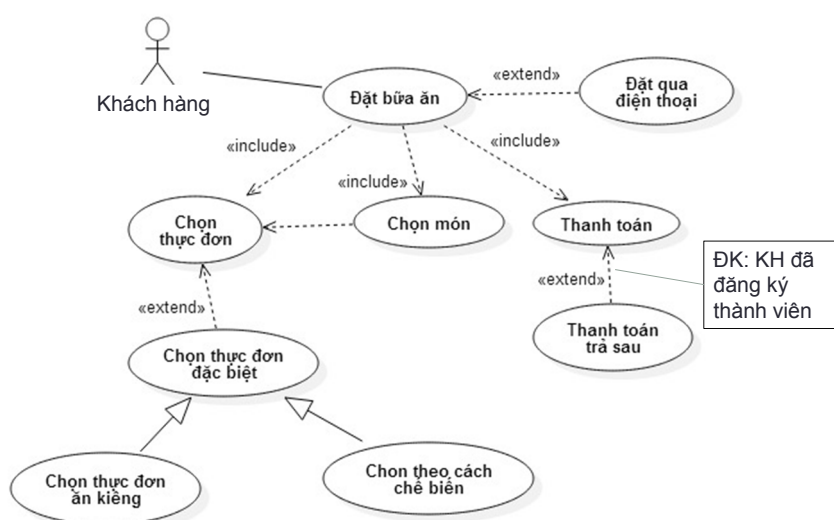
- Phụ thuộc (dependence): Ca sử dụng Y phụ thuộc X khi quá trình thực hiện Y phụ thuộc vào kết quả đã thực hiện X



- Lưu ý: Biểu đồ ca sử dụng không chỉ ra trình tự các bước thực hiện. Nội dung này sẽ được đặc tả bằng ngôn ngữ tự nhiên

23

## Ví dụ - Ca sử dụng đặt bữa tại nhà hàng



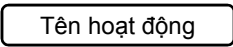

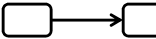


24

## Ví dụ: Đặc tả use-case

<b>UC-01</b>	<b>Khách hàng đặt bữa</b>
Đối tác chính	Khách hàng
Điều kiện	Không có
Luồng chính	<ol style="list-style-type: none"> <li>1. Khách hàng cung cấp các thông tin: Họ tên, Số điện thoại, Thời gian, Số người ăn...</li> <li>2. Hệ thống cung cấp các thực đơn của nhà hàng</li> <li>3. Khách hàng chọn thực đơn. Nếu cần, khách hàng có thể chọn các thực đơn đặc biệt dành cho người ăn kiêng, hoặc theo các chế biến (Nướng, Lẩu...)</li> <li>4. Hệ thống cung cấp các món ăn theo thực đơn khách đã chọn</li> <li>5. Khách hàng hoàn tất việc đặt món</li> <li>6. Hệ thống cung cấp hóa đơn đặt hàng</li> <li>7. Khách hàng xác nhận và thanh toán hóa đơn. Nếu khách hàng đã đăng ký thành viên trên hệ thống có thể chọn tùy chọn thanh toán trả sau</li> <li>8. ...</li> </ol>
Luồng phụ	Nếu khách hàng chưa đăng nhập khi chọn thanh toán trả sau, yêu cầu đăng nhập


25

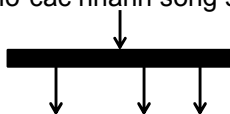
## Biểu đồ Hoạt động(Activity diagram)

- Biểu đồ hoạt động mô tả luồng xử lý của hệ thống khi thực hiện một chức năng nào đó
- Hoạt động: một công việc trong luồng xử lý
- Các ký pháp:
  - Hoạt động: 
  - Luồng điều khiển: 
  - Chuyển từ hoạt động này sang hoạt động khác: 
  - Nút khởi tạo: 
  - Nút kết thúc: 

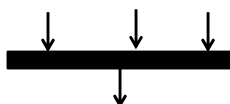
26

## Các ký pháp (tiếp)

- Rẽ nhánh: 
  - Quyết định: một luồng vào, nhiều luồng ra, các luồng ra phải loại trừ nhau
  - Hòa nhập: nhiều luồng vào, một luồng ra, nếu có một luồng vào nào đó xuất hiện, điểm hòa nhập sẽ được vượt qua
- Đồng bộ hóa: biểu diễn các luồng thực hiện song song
  - Nút chạc (fork): mở các nhánh song song



