

Kỹ nghệ phần mềm

Software Engineering

Đại học Kinh doanh và Công nghệ Hà Nội
Khoa CNTT

GV: Đào Thị Phụng

Email: phuongdt102@gmail.com

Page fb: facebook.com/it.hubt

Phone: 0946.866.817

Bài 7: Thiết kế hướng đối tượng



Nội dung

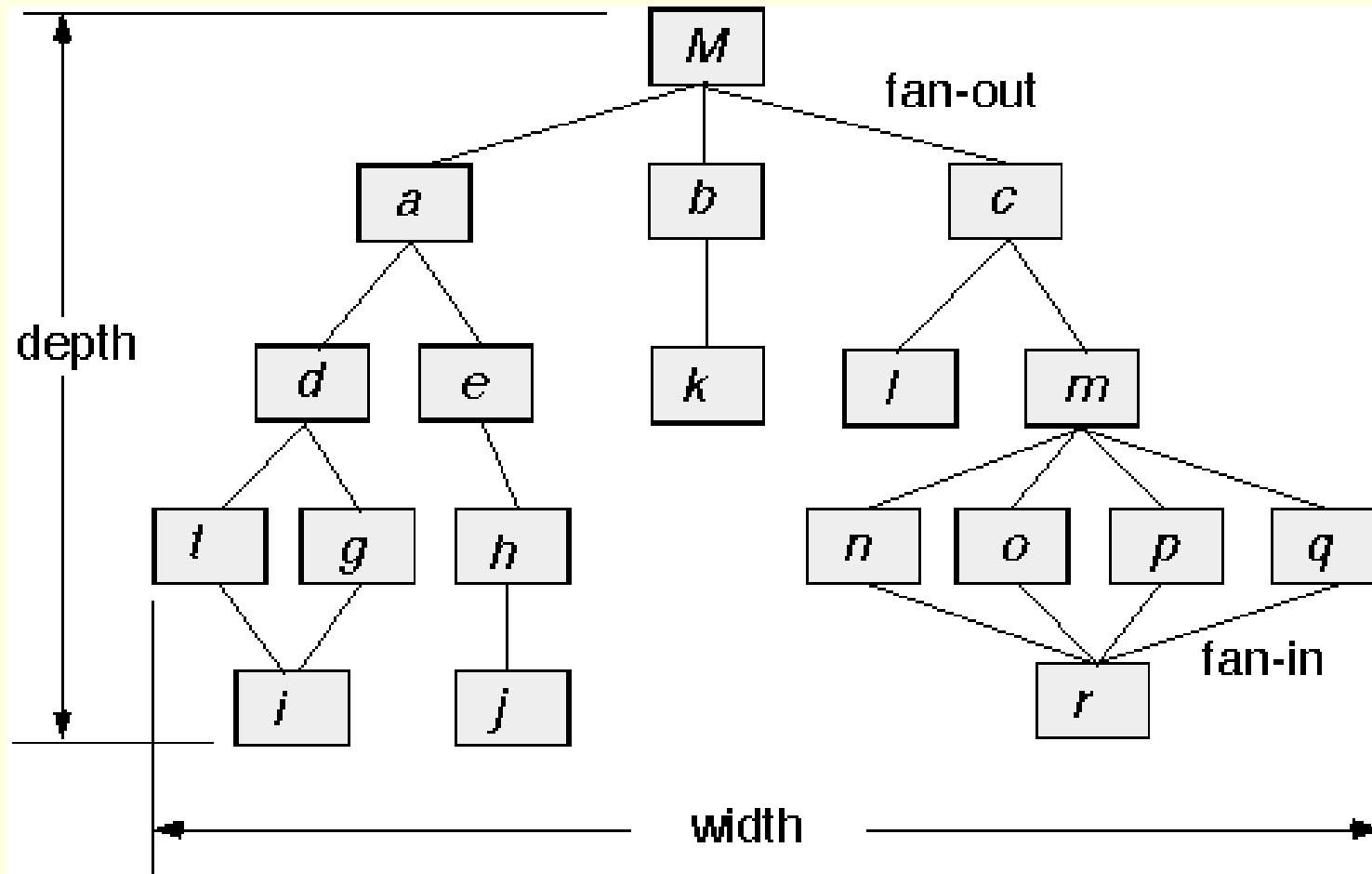
- Vấn đề tồn tại trong hướng kiến trúc
- Khái niệm liên quan đến đối tượng
- Ngôn ngữ UML
- Phân tích hướng đối tượng
- Thiết kế hướng đối tượng
- Sử dụng mẫu thiết kế

TÀI LIỆU THAM KHẢO



1. Nguyễn Văn Vy, Nguyễn Việt Hà. *Giáo trình kỹ nghệ phần mềm*. Nhà xuất bản Đại học Quốc gia Hà nội, 2008
2. Grady Booch, James Rumbaugh, Ivar Jacobson. *The Unified Modeling language User Guid*. Addison-Wesley, 1998.
3. M. Ould. *Managing Software Quality and Business Risk*, John Wiley and Sons, 1999.
4. Roger S.Pressman, *Software Engineering, a Practitioner's Approach*. Fifth Edition, McGraw Hill, 2001.
5. Ian Sommerville, *Software Engineering*. Sixth Edition, Addison-Wasley, 2001.
6. Nguyễn Văn Vy. *Phân tích thiết kế hệ thống thông tin hiện đại. Hướng cấu trúc và hướng đối tượng*, NXB Thống kê, 2002, Hà Nội.

Kiến trúc phần mềm truyền thống



Vấn đề của thiết kế hướng thủ tục



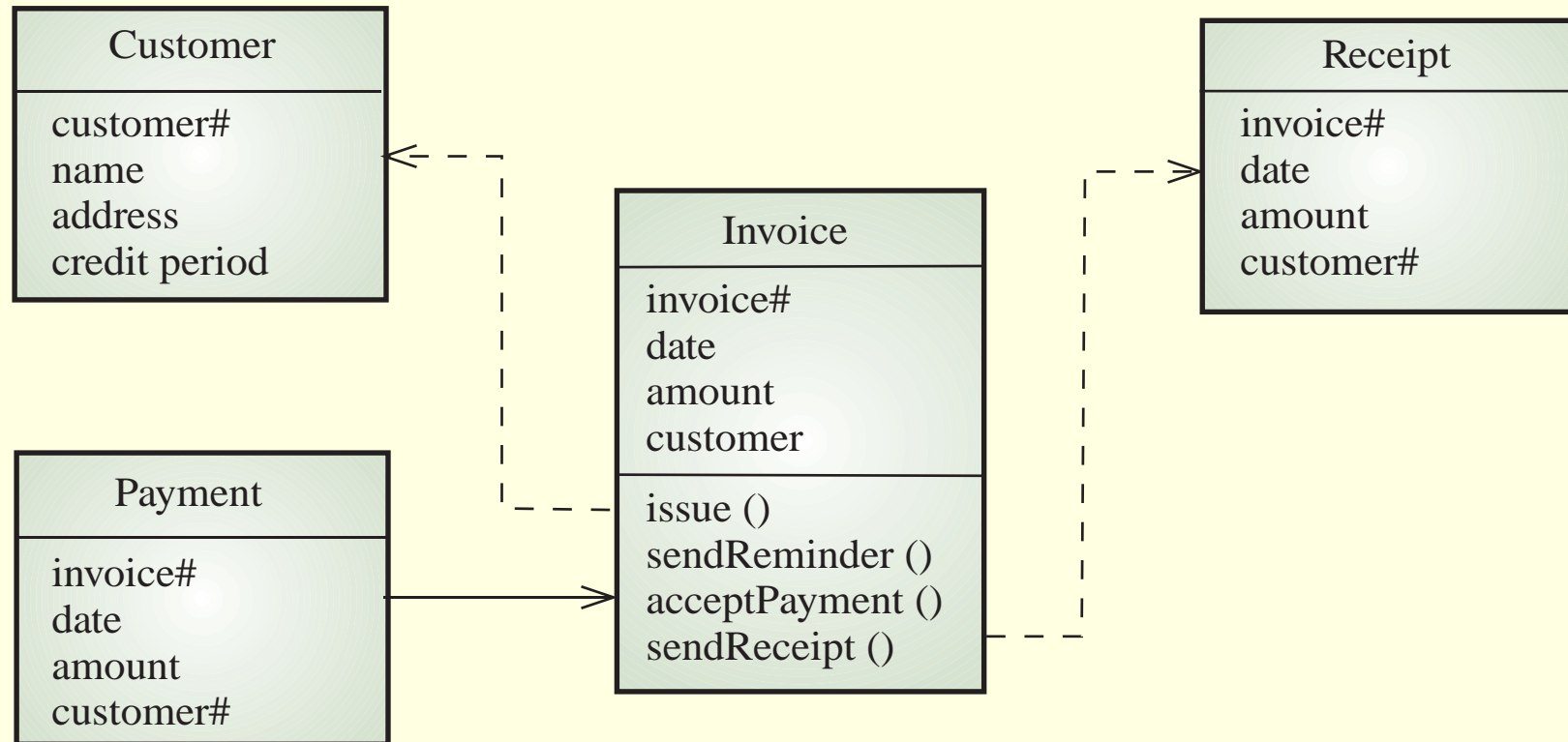
- Dữ liệu là chung cho cả hệ thống
 - ◆ Mọi thủ tục thao tác trên CSDL chung, đặc trưng cho trạng thái toàn hệ thống
 - ◆ Thao tác sai của 1 thủ tục lên dữ liệu gây sai lan truyền sang phần khác sử dụng dụng dữ liệu này
 - ◆ Sửa đổi 1 thủ tục có nguy cơ ảnh hưởng tới phần khác liên quan
- Thay đổi cấu trúc dữ liệu dẫn đến thay đổi tổng thể hệ thống → dữ liệu cần tổ chức tốt
- Hệ thống lớn, phức tạp: bảo trì khó khăn

