**ĐỀ CƯƠNG ÔN PTTK-HĐT**

**MỤC LỤC**

[Bài 2. CƠ SỞ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG 4](#_Toc196147435)

[2.1. Sự đóng gói (Encapsulation) 4](#_Toc196147436)

[2.2. Đối tượng (Object) 4](#_Toc196147437)

[2.3. Thông điệp (Message) 4](#_Toc196147438)

[2.4. Lớp (Class) 4](#_Toc196147439)

[2.5. Kế thừa (Inheritance) 4](#_Toc196147440)

[2.6. Đa hình (Polymorphism) 5](#_Toc196147441)

[BÀI 3. TỔNG QUAN VỀ NGÔN NGỮ UML 5](#_Toc196147442)

[3.1. UML 5](#_Toc196147443)

[3.2. Các góc nhìn của UML 5](#_Toc196147444)

[3.3. Các biểu đồ UML chính 5](#_Toc196147445)

[3.4. Câu hỏi 6](#_Toc196147446)

[BÀI 4. MÔ HÌNH HÓA HƯỚNG ĐỐI TƯỢNG 7](#_Toc196147447)

[4.1. Khái niệm mô hình và mô hình hóa 7](#_Toc196147448)

[4.2. Các phương pháp phát triển phần mềm 7](#_Toc196147449)

[4.3. Quản lý dự án phần mềm 7](#_Toc196147450)

[4.4. Câu hỏi 7](#_Toc196147451)

[BÀI 5. XÁC ĐỊNH YÊU CẦU 8](#_Toc196147452)

[5.1. Mục đích 8](#_Toc196147453)

[5.2. Các bước xác định yêu cầu 8](#_Toc196147454)

[5.3. Phân loại yêu cầu 8](#_Toc196147455)

[5.4. Phương pháp thu thập yêu cầu 8](#_Toc196147456)

[5.5. Ghi nhận và đặc tả yêu cầu 9](#_Toc196147457)

[5.6. Câu hỏi 9](#_Toc196147458)

[BÀI 6. MÔ HÌNH CA SỬ DỤNG 9](#_Toc196147459)

[6.1. Biểu đồ ca sử dụng (Use Case Diagram) 9](#_Toc196147460)

[6.2. Mô hình ca sử dụng 10](#_Toc196147461)

[6.3. Cách xác định ca sử dụng 10](#_Toc196147462)

[6.4. Quan hệ giữa các ca sử dụng 10](#_Toc196147463)

[6.5. Mô tả ca sử dụng 10](#_Toc196147464)

[6.6. Câu hỏi 10](#_Toc196147465)

[BÀI 7. BIỂU ĐỒ HÀNH ĐỘNG 11](#_Toc196147466)

[7.1. Biểu đồ hành động (activity diagram) 11](#_Toc196147467)

[7.2. Biểu đồ hành động dùng để miêu tả 11](#_Toc196147468)

[7.3. Các thành phần chính 11](#_Toc196147469)

[7.4. Câu hỏi 11](#_Toc196147470)

[BÀI 8. ĐỐI TƯỢNG VÀ LỚP 12](#_Toc196147471)

[8.1. Đối tượng 12](#_Toc196147472)

[8.2. Lớp 12](#_Toc196147473)

[8.3. Sự phụ thuộc 13](#_Toc196147474)

[8.4. Biểu đồ đối tượng 13](#_Toc196147475)

[8.5. Biểu đồ lớp 13](#_Toc196147476)

[8.6. Các kiểu liên kết giữa các lớp 13](#_Toc196147477)

[8.7. Câu hỏi 13](#_Toc196147478)

[BÀI 9: BIỂU ĐỒ LỚP (CLASS DIAGRAM) 15](#_Toc196147479)

[9.1. Biểu đồ lớp 15](#_Toc196147480)

[9.2. Mục tiêu chính 15](#_Toc196147481)

[9.3. Các bước xây dựng biểu đồ lớp 15](#_Toc196147482)

[9.4. Cách tiếp cận xây dựng biểu đồ lớp 15](#_Toc196147483)

[9.5. Phân tích danh từ 16](#_Toc196147484)

[9.6. Phân loại đối tượng 16](#_Toc196147485)

[9.7. Câu hỏi 16](#_Toc196147486)

[BÀI 10. KỸ THUẬT CRC (CLASS - RESPONSIBILITY - COLLABORATOR) 18](#_Toc196147487)

[10.1. Xác định các chức năng của lớp 18](#_Toc196147488)

[10.2. CRC 18](#_Toc196147489)

[10.3. Cách làm thẻ CRC 18](#_Toc196147490)

[10.4. Nhận xét về CRC 19](#_Toc196147491)

[10.5. Câu hỏi 19](#_Toc196147492)

[BÀI 11. CÁC BIỂU ĐỒ TƯƠNG TÁC 20](#_Toc196147493)

[11.1. Biểu đồ tương tác 20](#_Toc196147494)

[11.2. Biểu đồ trình tự (Sequence Diagram) 21](#_Toc196147495)

[11.3. Biểu đồ giao tiếp (Communication Diagram) 22](#_Toc196147496)

[11.4. Câu hỏi 22](#_Toc196147497)

[BÀI 12. BIỂU ĐỒ TRẠNG THÁI 23](#_Toc196147498)

[12.1. Các loại biểu đồ 23](#_Toc196147499)

[12.2. Thành phần biểu đồ 24](#_Toc196147500)

[12.3. Các loại trạng thái 24](#_Toc196147501)

[12.4. Cách xây dựng biểu đồ trạng thái 24](#_Toc196147502)

[12.5. Câu hỏi 24](#_Toc196147503)

[BÀI 15. CÁC NGUYÊN TẮC THIẾT KẾ 25](#_Toc196147504)

[15.1. Các khái niệm cơ bản trong Lập trình hướng đối tượng 25](#_Toc196147505)

[15.2. Nguyên tắc thiết kế SOLID 25](#_Toc196147506)

[15.3. Mẫu thiết kế (Design Patterns) 26](#_Toc196147507)