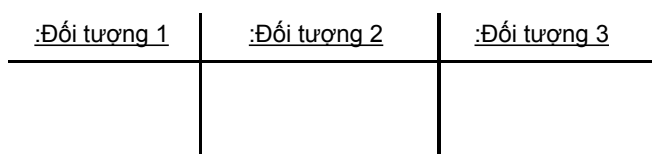
- Nút chụm (join): đóng các nhánh song song



27

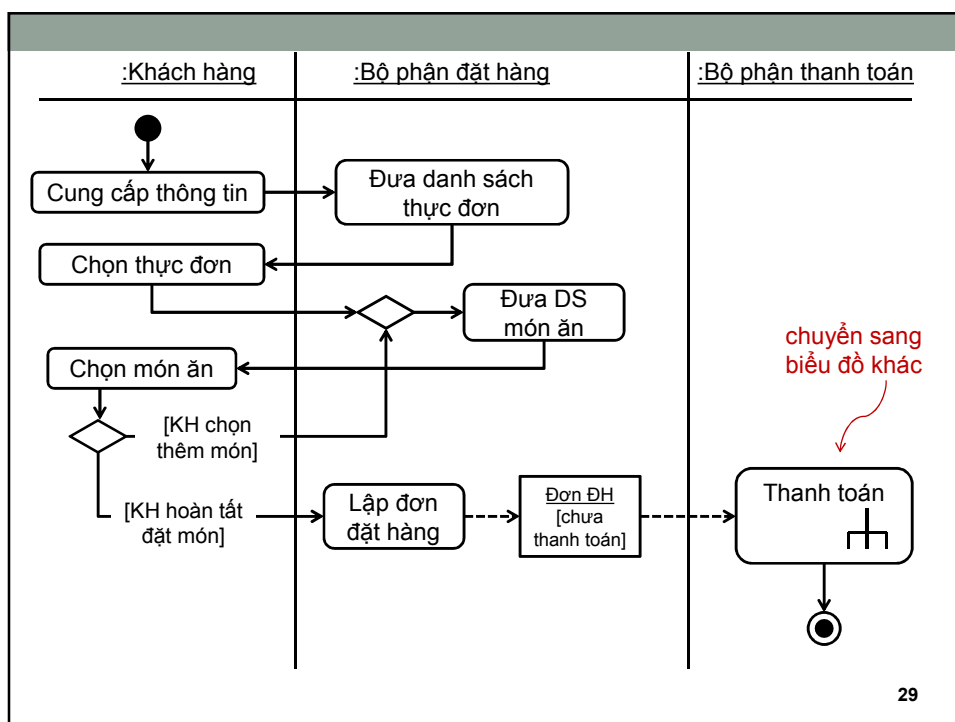
## Các ký pháp

- Điều kiện canh giữ: [Điều kiện]
  - Luồng chỉ được phép đi qua nếu điều kiện xảy ra
- Phân tuyến: sử dụng các đường swim-line để phân định hoạt động cho một (một nhóm) đối tượng thực hiện



- Phân vùng: 

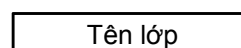
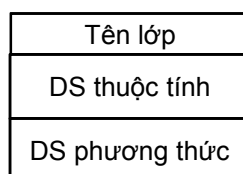
28



29

## Biểu đồ lớp

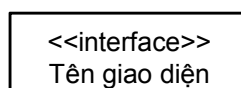
- Biểu diễn lớp



- Biểu diễn chỉ định truy cập:

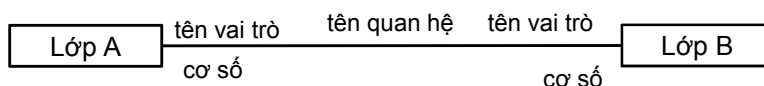
- public: +
- protected: #
- không chỉ định: ~
- private: -

- Biểu diễn giao diện

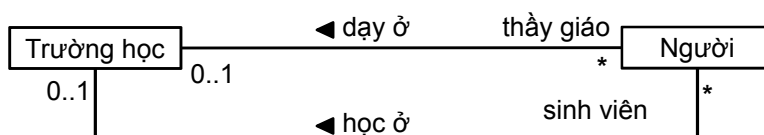


30

## Biểu diễn quan hệ giữa các lớp



- Tên liên kết: chỉ ra mối quan hệ giữa các lớp trên liên kết
- Tên vai trò: chỉ ra vai trò của mỗi lớp trên quan hệ đó
- Cơ số: chỉ ra số lượng các đối tượng của mỗi lớp tham gia vào liên kết:
  - Một số nguyên dương (1, 2, 3..., \*)
  - Một dải giá trị: 0..1, 2..4, 1..\*
  - Không ghi: mặc định là 1