Thiết kế hướng đối tượng- OOD



- Hiện đang trở nên phổ biến
- Là một cách tiếp cận khác, nhìn nhận hệ thống theo các quan điểm:
 - ◆ tập các đối tượng có tương tác với nhau
 - ◆ mỗi đối tượng bao gói cả dữ liệu và các xử lý trên chúng
 - ◆ Tương tác giữa các đối tượng bằng truyền thông báo
 - ◆ Các đối tượng có thể kế thừa nhau

Ví dụ kiến trúc hướng đối tượng



Ưu điểm của OOD



- Dễ bảo trì: các đối tượng được hiểu như các thực thể hoạt động độc lập
 - ◆ Bao gói thông tin
 - ◆ liên kết lỏng lẻo (trao đổi bằng truyền thông báo)
- Dễ tái sử dụng:
 - ◆ độ độc lập cao
 - ◆ có khả năng kế thừa
- Dễ hiểu: một vài hệ thống, có sự ánh xạ tường minh giữa thực thể thực thể giới thực và đối tượng hệ thống

Nội dung của OOD



- Xác định các tập đối tượng (gọi là lớp) và các đặc trưng của chúng
- Phân định vai trò và trách nhiệm của chúng trong hệ thống
- Thiết lập được sự tương tác của chúng để thực hiện chức năng của hệ thống phần mềm đặt ra

Các khái niệm của OOD



d1. Đối tượng

- Là các trừu tượng hóa thực thể của thế giới thực hoặc của một hệ thống
- Bao gồm: **định danh, các thuộc tính và các phương thức** thao tác trên các dữ liệu thuộc tính của nó
- Độc lập và đóng gói trạng thái thể hiện bằng giá trị các thuộc tính của nó ở một thời điểm
- Cung cấp dịch vụ cho đối tượng khác hay yêu cầu các đối tượng khác thực hiện một dịch vụ

Lớp đối tượng

- Lớp đối tượng là khuôn mẫu để tạo ra tập đối tượng có các đặc trưng chung
- Lớp đối tượng có thể kế thừa thuộc tính và dịch vụ từ lớp đối tượng khác
- Lớp được xác định bằng:
 - ◆ Tên
 - ◆ Bộ các thuộc tính
 - ◆ Các phương thức



Trừu tượng hóa



- Trừu tượng hóa cung cấp cái nhìn đơn giản đối với thực thể cần xử lý
 - ◆ chỉ mô tả các tính chất chúng ta quan tâm
 - ◆ che giấu các thông tin không cần thiết
- Được cài đặt như là:
 - ◆ kiểu dữ liệu trừu tượng, lớp đối tượng
 - ◆ các kiểu dữ liệu cơ sở có sẵn: int, double

Bao gói và che giấu thông tin



- Là khái niệm cơ sở của thiết kế/lập trình hướng đối tượng
- Che giấu thông tin để đối tượng không cần thiết không thể sử dụng
- Chỉ cung cấp chức năng, dịch vụ cần
- Che giấu các yếu tố có khả năng thay đổi
 - ◆ cấu trúc dữ liệu
 - ◆ cách thức cài đặt

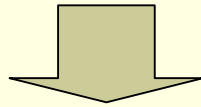
Cấu trúc dữ liệu



- Mô tả mối quan hệ giữa các khoản mục dữ liệu
- Cấu trúc vật lý chỉ ra cách thức chương trình thao tác với dữ liệu
- Các cấu trúc cơ sở
 - ◆ khoản mục vô hướng
 - ◆ mảng (một chiều, nhiều chiều)
 - ◆ danh sách móc nối
 - ◆ cây phân cấp

Ví dụ

```
struct Date {  
    int year, mon, day;  
};
```



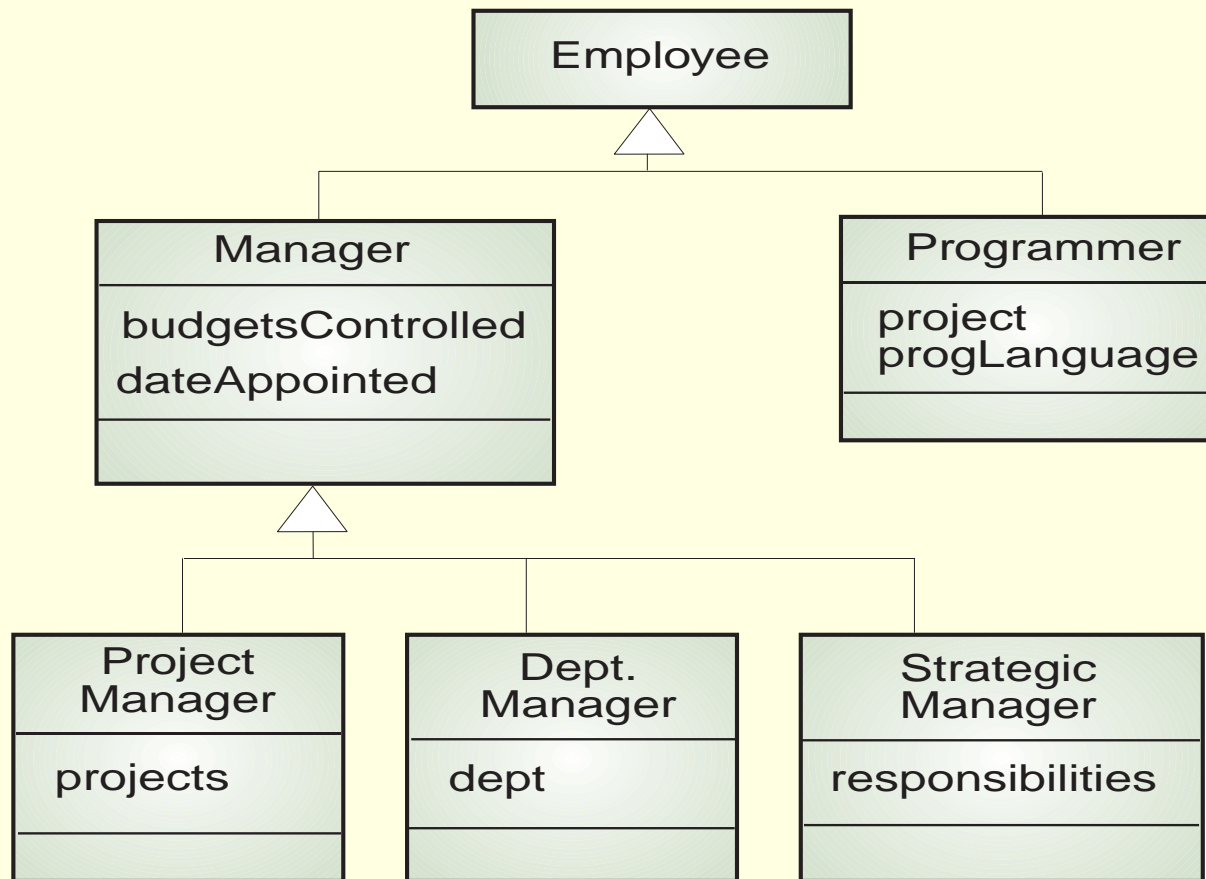
```
struct Date {  
    int year;  
    int mon_and_day;  
}
```

Tổng quát hóa và kế thừa



- Một lớp có thể là lớp con của 1 lớp các đối tượng tổng quát hơn, và gọi là kế thừa của lớp tổng quát
- trên cây kế thừa, lớp cha (*super class*) có thể là tổng quát hóa của 1 số các lớp con (*sub-class*)
- Lớp con kế thừa các thuộc tính và phương thức của lớp cha và có thể thêm/thay đổi phương thức, thuộc tính
- Sử dụng kế thừa giúp ta mô tả lớp con chỉ gồm các đặc trưng khác lớp cha

Ví dụ cây kế thừa

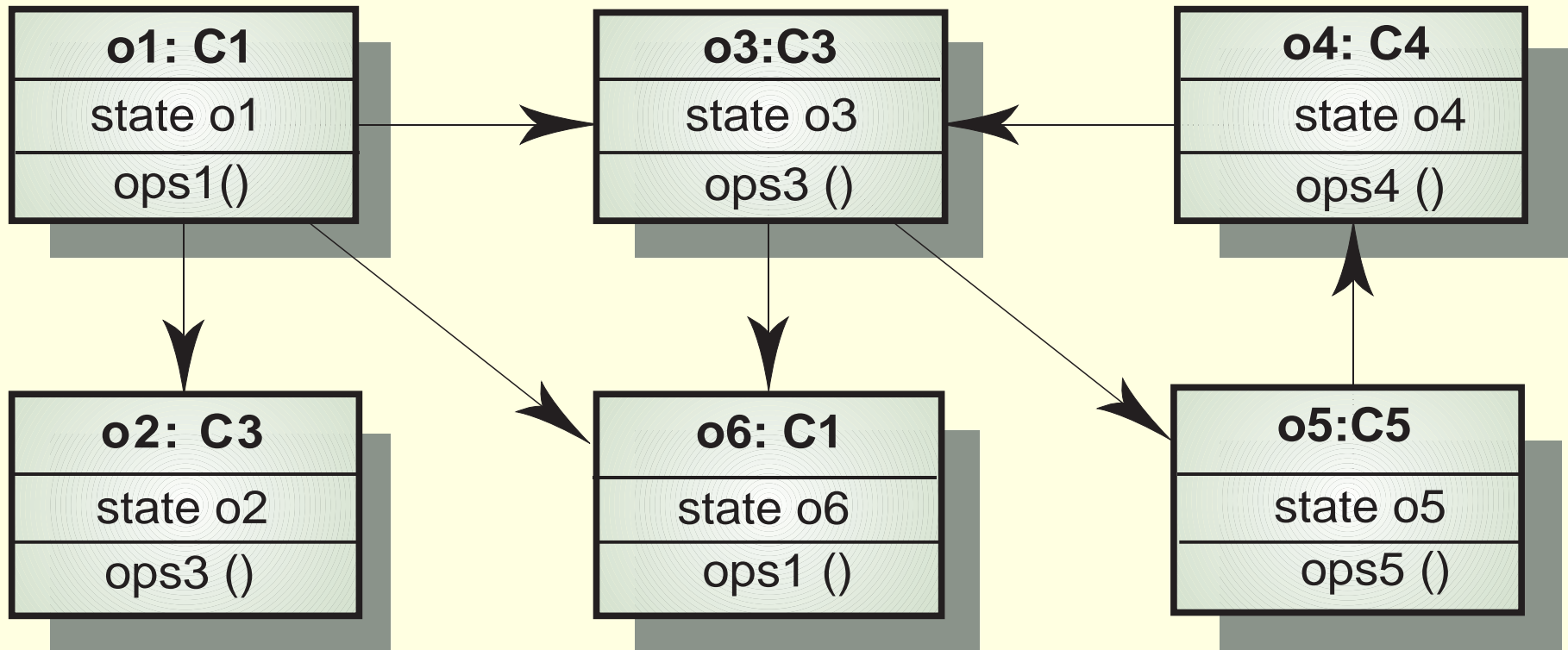


Ưu điểm của kế thừa



- Là cơ chế trừu tượng để phân loại các lớp (thực thể)
- Tái sử dụng cả ở mức thiết kế và mức lập trình
 - ◆ tái sử dụng cấu trúc dữ liệu
 - ◆ tái sử dụng phương thức:
 - giao diện
 - cài đặt (mã)
- Biểu đồ kế thừa là nguồn thông tin mang tính tổ chức về bài toán