[15.4. Các khái niệm GRASP bổ sung 26](#_Toc196147508)

# Bài 2. CƠ SỞ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

## 2.1. Sự đóng gói (Encapsulation)

Dữ liệu + Thao tác -> Nằm trong cũng 1 lớp

Chỉ đối tượng sở hữu mới có quyền chỉnh sửa -> giúp bảo vệ dữ liệu của bạn và hạn chế lỗi

## 2.2. Đối tượng (Object)

Là thực thể có:

* Thuộc tính (attributes): mô tả đặc điểm
* Phương thức (methods): mô tả hành vi

Đối tượng có thể là vật lý, trừu tượng hoặc phần mềm

## 2.3. Thông điệp (Message)

Là cách 1 đối tượng kích hoạt hành vi của đối tượng khác

Thông điệp != Phương thức:

* Thông điệp là lời gọi
* Phương thức là hành vi đc định nghĩa

## 2.4. Lớp (Class)

Là khuôn mẫu định nghĩa các đối tượng có chung đặc điểm

Đối tượng là thể hiện cụ thể của 1 lớp

Lớp giúp:

* Trừu tượng hóa
* Đóng gói
* Che dấu thông tin
* Tái sử dụng

## 2.5. Kế thừa (Inheritance)

Tạo một lớp mới dựa trên một lớp có sẵn

Một lớp có thể kế thừa từ nhiều lớp gọi là đa kế thừa

Tái sử dụng mã nguồn và mở rộng hệ thống dễ dàng

## 2.6. Đa hình (Polymorphism)

Một thông điệp có nhiều cách hiểu

Cho phép nhiều lớp xử lý cùng 1 phương thức theo cách riêng

VD: Animal::speak() có thể gọi Dog::speak() hoặc Cat::speak() tùy đối tượng

Giúp mở rộng hệ thống mà ko cần thay đổi phần mã có sẵn

# BÀI 3. TỔNG QUAN VỀ NGÔN NGỮ UML

## 3.1. UML

UML (Unified Modeling Language) là 1 bộ ký pháp để mô hình hóa hướng đối tượng

## 3.2. Các góc nhìn của UML

Gồm 5 góc nhìn:

* Use case view: góc nhìn từ người dung vào hệ thống
* Logical view: Góc nhìn thiết kế, cấu trúc logic
* Implementation view: Góc nhìn thực thi các thành phần phần mềm
* Process View: Góc nhìn động, mô tả quá trình và tương tác
* Deployment view: Góc nhìn triển khai phần cứng

## 3.3. Các biểu đồ UML chính

* Biểu đồ ca sử dụng (use case)
* Biểu đồ lớp (class)
* Biểu đồ giao tiếp
* Biểu đồ trình tự
* Biểu đồ trạng thái
* Biểu đồ gói
* Biểu đồ thành phần
* Biểu đồ triển khai

## 3.4. Câu hỏi

| **STT** | **Câu hỏi** | **Câu trả lời** | **Ví dụ cụ thể** |
| --- | --- | --- | --- |
| 1 | Hệ thống tương tác với thế giới bên ngoài như thế nào? | Hệ thống tương tác với thế giới bên ngoài thông qua biểu đồ ca sử dụng (Use Case Diagram), mô tả các chức năng mà hệ thống cung cấp cho các tác nhân (actor) bên ngoài như người dùng hoặc hệ thống khác. | Tác nhân “Khách hàng” sử dụng ca “Đặt vé”, “Thanh toán” |
| 2 | Các đối tượng sẽ tương tác với nhau như thế nào? | Các đối tượng trong hệ thống tương tác với nhau thông qua biểu đồ giao tiếp (Communication Diagram), thể hiện các liên kết giữa các đối tượng và các thông điệp được trao đổi qua lại giữa chúng. | UI → Controller → PaymentProcessor |
| 3 | Các đối tượng tương tác theo thứ tự thời gian ra sao? | Các đối tượng trong hệ thống tương tác theo trình tự thời gian thông qua biểu đồ trình tự (Sequence Diagram), giúp theo dõi luồng thông điệp từ trái sang phải theo thời gian từ trên xuống. | Người dùng → nhập mật khẩu → hệ thống kiểm tra → trả kết quả đăng nhập |
| 4 | Các đối tượng có những trạng thái nào? | Các đối tượng có các trạng thái khác nhau và chuyển đổi giữa các trạng thái đó thông qua biểu đồ máy trạng thái (State Machine Diagram), mô tả phản ứng của đối tượng trước các sự kiện. | “Đơn hàng”: Tạo mới → Xác nhận → Đang giao → Hoàn tất |
| 5 | Làm sao để mô-đun hóa hệ thống hiệu quả? | Hệ thống được mô-đun hóa thông qua biểu đồ gói (Package Diagram), dùng để nhóm các thành phần logic thành từng gói (package) nhằm quản lý và tổ chức hệ thống một cách hiệu quả. | Gói: Người dùng – Đặt hàng – Thanh toán – Báo cáo |
| 6 | Các thành phần phần mềm liên quan nhau như thế nào? | Các thành phần phần mềm liên quan với nhau thông qua biểu đồ thành phần (Component Diagram), thể hiện sự phụ thuộc giữa các module như giao diện, xử lý nghiệp vụ, truy xuất dữ liệu… | Component “UI” phụ thuộc “API Service” và “Xử lý hóa đơn” |
| 7 | Phần mềm được triển khai lên phần cứng ra sao? | Phần mềm được triển khai lên hạ tầng phần cứng thông qua biểu đồ triển khai (Deployment Diagram), mô tả các node vật lý như client, server, database và mối quan hệ giữa chúng. | Client → Web Server → App Server → Database Server |

# BÀI 4. MÔ HÌNH HÓA HƯỚNG ĐỐI TƯỢNG

## 4.1. Khái niệm mô hình và mô hình hóa

* Mô hình: Phiên bản trừu tượng của hệ thống thực
* Mô hình hóa: Sử dụng mô hình để diễn tả, phân tích và truyền đạt ý tưởng

## 4.2. Các phương pháp phát triển phần mềm

* Thác nước**(Waterfall):** Phát triển tuần tự, khó thích nghi với thay đổi
* Xoắn ốc **(Spiral):** Kết hợp lặp và quản lý rủi ro.
* RUP **(Rational Unified Process):**

Quá trình lặp và tang trưởng

Gồm 4 pha:

* + Inception (Khời đầu)
  + Elaboration (Triển khai)
  + Construction (Xây dựng)
  + Transition (Chuyển giao)

## 4.3. Quản lý dự án phần mềm

Dự án: tập hợp các công việc để đạt mục tiêu trong phạm vi - thời gian - chi phí

Công cụ:

* WBS
* Gantt chart
* Timeboxing

## 4.4. Câu hỏi

4.4.1. Vòng đời phát triển hệ thống (System Development Life Cycle - SDLC)

Là quy trình các bước để xây dựng hoàn thiện 1 hệ thống phần mềm từ đầu đến cuối.