31

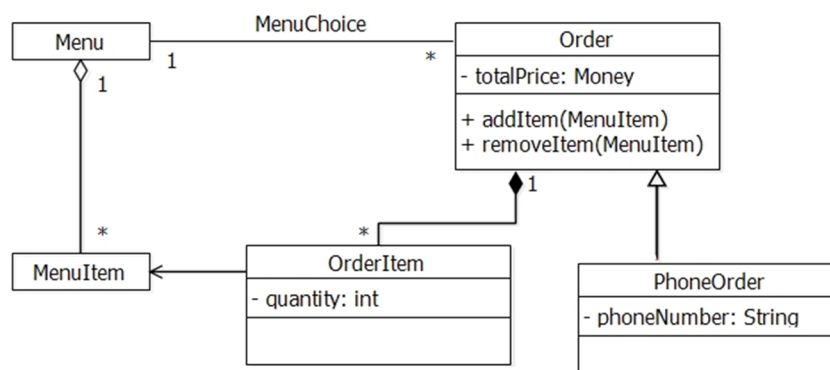
## Biểu đồ lớp – Quan hệ (tiếp)

- Quan hệ kế thừa:
- Quan hệ triển khai:
- Quan hệ kết tập:
- Quan hệ cấu thành:
- Quan hệ phụ thuộc:

32



## Biểu đồ lớp – Ví dụ



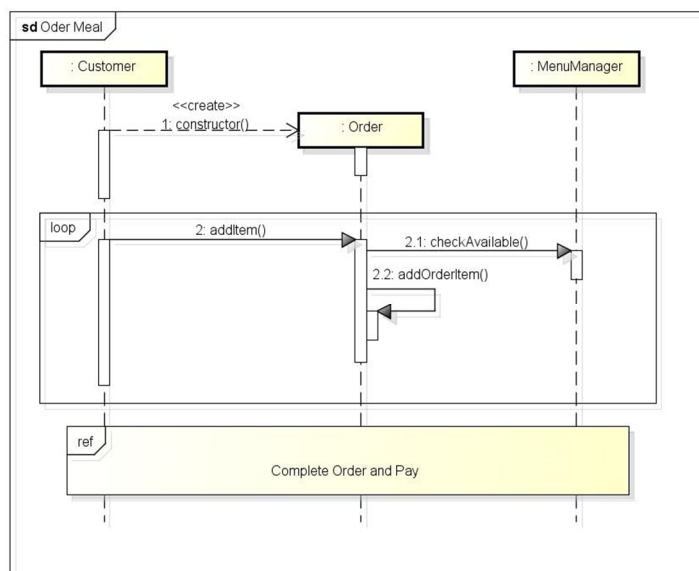
33

## Biểu đồ tuần tự (Sequence diagram)

- Biểu đồ tuần tự: biểu diễn trình tự thực hiện theo thời gian của các thông điệp trao đổi giữa các đối tượng
- Thông điệp gọi:  $\longrightarrow$ 
  - Thông điệp kiểm được (found message): không rõ nguồn  $\bullet \longrightarrow$
  - Thông điệp mất hút (lost message): không rõ đích  $\longrightarrow \bullet$
- Thông điệp được lặp nhiều lần: \*
- Biểu đồ trình tự biểu diễn theo 2 chiều:
  - Chiều ngang: bố trí các đối tượng, thứ tự các đối tượng không quan trọng, song đối tượng khởi tạo quá trình nên nằm bên trái nhất
  - Chiều dọc: trục thời gian hướng xuống dưới, biểu diễn đường đời của đối tượng. Khi đối tượng bị hủy, kết thúc đường đời bằng dấu gạch chéo (X)

34

## Ví dụ



35

powered by Astah

## Tài liệu tham khảo

- Mẫu thiết kế (Design Pattern)
  - Brett D. McLaughlin, et al, *Head First Design Patterns* .
  - <http://www.oodeesign.com>
  - [http://www.tutorialspoint.com/design\\_pattern/](http://www.tutorialspoint.com/design_pattern/)
- Phân tích và thiết kế hướng đối tượng:
  - Brett D. McLaughlin, et al, *Head First Object-Oriented Analysis & Design*.
- Tài liệu UML
  - J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual, 2nd Edition*

36