Tương tác giữa các đối tượng



Tương tác giữa các đối tượng



- Các đối tượng giao tiếp bằng trao đổi thông báo
- Thông báo
 - ◆ Tên dịch vụ được yêu cầu
 - ◆ Thông tin dùng để thực hiện dịch vụ
- Các loại đối tượng
 - ◆ actor: chỉ gửi thông báo
 - ◆ agent: gửi và nhận thông báo
 - ◆ server: chỉ nhận thông báo
- Thực tế, thông báo được cài đặt bằng lời gọi hàm
 - ◆ Tên = tên hàm
 - ◆ Thông tin = danh sách tham số

Một số vấn đề dùng kế thừa



- Đối tượng không tự chứa, không thể hiểu nếu không tham chiếu đến lớp cha
- Lạm dụng các sơ đồ kế thừa trong bước phân tích có thể dẫn đến sự kém hiệu quả

Kế thừa và OOD



- Có hai quan niệm
 1. Kế thừa là yếu tố cơ bản của OOD và được cài đặt thông qua các ngôn ngữ hướng đối tượng
 2. Kế thừa sử dụng cơ chế đặc biệt để tái sử dụng thuộc tính và phương thức. Thiết kế kế thừa sẽ tạo ra các ràng buộc cho cài đặt
- Kế thừa không dễ hiểu và do đó nên tránh, nhất là đối với các hệ thống đặc biệt quan trọng.

e. Ngôn ngữ mô hình hóa thống nhất *Unified Modeling Language*



- Là một ngôn ngữ mô hình để phát triển phần mềm hướng đối tượng
- Các đặc trưng: UML là ngôn ngữ:
 - độ họa
 - làm trực quan hóa
 - đặc tả
 - xây dựng mô hình
 - làm tài liệu

Ngôn ngữ mô hình hóa thống nhất



■ Gồm 3 khối cơ bản:

A. Các sự vật (things)

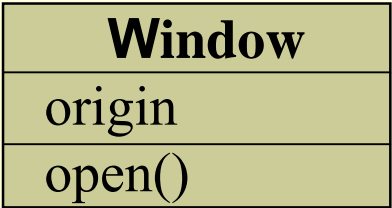


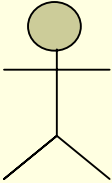
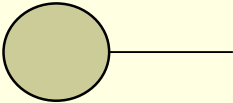
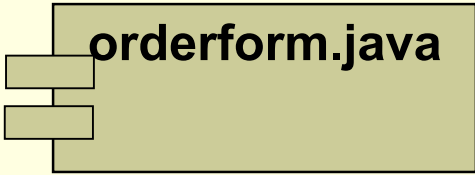
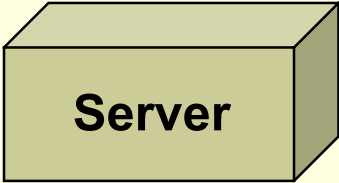
1. Các sự vật cấu trúc (structural)
2. Các sự vật hành vi (behavioral)
3. Các sự vật nhóm gộp(grouping)
4. Các sự vật giải thích (annotational)

B. Các quan hệ (Relationships)

C. Các biểu đồ (Diagrams)

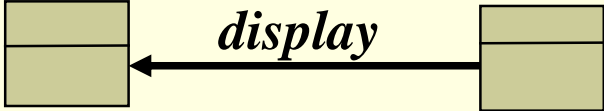

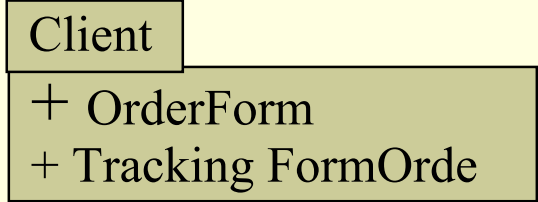
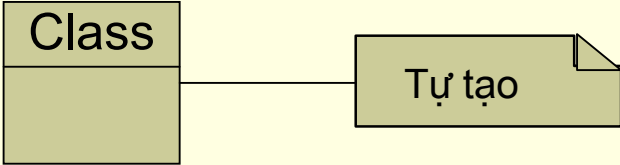
Ngôn ngữ mô hình hóa thống nhất

■ Các sự vật cấu trúc (structural)

Class – <i>lớp</i>		use ase – <i>ca sử dụng</i>	Collaboration <i>sự cộng tác</i>
			
Actor	Interface	Component- <i>thành phần</i>	Node - <i>nút</i>
			

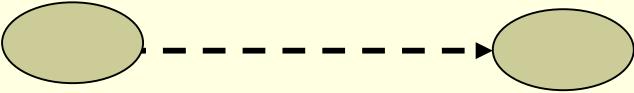
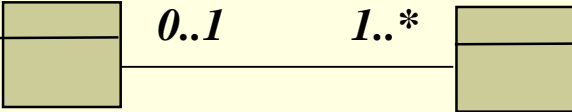
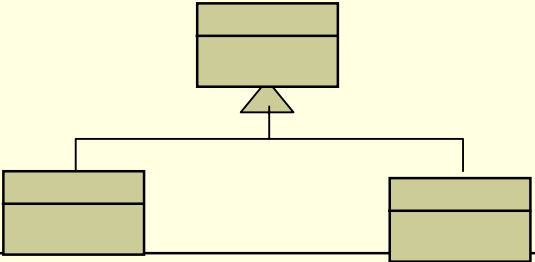
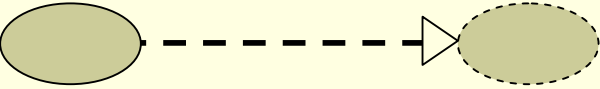
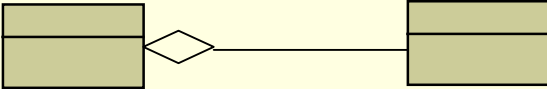
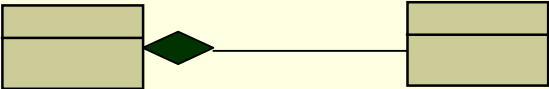
Ngôn ngữ mô hình hóa thống nhất