4.4.2. Những pha chính trong phát triển hệ thống hướng đối tượng?

Có 4 pha chính:

* + Inception (Khời đầu)
  + Elaboration (Triển khai)
  + Construction (Xây dựng)
  + Transition (Chuyển giao)

4.4.3. Mối quan hệ giữa pha (phases) và bước thực hiện (workflows)?

Mỗi pha bao gồm nhiều workflow như: mô hình hóa nghiệp vụ, phân tích, thiết kế, kiểm thử…

Workflows được lặp lại qua các pha (theo vòng lặp).

4.4.4. ****Phân biệt “vòng đời” và “phương pháp phát triển”?****

| **Vòng đời (Lifecycle)** | **Phương pháp phát triển (Methodology)** |
| --- | --- |
| Tổng thể các pha phát triển | Cách làm cụ thể trong từng pha |
| Ví dụ: SDLC, RUP | Ví dụ: Waterfall, Spiral, Agile, XP |

4.4.5. UML và RUP là gì?

UML: Ngôn ngữ mô hình hóa chuẩn để biểu diễn hệ thống (biểu đồ lớp, use case, tuần tự…)

RUP: Quy trình phát triển phần mềm theo hướng đối tượng, dùng UML làm công cụ mô hình hóa.

# BÀI 5. XÁC ĐỊNH YÊU CẦU

## 5.1. Mục đích

Hiểu mong muốn người dùng

Làm cơ sở cho các bước thiết kế, phân tích

## 5.2. Các bước xác định yêu cầu

**Thu thập (Elicitation**):Dùng phỏng vấn, bảng hỏi, họp nhóm...

**Ghi nhận (Recording**):Viết tài liệu, đặc tả yêu cầu

**Thẩm định (Validation**): Xác nhận đúng yêu cầu, tránh sai sót

## 5.3. Phân loại yêu cầu

Yêu cầu chức năng: Những gì hệ thống thực hiện (đăng nhập, đặt hàng…)

Yêu cầu phi chức năng: Tốc độ, bảo mật, khả năng mở rộng…

## 5.4. Phương pháp thu thập yêu cầu

Phỏng vấn: thu thập sâu, linh hoạt

Bảng hỏi: thu thập rộng, dễ thống kê

Họp nhóm, quan sát, nghiên cứu tài liệu

## 5.5. Ghi nhận và đặc tả yêu cầu

Ghi rõ:

* Vấn đề hiện tại
* Mục tiêu hệ thống mới
* Phạm vi hệ thống
* Danh sách yêu cầu (R1, R2, ...)

## 5.6. Câu hỏi

5.6.1. Các bước chính để xác định yêu cầu

* Thu thập
* Ghi nhận
* Thẩm định

5.6.2. Tài liệu cần trước và sau phỏng vấn

* Trước: Kế hoạch phỏng vấn, danh sách câu hỏi
* Sau: Tóm tắt phỏng vấn, đặc tả yêu cầu

5.6.3. Dùng bảng hỏi khi

* Khi cần thu thập ý kiến từ nhiều người
* Khi không thể gặp trực tiếp để phỏng vấn
* Khi cần đánh giá xu hướng chung

5.6.4. Các phần chính của đặc tả yêu cầu

* Vấn đề hệ thống hiện tại
* Mục tiêu hệ thống mới
* Phạm vi hệ thống
* Danh sách yêu cầu

# BÀI 6. MÔ HÌNH CA SỬ DỤNG

## 6.1. Biểu đồ ca sử dụng (Use Case Diagram)

Mô tả hành vi hệ thống từ góc nhìn người dùng

Gồm:

* Tác nhân (Actor): người/vật tương tác với hệ thống
* Ca sử dụng (Use case): chức năng mà tác nhân thực hiện

## 6.2. Mô hình ca sử dụng

Biểu đồ ca sử dụng

Mô tả ca sử dụng (Use case description): Tên, tác nhân, mục tiêu, dòng sự kiện chính/phụ, điều kiện đầu/cuối…

## 6.3. Cách xác định ca sử dụng

* Qua tác nhân: Ai dùng hệ thống → chức năng nào cần?
* Qua kịch bản: Từ tình huống thực tế → chức năng cần hỗ trợ

## 6.4. Quan hệ giữa các ca sử dụng

* <<include>>: Một phần bắt buộc dùng chung
* <<extend>>: Mở rộng có điều kiện

## 6.5. Mô tả ca sử dụng

* Có thể rút gọn (high-level) hoặc đầy đủ (expanded) tùy mục đích
* Mô tả đầy đủ nên áp dụng cho chức năng phức tạp, cần làm rõ logic xử lý

## 6.6. Câu hỏi

6.6.1. Ca sử dụng được xây dựng chủ yếu trong những pha nào?

Chủ yếu trong pha Elaboration (Triển khai), cũng có thể bắt đầu từ Inception

6.6.2. Mô hình ca sử dụng gồm những gì?

* Biểu đồ ca sử dụng
* Mô tả ca sử dụng

6.6.3. Các cách xác định ca sử dụng?

* Qua tác nhân
* Qua kịch bản

6.6.4. Khi nào 1 chức năng đc coi là 1 ca sử dụng?

* Phải là chức năng mang giá trị cho tác nhân, có mục tiêu rõ ràng
* Không nên chia nhỏ quá mức