■ Các sự vật hành vi , nhóm gộp, giải thích

interaction	state machine	package
		
Note		
		

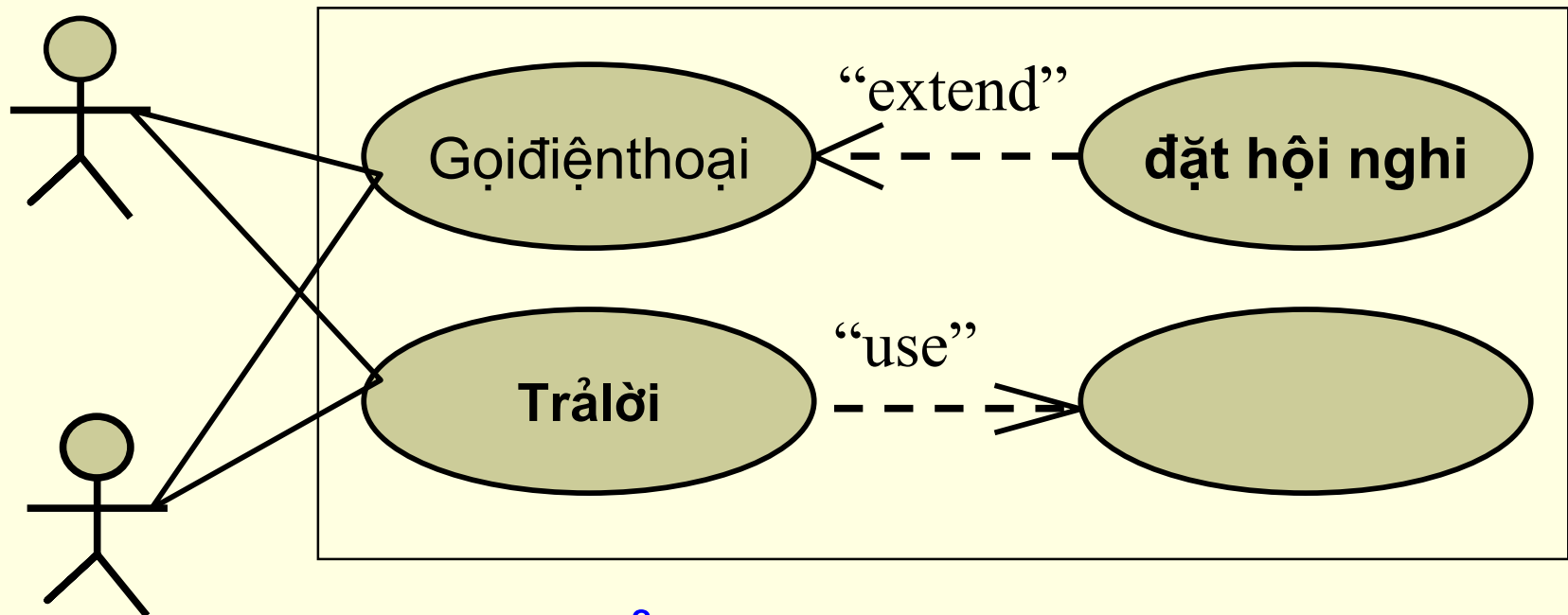
Ngôn ngữ mô hình hóa thống nhất

B. Các mối quan hệ

dependence	association	generalization
		
realization	aggregation	composite
		

Ngôn ngữ mô hình hóa thống nhất

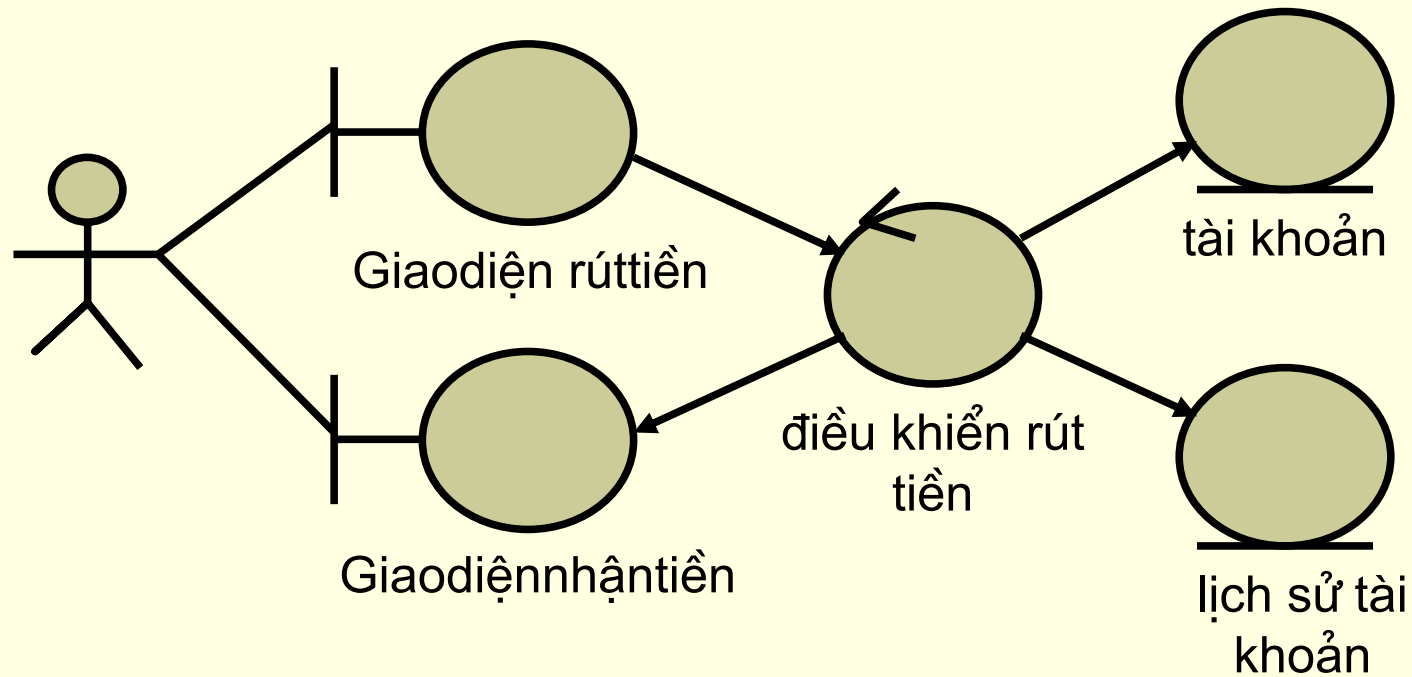
C. Các biểu đồ



Biểu đồ cửa sổ sử dụng

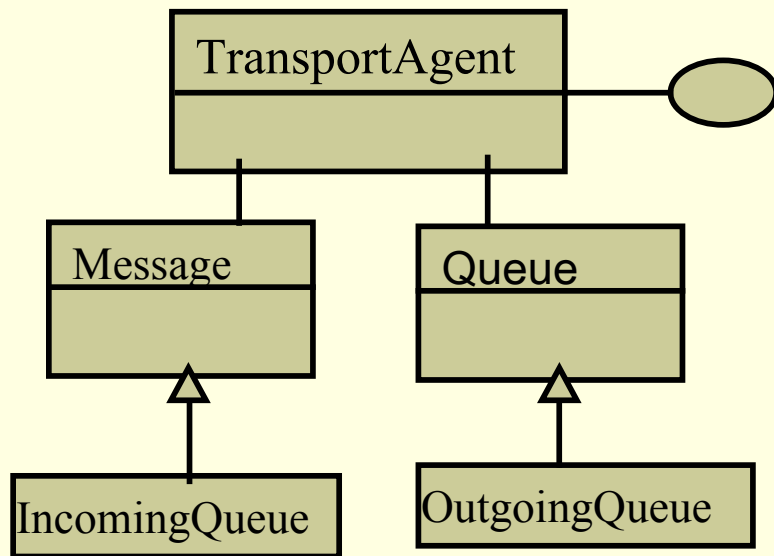
Ngôn ngữ mô hình hóa thống nhất

C. Các biểu đồ

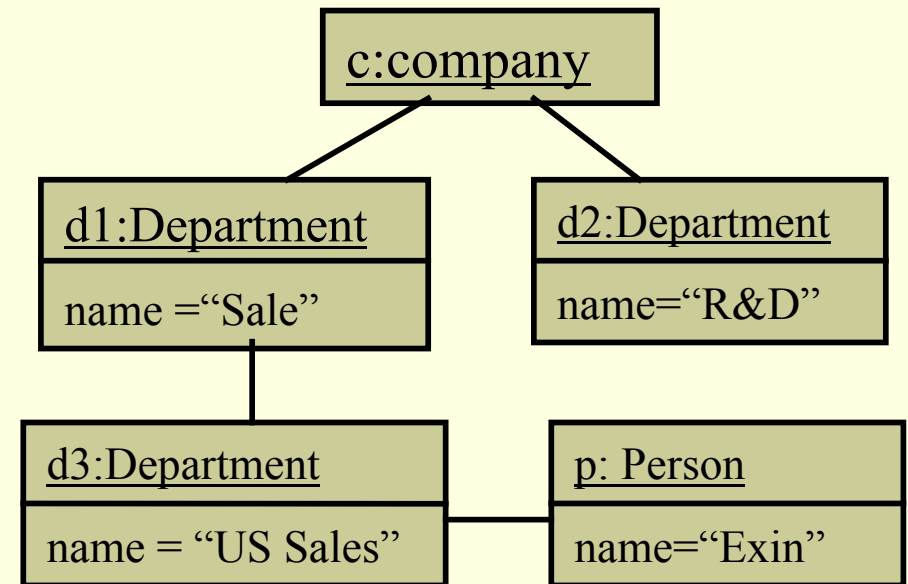


Biểu đồ cửa công tác phân tích

Ngôn ngữ mô hình hóa thống nhất



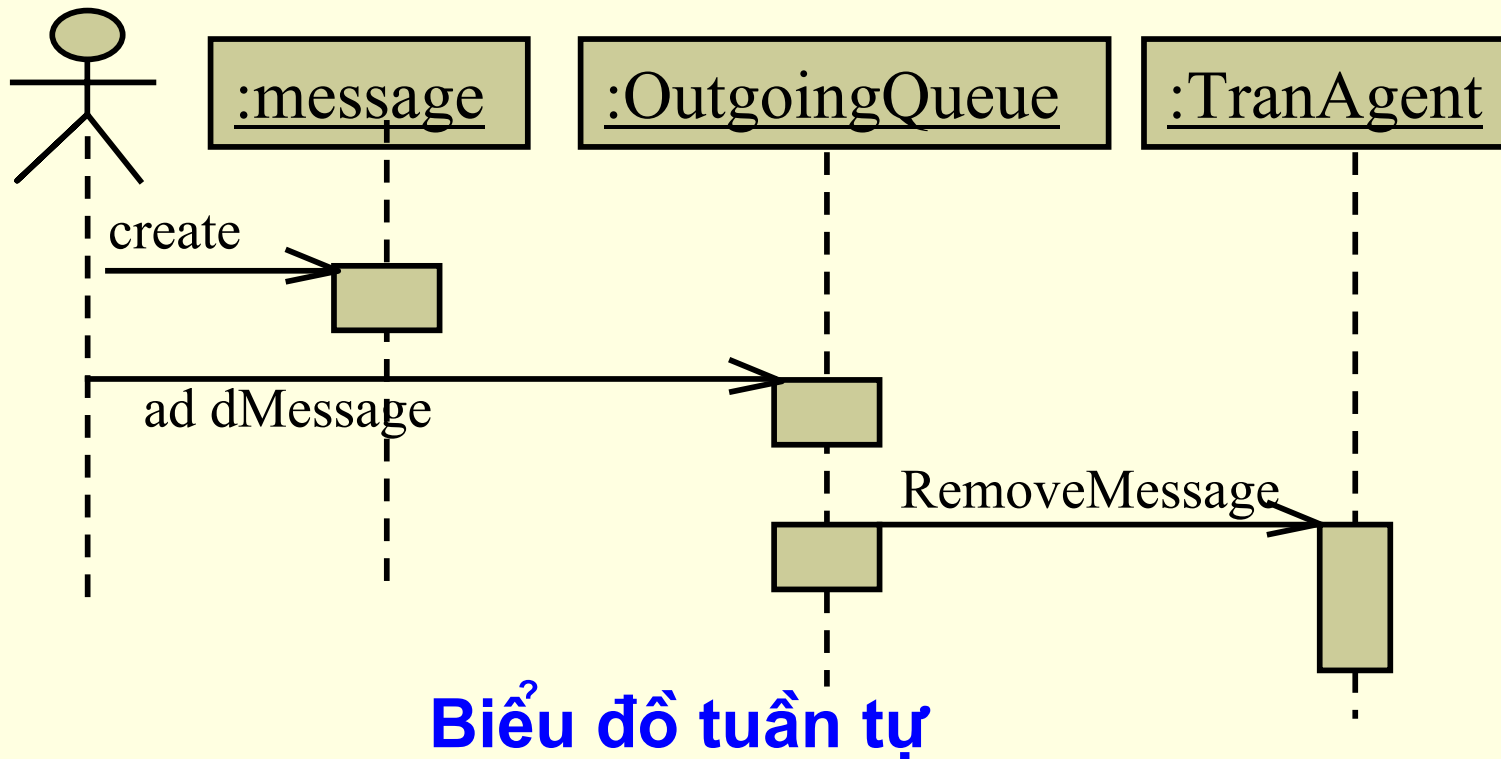
Biểu đồ lớp



Biểu đồ đối tượng

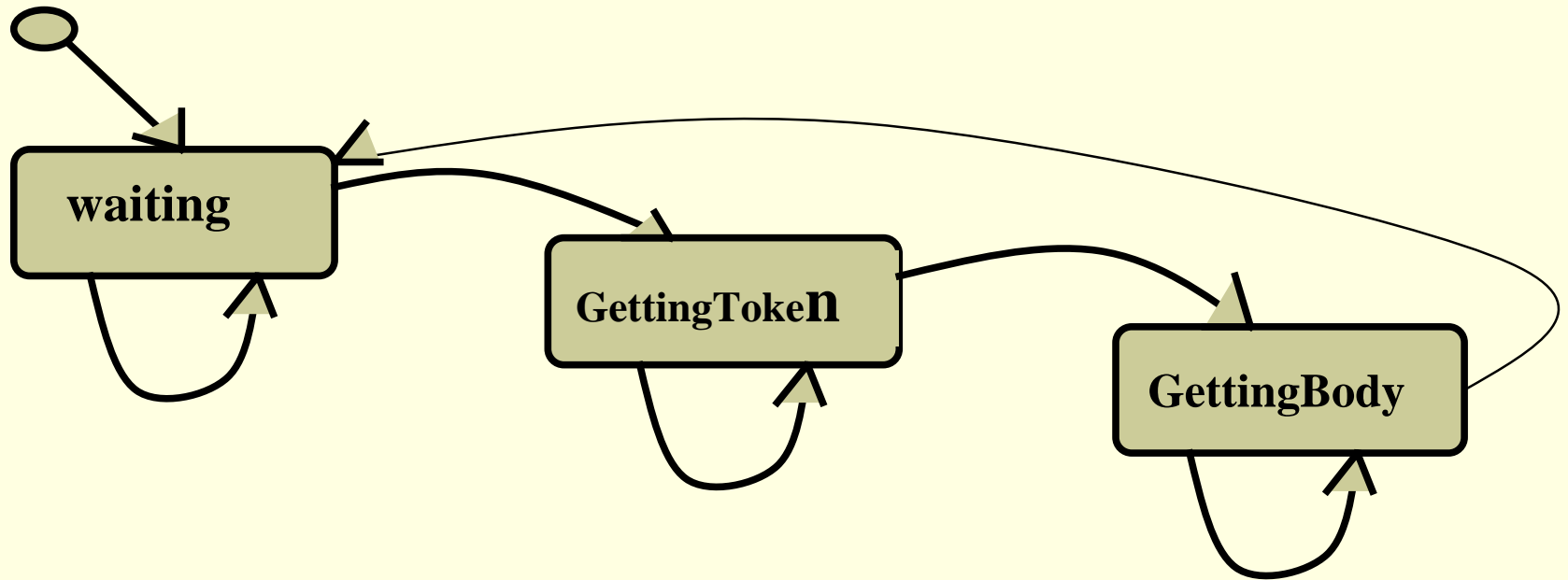
Ngôn ngữ mô hình hóa thống nhất

C. Các biểu đồ



Ngôn ngữ mô hình hóa thống nhất

C. Các biểu đồ



Biểu đồ trạng thái

Phân tích/thiết kế hướng đối tượng



■ Mô hình phân tích

■ Mô hình nghiệp vụ

- Mô hình miền
- Biểu đồ hoạt động