6.6.5. Mqh giữa kịch bản và ca sử dụng

* Kịch bản mô tả chi tiết các bước của 1 ca sử dụng
* Một ca sử dụng có thể có nhiều kịch bản: điển hình, ngoại lệ,…

6.6.6. Mô tả rút gọn

* Gồm: Tên, tác nhân, mục tiêu, mô tả ngắn gọn
* Dùng khi chỉ cần nhìn tổng thể

6.6.7. Mô tả đầy đủ

* Gồm: Tên, tác nhân, mục tiêu, tổng quan, tiền điều kiện, dòng sự kiện chính, dòng phụ, hậu điều kiện
* Dùng khi muốn lập trình, kiểm thử chi tiết

6.6.8. Các loại quan hệ giữa ca sử dụng

* Loại <<include>>: A bao gồm B, B là thành phần bắt buộc trong A
* Loại <<extend>>: A là thành phần mở rộng có thể xảy ra nếu đk đúng

# BÀI 7. BIỂU ĐỒ HÀNH ĐỘNG

## 7.1. Biểu đồ hành động (activity diagram)

Là công cụ để mô hình các quá trình phức tạp gồm nhiều bước thực hiện

## 7.2. Biểu đồ hành động dùng để miêu tả

* Luồng công việc của hệ thống
* Những hành động trong tựng kịch bản của ca sử dụng
* Các chi tiết hoạt động của 1 chức năng
* Các thuật toán phức tạp

## 7.3. Các thành phần chính

* Hành động (action): thể hiện 1 hành động
* Rẽ nhánh (decision): thể hiện lựa chọn
* Merge: hợp nhất các nhánh
* Fork-join: chia tách - đồng thời và hội tụ - song song
* Object flow: dòng dữ liệu đối tượng
* Swimlanes: phân chia trách nhiệm giữa các thực thể
* Loop: vòng lặp
* Database interaction: kết nối với CSDL

## 7.4. Câu hỏi

7.4.1. Mục đích của biểu đồ hành động

Mô hình hóa chi tiết các bước xử lý trong một hoạt động, thuật toán hay ca sử dụng.

7.4.2. Những loại quá trình có thể đc mô tả = biểu đồ hành động

Quá trình nghiệp vụ, các bước thực hiện chức năng, các thuật toán, xử lý song song và điều kiện rẽ nhánh.

7.4.3. So sánh biểu đồ hành động và sơ đồ khối

Activity diagram mạnh hơn vì có thể mô hình hóa đồng thời (parallel), phân vai (swimlane)

Flowchart thiên về quy trình tuyến tính.

7.4.4. Những yếu tố trong hầu hết các ngôn ngữ lập trình có thể đc mô tả trong biểu đồ hành động

Yếu tố có thể mô tả: → Lặp, điều kiện rẽ nhánh, gọi phương thức, tương tác CSDL, hành động đồng thời.

# BÀI 8. ĐỐI TƯỢNG VÀ LỚP

## 8.1. 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.

Đối tượng là một thực thể độc lập

Một đối tượng gồm

* Định danh (identity): mỗi đối tượng là duy nhất trong bộ nhớ
* Trạng thái (state): định hình bởi giá trị các thuộc tính của đối tượng
* Ứng xử (behaviour): thể hiện bởi các hành động có thể của đối tượng

## 8.2. Lớp

Lớp (class) là một định nghĩa trừu tượng (abstract definition) của các đối tượng có cùng những đặc tính chung

Đối tượng (object) là thể hiện cụ thể (instance) của một lớp

Tác dụng của lớp:

* Trừu tượng hoá dữ liệu (data abstraction)
* Bao gói (encapsulation): dữ liệu + thao tác
* Che giấu thông tin (information hiding)

## 8.3. Sự phụ thuộc

Hai thành phần là phụ thuộc vào nhau nếu có một thành phần sử dụng các dịch vụ cung cấp bởi thành phần kia

## 8.4. Biểu đồ đối tượng

Biểu đồ đối tượng giúp hình dung rõ hơn về các instance cụ thể.

## 8.5. Biểu đồ lớp

Biểu đồ lớp mô tả cấu trúc hệ thống.

## 8.6. Các kiểu liên kết giữa các lớp

Kết hợp (association)

* Các đối tượng của hai lớp có thể tương tác với nhau (thông qua truyền thông điệp – message passing)
* Liên kết có thể được ghi rõ
* Đôi khi có thể ghi rõ vai trò (role name) ở mỗi đầu liên kết

Kết tập (aggregation)

* Thể hiện quan hệ tổng thể - thành phần (whole-part)

Gộp (composition)

* Là một dạng quan hệ tổng thể - thành phần
* Mạnh hơn quan hệ kết tập, trong đó các thành phần không tồn tại riêng rẽ với tổng thể

Kế thừa (inheritance)

* Là quan hệ “is a”, hoặc “is a kind of”

## 8.7. Câu hỏi

8.7.1. Các đặc trưng của một đối tượng?

* Định danh (identity): mỗi đối tượng là duy nhất trong bộ nhớ
* Trạng thái (state): định hình bởi giá trị các thuộc tính của đối tượng
* Ứng xử (behaviour): thể hiện bởi các hành động có thể của đối tượng

8.7.2. Sự khác nhau giữa đối tượng và lớp? Ký hiệu UML của chúng thế nào?

Lớp là bản thiết kế

Đối tượng là thể hiện cụ thể.

UML:

* Lớp là hình chữ nhật với ba phần
* Đối tượng thường ghi rõ tên: ClassName:InstanceName.

8.7.3. Các đối tượng giao tiếp với nhau bằng cách nào?

Gửi thông điệp (message)

8.7.4. Liệt kê 4 loại liên kết giữa các lớp?

* Association
* Aggregation
* Composition
* Inheritance.

8.7.5. Khác nhau giữa kết hợp (association) và kết tập (aggregation)

* Association là liên kết nói chung
* Aggregation là mối quan hệ tổng thể–thành phần (part-whole), nhưng thành phần có thể tồn tại độc lập.