■ Mô hình ca sử dụng

■ Mô hình lớp phân tích

■ Mô hình gói lớp

■ Mô hình thiết kế

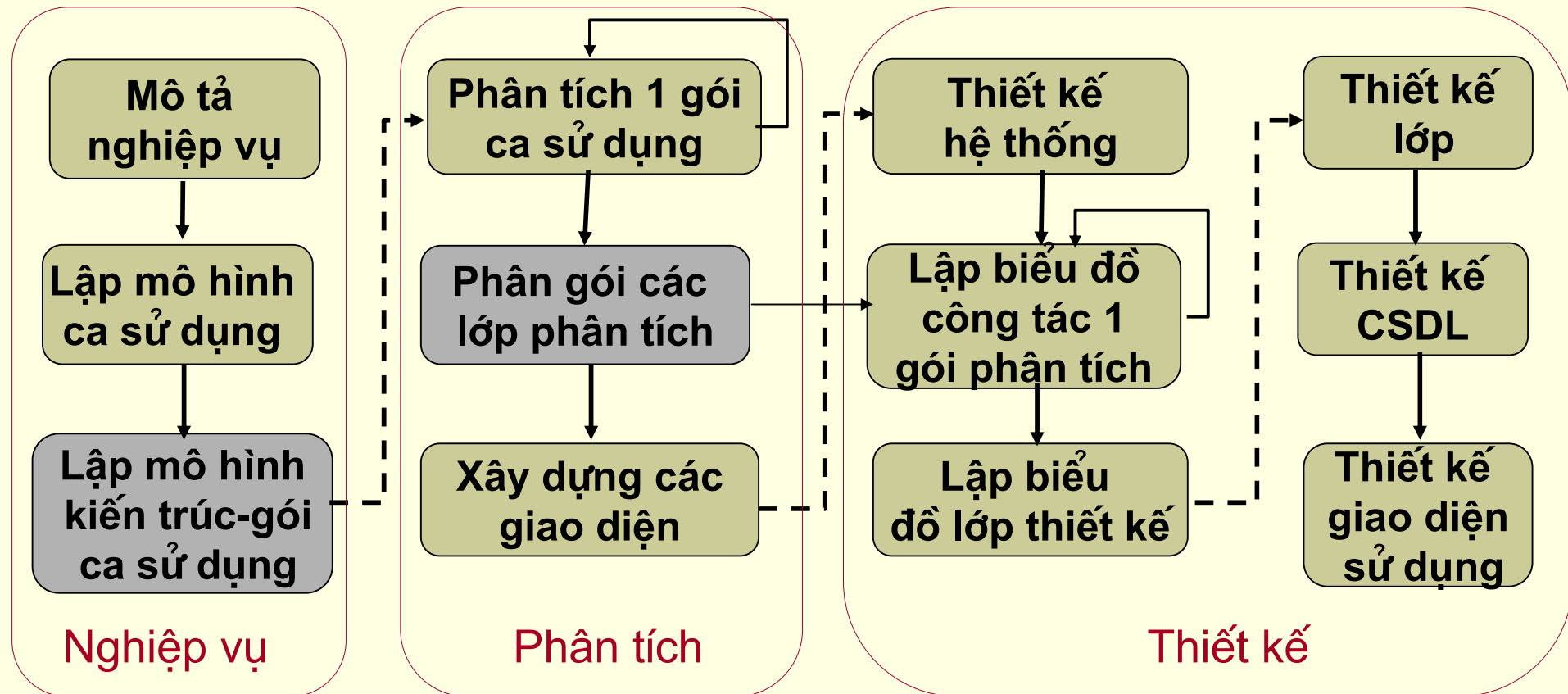
■ Mô hình cấu trúc gói

■ Mô hình cộng tác

■ Mô hình lớp

■ đặc tả lớp, giao diện

Tiến trình phân tích-thiết kế DT



Tiến trình phân tích và thiết kế hướng đối tượng

Phân tích hướng đối tượng



1. Mô tả nghiệp vụ

- Bằng lời
- Bằng biểu đồ hoạt động

2. Xây dựng mô hình nghiệp vụ

- Mô hình miền lĩnh vực
- Mô hình ca sử dụng

3. Phân tích xác định cấu trúc (khởi thảo)

- Làm mịn mô hình ca sử dụng
- Xác định các gói ca sử dụng, giao diện

Phân tích hướng đối tượng



1. Phân tích một ca sử dụng

- Tìm các lớp phân tích
- Xác định liên kết giữa các lớp

2. Phân gói lại các lớp phân tích (tăng cường kiến trúc)

- Tách các lớp dịch vụ & ứng dụng
- Phân gói các lớp phân tích theo tầng

3. Xác định và mô tả các giao diện

- Xác định giao diện giữa các gói
- Xác định liên kết giữa các gói