8.7.6. Khác nhau giữa kết tập (aggregation) và gộp (composition)?

* Composition: thành phần không thể tồn tại độc lập với tổng thể.
* Aggregation là mối quan hệ tổng thể–thành phần (part-whole), nhưng thành phần có thể tồn tại độc lập.

8.7.7. Khi nào có thể mô hình một lớp là lớp con của một lớp khác?

Khi lớp con là một "is-a" của lớp cha (ví dụ Cat là một Animal).

8.7.8. Sự khác nhau giữa một hoạt động (operation) và một phương thức (method)?

* Operation: là yêu cầu có thể thực hiện
* Method: là hiện thực cụ thể.

8.7.9. Một số tiêu chuẩn thiết kế của một lớp được coi là tốt?

Problem domain: Trong quá trình phân tích, các lớp nên phản ánh đúng các đối tượng trong phạm vi của bài toán

Functionality: Một lớp cần có cả dữ liệu và các hành vi (ít nhất là trong quá trình phân tích). Nếu một lớp chỉ có các hành vi thì nó nên thuộc vào các lớp khác. Một lớp cũng không nên chỉ có các thuộc tính (dù có thể có thêm một số hàm set/get)

Cohesion: Mỗi lớp nên có tính cố kết cao, chỉ nên liên quan đến một việc chính

Substituability: Khi có kế thừa, đối tượng của lớp dẫn xuất (lớp con) cần có thể thay thế được cho một đối tượng của lớp cơ sở (lớp cha)

# BÀI 9: BIỂU ĐỒ LỚP (CLASS DIAGRAM)

## 9.1. Biểu đồ lớp

Biểu đồ lớp là trung tâm của phân tích thiết kế hướng đối tượng.

## 9.2. Mục tiêu chính

* Định nghĩa kiến trúc phần mềm
* Mô hình hóa cấu trúc hệ thống
* Thể hiện cấu trúc các đối tượng và mối quan hệ của chúng trong hệ thống

## 9.3. Các bước xây dựng biểu đồ lớp

* Xác định đối tượng -> xác định các lớp tương ứng
* Xác định thuộc tính
* Xác định mối quan hệ giữa các lớp
* Phân bổ trách nhiệm của từng lớp (từ đó xác định các hoạt động mà mỗi lớp có thể thực hiện)

## 9.4. Cách tiếp cận xây dựng biểu đồ lớp

Mô hình hóa lĩnh vực:

* Tìm các lớp đối tượng trong phạm vi bài toán qua phân loại đối tượng hoặc qua phân tích danh từ
* Xác định các thuộc tính và các trách nhiệm của từng lớp

Hiện thực hóa các ca sử dụng

* Xem xét từng ca sử dụng của bài toán
* Xác định các lớp cần thiết để hoàn thành từng ca sử dụng đó
* Xem xét các tương tác giữa các đối tượng để xác định hành vi của lớp
* Tổng hợp các lớp từ các ca sử dụng thành biểu đồ lớp chung

## 9.5. Phân tích danh từ

Các bước thực hiện:

* Xác định các danh từ trong:
  + Định nghĩa bài toán
  + Yêu cầu hệ thống
  + Mô tả ca sử dụng
* Gạch chân các danh từ, cụm danh từ có khả năng đại diện cho lớp.
* Loại bỏ các danh từ không phù hợp.
* Phân loại, nhóm các danh từ còn lại thành lớp.

Các lý do loại bỏ danh từ:

* Chỉ có thể chứa giá trị (sẽ trở thành các thuộc tính)
* Bị trùng lặp về ý nghĩa (với một danh từ đã lựa chọn)
* Không rõ ràng hoặc không phản ảnh các đối tượng của bài toán
* Nằm bên ngoài phạm vi hệ thống hoặc là toàn bộ hệ thống

## 9.6. Phân loại đối tượng

Có 2 cách phân loại đối tượng:

* Theo Shlaer và Mellor
* Theo Ross

## 9.7. Câu hỏi

9.7.1. Hãy nêu tầm quan trọng của mô hình hóa cấu trúc hệ thống (structural modeling)?

* Mô hình hóa cấu trúc là trung tâm của quá trình phân tích thiết kế.
* Biểu đồ lớp chính là công cụ mô hình hóa cấu trúc của hệ thống.
* Nó định nghĩa cấu trúc phần mềm, thể hiện cấu trúc các đối tượng, từ đó giúp hiểu và tổ chức hệ thống một cách khoa học.
* Biểu đồ lớp cho thấy mối quan hệ giữa các lớp và là nền tảng để phát triển các thành phần cài đặt về sau.

9.7.2. Nêu các bước chính để xây dựng biểu đồ lớp?

* Xác định đối tượng -> xác định các lớp tương ứng
* Xác định thuộc tính
* Xác định mối quan hệ giữa các lớp
* Phân bổ trách nhiệm của từng lớp (từ đó xác định các hoạt động mà mỗi lớp có thể thực hiện)

9.7.3. Hiện thực hóa ca sử dụng (use case realization) là gì?

Là cách xây dựng biểu đồ lớp

Các bước để hiện thực hóa ca sử dụng:

* Xem xét từng ca sử dụng của bài toán
* Xác định các lớp cần thiết để hoàn thành từng ca sử dụng đó
* Xem xét các tương tác giữa các đối tượng để xác định hành vi của lớp
* Tổng hợp các lớp từ các ca sử dụng thành biểu đồ lớp chung

9.7.4. Khi phân loại đối tượng, hãy liệt kê 4 loại đối tượng thường gặp?

Vật thể hữu hình (Tangible things): Là vật có thể nhìn thấy, sờ được, tồn tại vật lý

VD: xe ô tô, máy tính, điện thoại,…

Vai trò (Roles): Là vai trò mà 1 người hay đối tượng có thể đảm nhận trong ngữ cảnh hệ thống

VD: Mẹ, giáo viên, chính trị gia, khách hàng, người bán hàng,...

Sự kiện (Events): Là những điều xảy ra trong hệ thống hoặc môi trường liên quan đến hệ thống

VD: Hạ cánh, ngắt (interrupt), yêu cầu (request), lỗi xảy ra,...

Tương tác (Interactions): Là các hành động hay sự trao đổi giữa các đối tượng.

VD: Vay mượn, cuộc họp, giao dịch,…

9.7.5. Trình bày về kỹ thuật phân tích danh từ (noun analysis)

Các bước thực hiện:

* Xác định các danh từ trong:
  + Định nghĩa bài toán
  + Yêu cầu hệ thống
  + Mô tả ca sử dụng
* Gạch chân các danh từ, cụm danh từ có khả năng đại diện cho lớp.
* Loại bỏ các danh từ không phù hợp.
* Phân loại, nhóm các danh từ còn lại thành lớp.

9.7.6. Khi phân tích danh từ, hãy liệt kê các lý do để loại bỏ một danh từ trong quá trình tìm ra các lớp?

Các lý do loại bỏ danh từ:

* Chỉ có thể chứa giá trị (sẽ trở thành các thuộc tính)
* Bị trùng lặp về ý nghĩa (với một danh từ đã lựa chọn)
* Không rõ ràng hoặc không phản ảnh các đối tượng của bài toán
* Nằm bên ngoài phạm vi hệ thống hoặc là toàn bộ hệ thống

9.7.7. Hãy liệt kê 4 loại quan hệ (relationships) giữa các lớp?

* Kết hợp
* Kết tập
* Gộp
* Kế thừa

# BÀI 10. KỸ THUẬT CRC (CLASS - RESPONSIBILITY - COLLABORATOR)

## 10.1. Xác định các chức năng của lớp

Dựa trên phân tích động từ:

* Tương tự phân tích danh từ
* Ưu điểm: Đơn giản, dễ thực hiện
* Nhược điểm: Có thể có quá nhiều động từ; Gán vào đúng lớp

Sử dụng kỹ thuật CRC (Class – Responsibility – Collaborator):

* Đề xuất bởi Ward Cunningham và Kent Beck
* Chia chức năng của hệ thống thành trách nhiệm cho các lớp

## 10.2. CRC

CRC là kỹ thuật xác định trách nhiệm của lớp và mối quan hệ cộng tác giữa các lớp.

Gồm:

* Class: tên lớp
* Responsibility: nhiệm vụ
* Collaborator: lớp cộng tác

## 10.3. Cách làm thẻ CRC

Mỗi lớp ghi vào 1 bìa riêng (kích thước 10x15 cm)

Chia thành 3 phần: Class – Responsibility – Collaborator

Viết rõ trách nhiệm và cộng tác

## 10.4. Nhận xét về CRC

Ưu điểm:

* Nhanh chóng xác định được trách nhiệm của từng lớp
* Linh hoạt, dễ dàng trao đổi, hoàn thiện và phù hợp với làm việc nhóm
* Có thể mô tả tương tác giữa các đối tượng

Nhược điểm:

* Không mô tả được chi tiết trình tự tương tác giữa các đối tượng
* Có thể khắc phục bằng cách sử dụng các loại biểu đồ khác

## 10.5. Câu hỏi

10.5.1. Trình bày mục đích của kỹ thuật CRC?

* Xác định trách nhiệm của lớp
* Mối quan hệ cộng tác giữa các lớp.
* Hỗ trợ xây dựng biểu đồ lớp, xác định các phương thức tương ứng với trách nhiệm.

10.5.2. Những yếu tố nào của một lớp có thể nắm bắt bởi CRC?

Tên lớp (Class) – Trách nhiệm của lớp (Responsibility) – Sự cộng tác (Collaborators)

10.5.3. Có thể sử dụng CRC trong làm việc theo nhóm? Cách làm?

Có thể sử dụng CRC trong làm việc theo nhóm

Cách làm:

* Mỗi lớp được viết trên một tờ bìa 10\*15cm
* Các thành viên phụ trách 1 hoặc nhiều thẻ

10.5.4. Ưu điểm và nhược điểm của kỹ thuật phân tích động từ?

Ưu điểm: Đơn giản, dễ thực hiện

Nhược điểm:

* Có thể có quá nhiều động từ
* Khó khăn trong việc gán vào đúng lớp

10.5.5. Ưu điểm và nhược điểm của kỹ thuật CRC?

Ưu điểm:

* Nhanh chóng xác định được trách nhiệm của từng lớp
* Linh hoạt, dễ dàng trao đổi, hoàn thiện và phù hợp với làm việc nhóm
* Có thể mô tả tương tác giữa các đối tượng

Nhược điểm:

* Không mô tả được chi tiết trình tự tương tác giữa các đối tượng
* Có thể khắc phục bằng cách sử dụng các loại biểu đồ khác

10.5.6. Trình bày về hai loại biểu đồ tương tác: tác dụng, đặc điểm?

a. Biểu đồ trình tự (sequence diagram)

* Tác dụng:
  + Thể hiện thứ tự các hành động giữa các đối tượng theo thời gian
  + Giúp hiểu luồng thực hiện trong 1 ca sử dụng
* Đặc điểm:
  + Đọc từ trên xuống dưới theo thứ tự thời gian
  + Gồm: đối tượng (object), đường sống (lifeline), thông điệp (message), vùng hoạt động (activation bar)
  + Hiển thị rõ ai tương tác với ai

b. Biểu đồ giao tiếp (communication diagram)

* Tác dụng
  + Thể hiện các đối tượng tương tác và kết nối giữa chúng
  + Giúp hình dung ai liên kết với ai, dung khi thiết kế sơ bộ
* Đặc điểm
  + Dạng mạng lưới, ko theo thời gian
  + Mỗi liên kết có 1 số thứ tự để biết trình tự
  + Dễ sửa, phù hợp làm nhanh trên bảng, giấy

# ****BÀI 11. CÁC BIỂU ĐỒ TƯƠNG TÁC****

## ****11.1. Biểu đồ tương tác****

Biểu đồ dùng để mô hình hóa sự tương tác giữa các đối tượng trong hệ thống.