Thiết kế hướng đối tượng



1. Thiết kế biểu đồ tương tác mỗi gói

- Xác định lại các lớp
- Xây dựng biểu đồ tương tác

2. Phát triển biểu đồ lớp thiết kế

- Chuyển biểu đồ công tác sang biểu đồ lớp
- Hoàn thiện các quan hệ công tác

3. Thiết kế các lớp

- Thiết kế các thuộc tính
- Thiết kế các phương thức
- Thiết kế CSDL

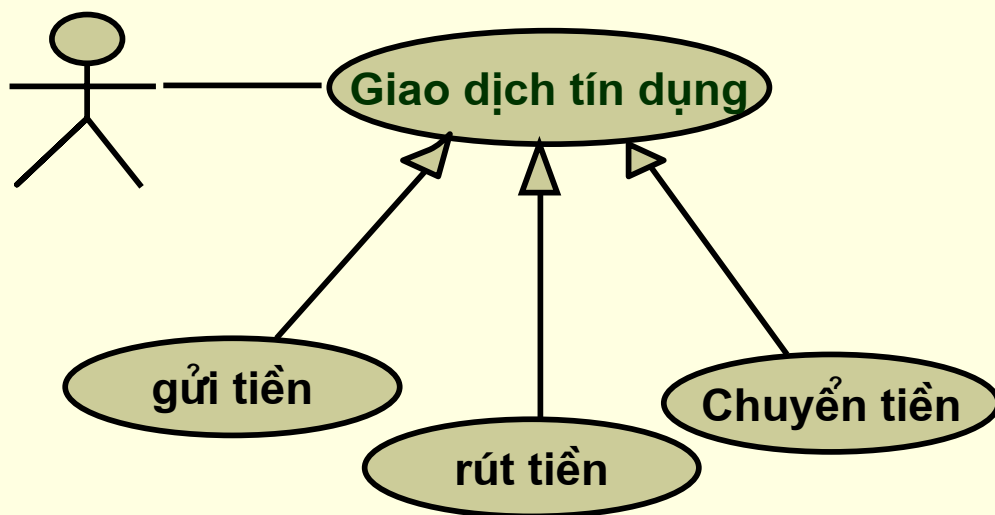
4. Thiết kế giao diện người dùng

Ví dụ: phân tích hướng đối tượng

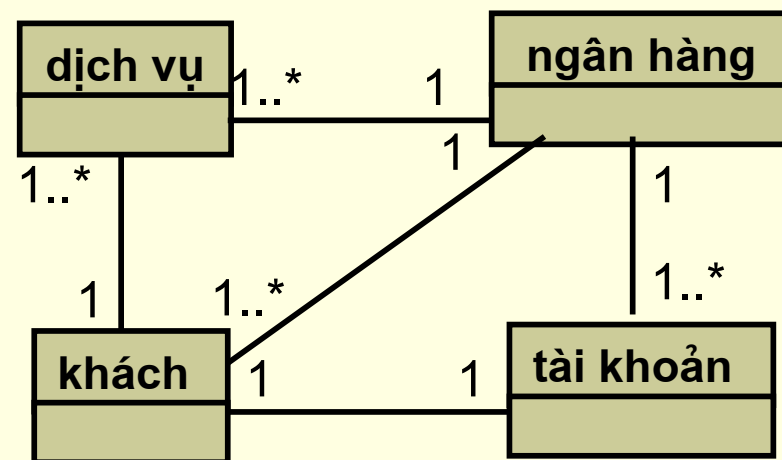


1. Bài toán: giao dịch tín dụng sử dụng máy ATM

2. Mô hình nghiệp vụ



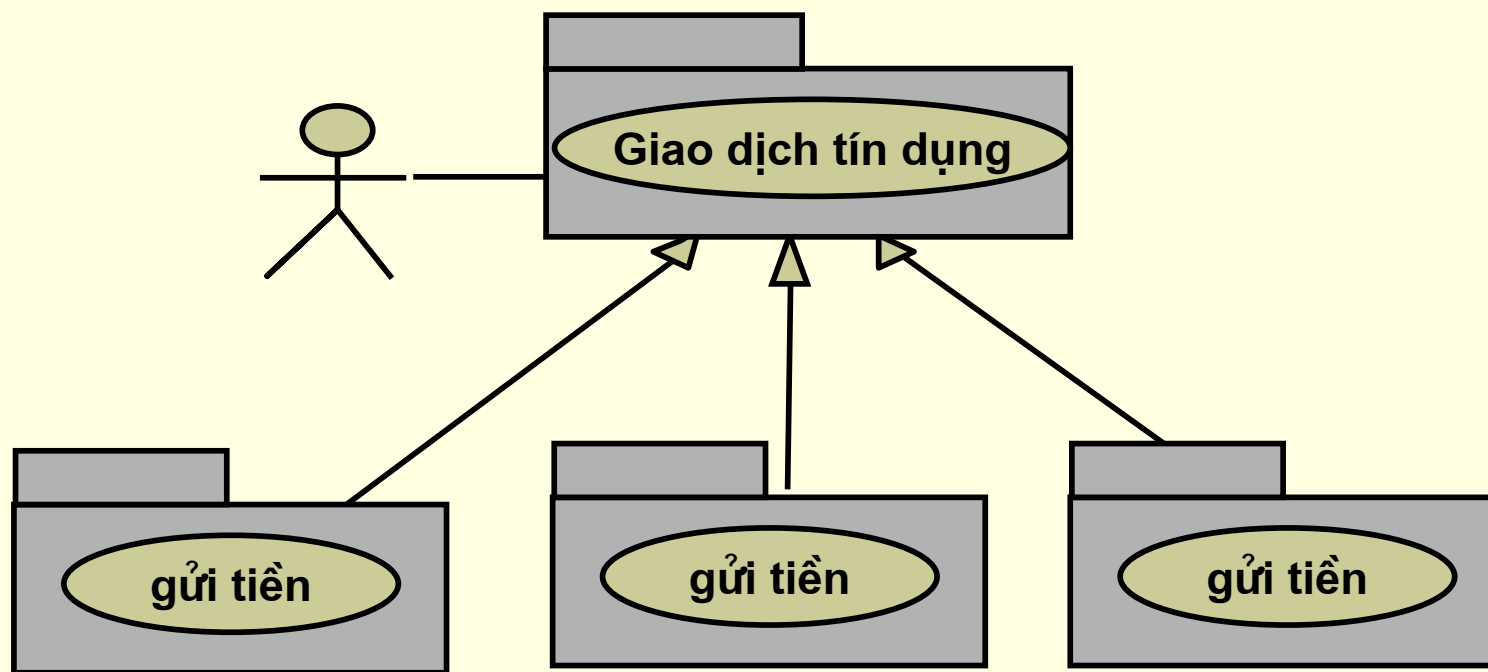
Biểu đồ ca sử dụng



Mô hình miền

Ví dụ: phân tích hướng đối tượng

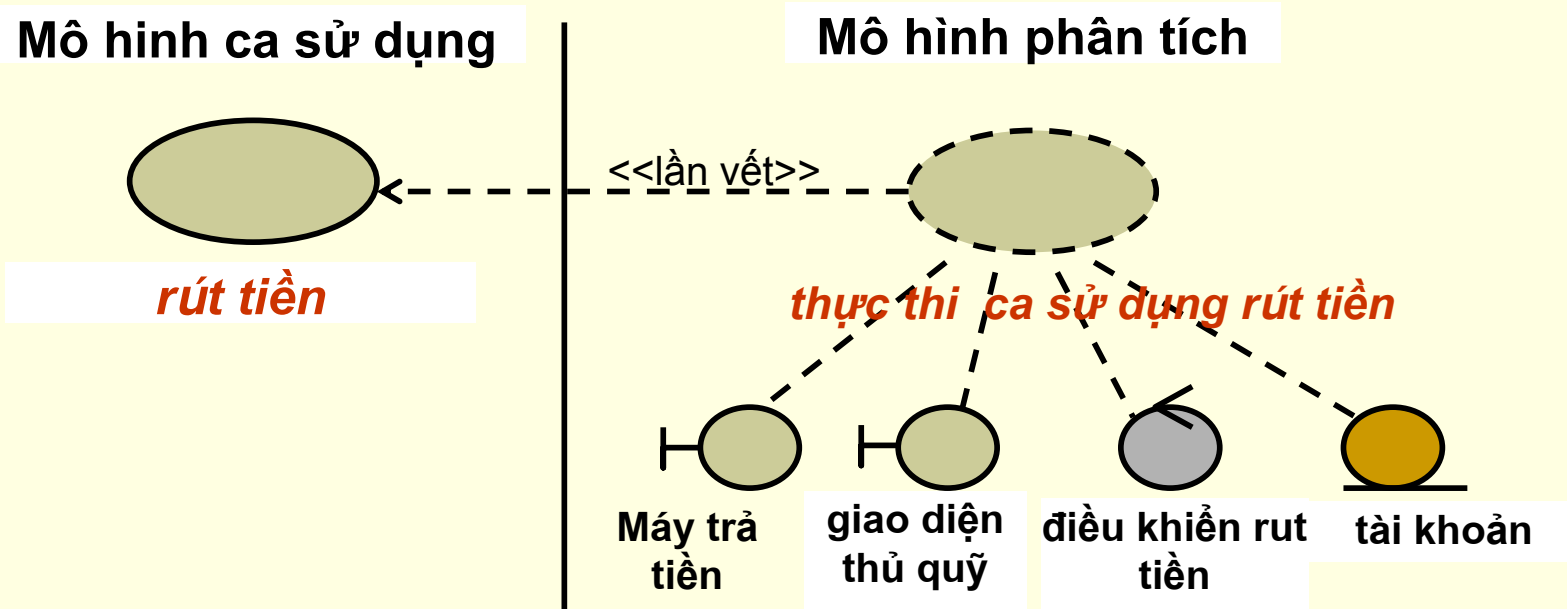
■ Xác định gói các ca sử dụng



Biểu đồ gói ca sử dụng

Ví dụ: phân tích hướng đối tượng

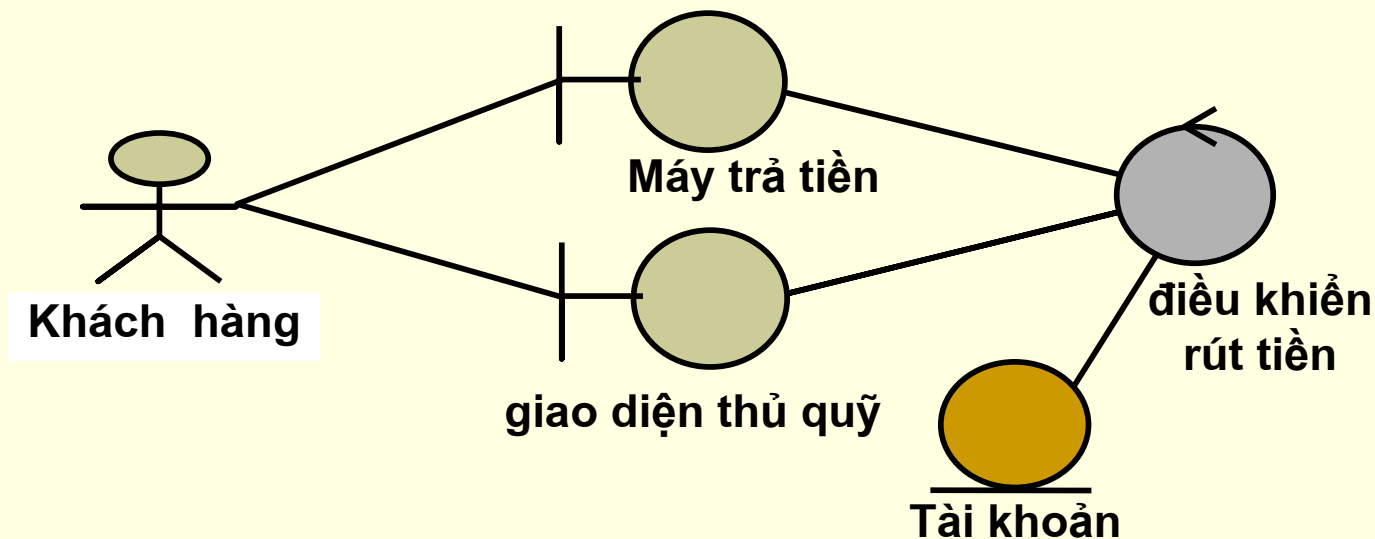
■ Phân tích một gói các ca sử dụng



Các lớp phân tích thực thi ca sử dụng *rút tiền*.