Hai loại chính:

* Biểu đồ trình tự (Sequence): mô tả luồng điều khiển giữa các đối tượng theo thời gian.
* Biểu đồ cộng tác (Collaboration): tập trung vào quan hệ giữa các đối tượng và thông điệp trao đổi.

## 11.2. ****Biểu đồ trình tự (Sequence Diagram)****

Xây dựng biểu đồ trình tự:

* Xác định ngữ cảnh (tình huống, ca sử dụng).
* Nhận diện tác nhân và đối tượng tham gia.
* Thiết lập các lifeline cho từng đối tượng.
* Vẽ các thông điệp qua lại giữa các đối tượng.
* Có thể thêm activation (thanh dọc mô tả quá trình xử lý).

Thành phần:

* Tác nhân (actor)
  + Là người hoặc hệ thống bên ngoài hệ thống
  + Gửi/ nhận thông điệp từ biểu đồ
  + Vẽ bằng hình người que hoặc hình chữ nhật có <<actor>>
* Đối tượng (object)
  + Là 1 thực thể trong hệ thống, tham gia vào trình tự
  + Gửi/ nhận thông điệp giống tác nhân
  + Vẽ bằng bình chữ nhật: tenDoiTuong:tenLop
* Đường sống (lifeline)
  + Thể hiện thời gian sống của đối tượng trong trình tự
  + Là đường chấm dọc, có thể kết thúc bằng dấu X nếu bị hủy
* Vùng hoạt động (execution occurrence)
  + Là hình chữ nhật đặt trên lifeline
  + Cho biết khoảng thời gian đối tượng đang xử lý
* Thông điệp (message)
  + Dữ liệu hoặc lời gọi hàm giữa các object
  + Có 2 dạng: Gửi (mũi tên liền) và Trả về (mũi tên đứt)
* Hủy đối tượng (object destruction)
  + Khi 1 đối tượng bị hủy cuối lifeline, ta đánh dấu = dấu X

Tác dụng:

* Mô hình luồng logic trong hệ thống một cách trực quan
* Một trong những loại biểu đồ quan trọng, được sử dụng trong:
  + Phân tích: biểu đồ trình tự hệ thống; Thể hiện trình tự tương tác trong từng ca sử dụng
  + Thiết kế: bổ sung các lớp giao diện, điều khiển và cơ sở dữ liệu

VD: Hệ thống mua vé, xử lý đơn hàng, ...

## 11.3. ****Biểu đồ giao tiếp (Communication Diagram)****

Tác dụng:

* Thể hiện sự tương tác giữa các đối tượng
* Tương đương biểu đồ trình tự
* Ưu điểm: dễ dàng sửa đổi khi thiết kế (trên giấy, bảng)
* Nhược điểm: Biểu đồ cộng tác không mô tả được trình tự thực hiện

Xây dựng biểu đồ giao tiếp

* Xác định ngữ cảnh.
* Nhận diện các đối tượng và tác nhân.
* Thêm thông điệp và đánh số thứ tự.

## 11.4. Câu hỏi

11.4.1. Trình bày về hai loại biểu đồ tương tác: tên gọi, tác dụng, cách xây dựng, các ký hiệu và đặc điểm chính?

|  |  |  |
| --- | --- | --- |
|  | Biểu đồ trình tự | Biểu đồ giao tiếp |
| Tên gọi | Sequence Diagram (TT) | Communication Diagram (GT) |
| Tác dụng | Mô hình hóa luồng logic giữa các đối tượng theo thời gian | Hiển thị các đối tượng và liên kết truyển thông điệp |
| Cách xây dựng | Xác định ngữ cảnh (ca sử dụng).  Nhận diện tác nhân và đối tượng.  Thiết lập đường sống (lifeline).  Thêm thông điệp tương tác.  Thêm các vùng kích hoạt | Xác định ngữ cảnh.  Nhận diện tác nhân, đối tượng  Thêm thông điệp và mỗi liên kết giữa chúng |
| Các ký hiệu | Actor, Object, Lifeline, Message, Execution bar, Object Destruction | Object, Liên kết, Message, Điều kiện/lặp |
| Đặc điểm chính | Tập trung vào thời gian và trình tự thực hiện.  Dễ hình dung luồng logic của hệ thống.  Hữu ích trong phân tích, thiết kế, kiểm thử. | Tập trung vào ai tương tác với ai.  Không thể hiện rõ thời gian như sequence diagram.  Dạng mạng lưới, dễ dùng khi thiết kế nhanh. |

11.4.2. Sự khác nhau giữa biểu đồ trình tự và biểu đồ giao tiếp?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Sequence Diagram (TT) | | Communication Diagram (GT) | | --- |  |  | | --- | |  | |
| Trình tự thực hiện | Rõ ràng (theo trình tự thời gian) | Không rõ ràng, đánh dấu = STT |
| Tập trung vào | Thứ tự tương tác | Cấu trúc liên kết giữa đối tượng |
| Sửa đổi | Khó hơn | Dễ hơn |

11.4.3. Nêu mối quan hệ giữa một kịch bản và biểu đồ hành động tương ứng với nó?

Một kịch bản là mô tả một chuỗi tương tác.

Biểu đồ trình tự chính là cách trực quan hóa kịch bản đó

Nó thể hiện:

* Các đối tượng tham gia.
* Trình tự gửi – nhận thông điệp giữa chúng.

# ****BÀI 12. BIỂU ĐỒ TRẠNG THÁI****

## 12.1. Các loại biểu đồ

* Biểu đồ lớp: Tổ chức, cấu trúc của dữ liệu (các lớp) trong hệ thống
* Biểu đồ trình tự hoặc cộng tác: Sự tương tác giữa các đối tượng
* Biểu đồ trạng thái:
  + Ứng xử của các đối tượng trong một lớp
  + Thể hiện các trạng thái (state) khác nhau của đối tượng và những sự kiện (event) gây ra những thay đổi trạng thái đó

## 12.2. ****Thành phần biểu đồ****

* State (trạng thái): mô tả tình huống hiện tại của đối tượng.
* Event (sự kiện): tác nhân gây thay đổi trạng thái.
* Transition: mũi tên chỉ chuyển từ trạng thái này sang trạng thái khác.

## 12.3. ****Các loại trạng thái****

* Activity state: đang thực hiện hành động nào đó.
* Internal state: trạng thái bên trong.
* Superstate: siêu trạng thái, bao gồm các trạng thái con.

## 12.4. ****Cách xây dựng biểu đồ trạng thái****

* Xác định ngữ cảnh
* Xác định trạng thái đầu và cuối của đối tượng
* Xác định thứ tự các trạng thái (state) mà đối tượng sẽ trải qua
* Xác định các sự kiện (evet), điều kiện liên quan tới các chuyển tiếp.

## 12.5. Câu hỏi

12.5.1. Tác dụng của biểu đồ trạng thái?

Mô tả các trạng thái khác nhau của một đối tượng.

Cho biết những sự kiện nào khiến trạng thái thay đổi.

Giúp hiểu rõ ứng xử động của đối tượng trong hệ thống.

12.5.2. Có thể vẽ biểu đồ trạng thái cho cả hệ thống?

Không nên.

Vì: Biểu đồ trạng thái chỉ áp dụng cho từng lớp hoặc từng đối tượng cụ thể.

VD: Lớp bankAccount có thể có trạng thái “In Credit”, “Overdrawn”,…

12.5.3. Nêu các bước xây dựng biểu đồ trạng thái?

* Xác định ngữ cảnh
* Xác định trạng thái đầu và cuối của đối tượng
* Xác định thứ tự các trạng thái (state) mà đối tượng sẽ trải qua
* Xác định các sự kiện (evet), điều kiện liên quan tới các chuyển tiếp

12.5.4. Nêu sự khác nhau giữa trạng thái (state) và sự kiện (event)?

State: Là tình trạng hiện tại của đối tượng tại một thời điểm

Event: Là sự kiện hoặc hành động xảy ra khiến đối tượng chuyển trạng thái

12.5.5. Khi nào cần có siêu trạng thái?

Khi nhiều trạng thái con có hành vi giống nhau hoặc chuyển đổi tương tự, có thể gom nhóm chúng vào một siêu trạng thái.

Giúp giảm lặp lại, biểu đồ gọn hơn.

12.5.6. Những hệ thống nào thì biểu đồ trạng thái hữu ích nhất?

Những hệ thống có đối tượng có nhiều trạng thái hoặc ứng xử != nhau theo trạng thái.

VD: Máy ATM, máy bán hang tự động, tài khoản ngân hang, giao diện người dung theo từng bước

# BÀI 15. CÁC NGUYÊN TẮC THIẾT KẾ

## 15.1. Các khái niệm cơ bản trong Lập trình hướng đối tượng

* Polymorphism (Đa hình): Một hàm hoặc phương thức có thể hoạt động khác nhau tùy theo đối tượng.
* Inheritance (Kế thừa): Lớp con kế thừa thuộc tính và phương thức từ lớp cha.
* Encapsulation (Đóng gói): Che giấu dữ liệu nội bộ, chỉ cho phép truy cập qua phương thức.
* Abstraction (Trừu tượng hóa): Ẩn chi tiết phức tạp, chỉ thể hiện phần cốt lõi.

## 15.2. Nguyên tắc thiết kế SOLID

Gồm 5 nguyên tắc giúp viết mã rõ ràng, dễ bảo trì:

* **SRP – Single Responsibility Principle**
  + Mỗi lớp chỉ nên có một lý do để thay đổi.
  + Nếu một lớp có nhiều nhiệm vụ → nên tách thành nhiều lớp.
* **OCP – Open/Closed Principle**
  + Lớp mở rộng được nhưng không cần sửa đổi mã nguồn.
  + Có thể thay đổi chức năng mà không làm ảnh hưởng đến phần đang hoạt động.
* **LSP – Liskov Substitution Principle**
  + Lớp con có thể thay thế lớp cha mà không làm hỏng chương trình.

VD: Nếu Bird có phương thức bay, thì Ostrich không nên là con của Bird nếu nó không bay được.

* **ISP – Interface Segregation Principle**
  + Tránh tạo interface "quá lớn".
  + Interface nên chia nhỏ, phù hợp với từng nhóm client khác nhau.
* **DIP – Dependency Inversion Principle**
  + Lớp cấp cao không nên phụ thuộc vào lớp cấp thấp → cả hai nên phụ thuộc vào interface.
  + Chi tiết phụ thuộc vào trừu tượng, không ngược lại.

## 15.3. Mẫu thiết kế (Design Patterns)

**GoF Patterns (Gang of Four)**

* 23 mẫu kinh điển từ 4 tác giả (Gamma, Helm, Johnson, Vlissides).
* Mỗi mẫu gồm: Tên, Vấn đề, Giải pháp.
* Áp dụng trong nhiều ngữ cảnh phần mềm.

**GRASP Patterns (Craig Larman)**

* Các mẫu gán trách nhiệm:
* Low Coupling: Kết nối lỏng lẻo giữa các lớp.
* High Cohesion: Lớp có chức năng thống nhất, không bị phân mảnh.
* Information Expert: Giao trách nhiệm cho lớp “biết rõ nhất”.
* Creator: Lớp tạo đối tượng nên là lớp có quan hệ gần (sử dụng, chứa, ...).
* **Polymorphism:** Giao trách nhiệm xử lý hành vi khác nhau cho lớp con.

## 15.4. Các khái niệm GRASP bổ sung

**Low Coupling (Kết nối lỏng):**

* Hạn chế sự phụ thuộc giữa các lớp.
* Dễ bảo trì, tái sử dụng.

**High Cohesion (Cố kết mạnh):**

* Tập trung một chức năng duy nhất.
* Tránh “lớp làm quá nhiều việc”.

**Information Expert:**

* Lớp nào có **đủ dữ liệu** thì được giao **tính toán, xử lý**.

**Creator:**

* Lớp tạo ra đối tượng nên là lớp chứa hoặc sử dụng đối tượng đó.