Ví dụ: phân tích hướng đối tượng

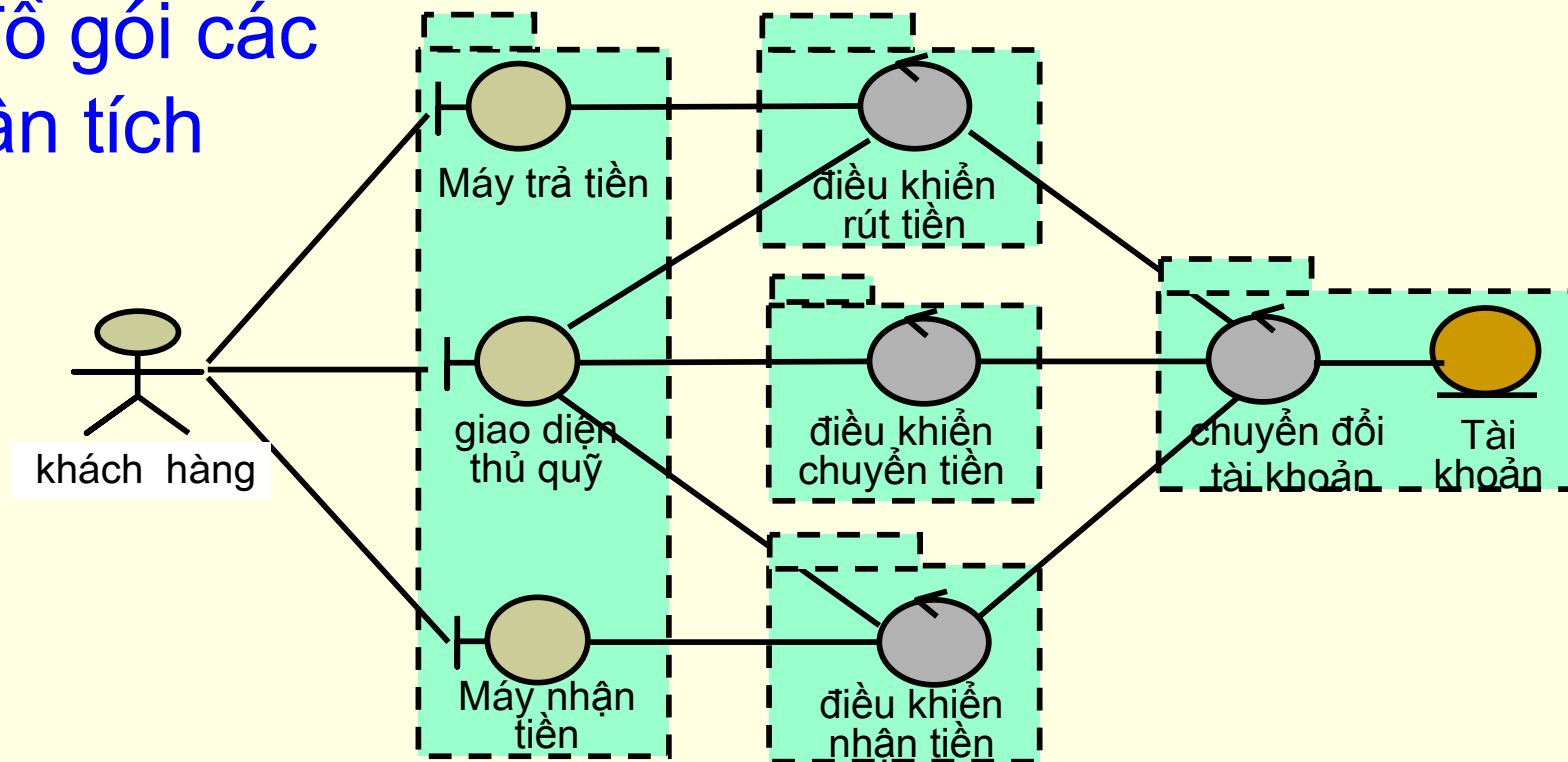
■ Biểu đồ phân tích một gói ca sử dụng



Các lớp phân tích và quan hệ giữa chúng

Ví dụ: phân tích hướng đối tượng

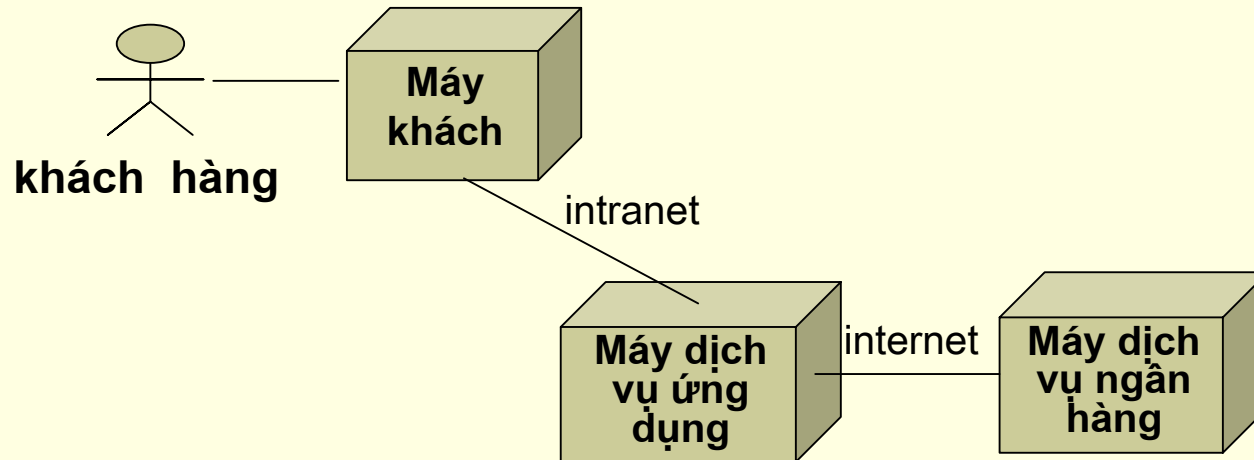
■ Biểu đồ gói các lớp phân tích



Các gói của các lớp phân tích

Ví dụ: Thiết kế hướng đối tượng

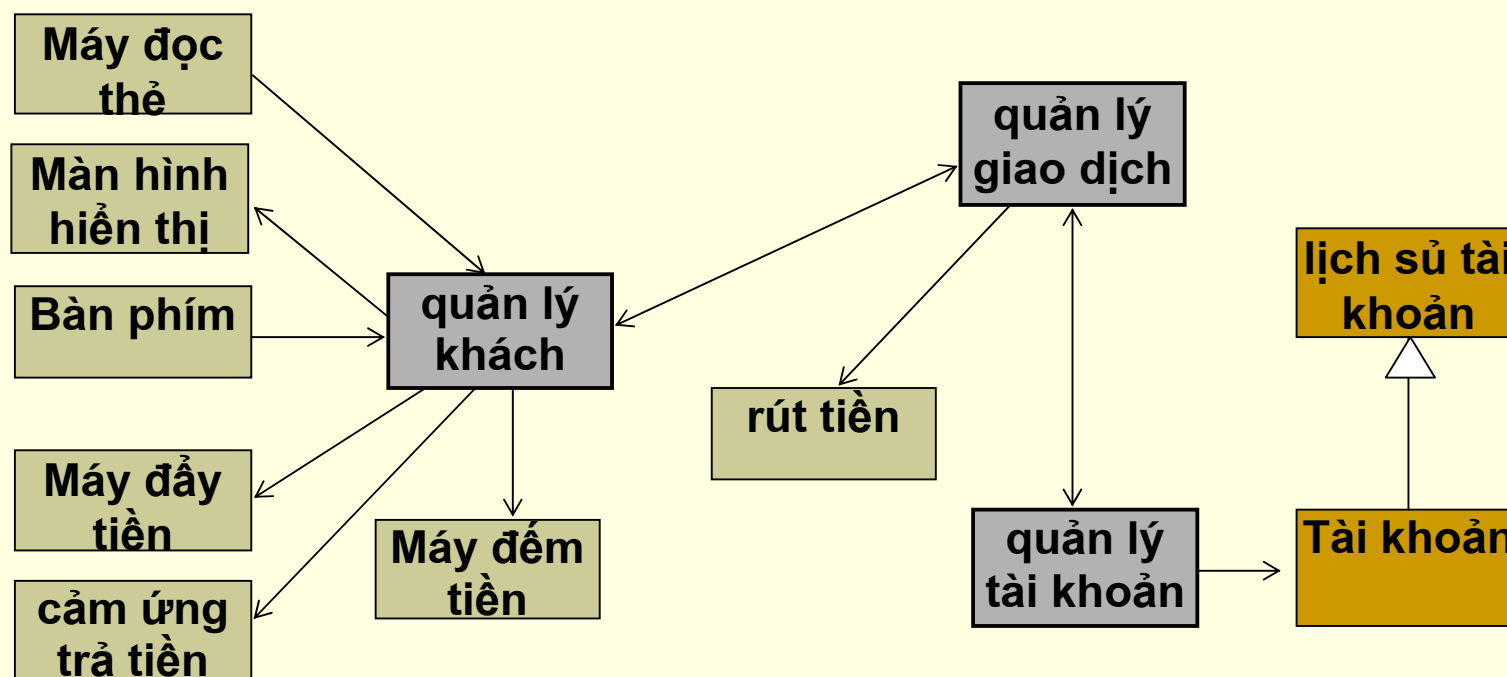
■ Thiết kế hệ thống



Biểu đồ bố trí các nút của hệ thống

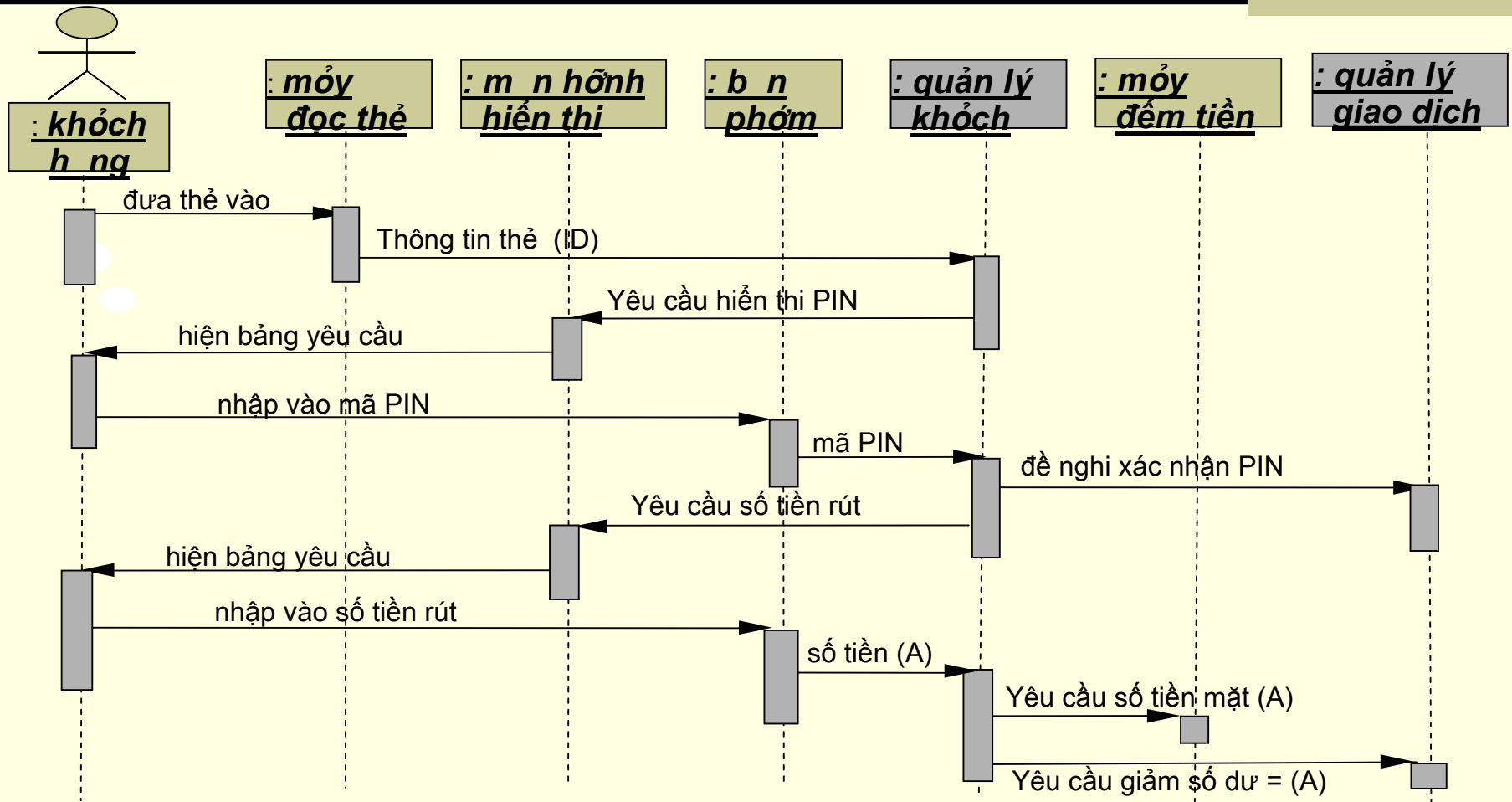
Ví dụ: Thiết kế hướng đối tượng

■ Biểu đồ các lớp thiết kế



Các lớp thiết kế tham gia thực hiện ca sử dụng rút tiền

Ví dụ: Thiết kế hướng đối tượng



Biểu đồ tuần tự thực hiện ca sử dụng rút tiền

Ví dụ: Thiết kế hướng đối tượng

■ Thiết kế lớp: tài khoản

a. Bảng các thuộc tính

Tên thuộc tính	kiểu	Nội dung
IDtaikh	<i>string</i>	Định danh tài khoản
sotkh	<i>string</i>	Số tài khoản dành cho một khách hàng gồm chữ, số, dấu
sodu	<i>money</i>	Số dư có trong tài khoản, đơn vị đo là tiền tệ

Ví dụ: Thiết kế hướng đối tượng

■ Thiết kế lớp: Tài khoản

b. Bảng các thao tác (tác vụ)

Tên thao tác	ý nghĩa
taolap()	Tạo một tài khoản cho khách hàng mới
gui()	Bổ sung tiền gửi vào tài khoản
chuyen()	Chuyển một số tiền từ 1 tài khoản sang 1 tài khoản khác
rut()	Rút một số tiền từ tài khoản
dong()	Đóng tài khoản

Ví dụ: Thiết kế hướng đối tượng

■ Thiết kế lớp

Taikhoan
IDtaikh: <i>string</i> sotaikh: <i>string</i> sodu: <i>money</i>
taolap (sotkh: <i>string</i> , sotien; <i>money</i>) + gui(sotkh: <i>string</i> , soien: <i>money</i>) + chuyen(sotkh: <i>string</i> , sotien: <i>money</i> , sotkh2: <i>string</i>) + rut (sotk: <i>h:string</i> , sotien; <i>money</i>) dong()

Lớp tài khoản với các thuộc tính và các thao tác

Mẫu thiết kế - Pattern



- Mẫu thiết kế: Pattern
 - khi thiết kế có nhiều trường hợp có sự tương tự
 - Mô tả giải pháp của một trường chung có thể áp cho trường hợp khác tương tự □ gọi là **mẫu thiết kế**
- Mô tả một mẫu bao gồm:
 - Vấn đề đặt ra (ngữ cảnh)
 - Giải pháp: phát biểu như một châm ngôn
 - Kết quả
 - Các mẫu liên quan
 - Mô hình mẫu

Ví dụ một số mẫu thiết kế



■ Bản chất mẫu

- ❖ Mẫu thiết kế không phải là cái gì mới mẻ
- ❖ Mẫu thiết kế là một sự đúc kết từ kinh nghiệm

■ 5 mẫu phần mềm gán trách nhiệm chung **GRASP** thường được sử dụng nhiều nhất là:

- Expert (chuyên gia)
- Creator (bộ tạo lập)
- Low Coupling (ghép nối thấp)
- High Cohension (kết dính cao)
- Controller (bộ điều khiển)

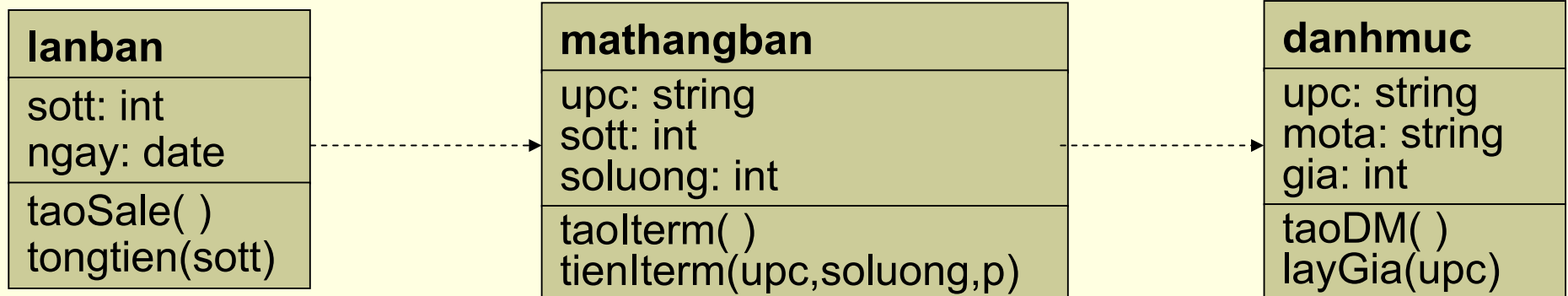
Ví dụ một số mẫu thiết kế



- Mô hình mẫu của mẫu chuyên gia
 - **Vấn đề:** Nguyên tắc gán trách nhiệm cho 1 đối tượng là gì?
 - **Giải pháp:** Hãy gán trách nhiệm cho đối tượng có đủ thông tin để thực hiện trách nhiệm đó
 - **Kết quả:** Giảm sự phụ thuộc vào lớp khác
 - **Mẫu liên quan:** kết dính cao, ghép nối lỏng

Ví dụ một số mẫu thiết kế

- Mô hình mẫu chuyên gia: bài toán bán hàng
 - Gán trách nhiệm cho **mathangban** tính tổng tiền bán 1 mặt hàng **tienlterm(upc, soluong)** vì nó có thông tin **soluong** và lấy giá từ **danhmuc**
 - Gán trách nhiệm cho **lanban** tính tổng tiền một lần bán **tongtien(sott)** vì nó biết số tiền từng mặt hàng thuộc lần bán



Lợi ích sử dụng mẫu thiết kế

- Cho ta giải pháp của vấn đề không cần tìm kiếm
- Dùng lại cái đã có, đỡ tốn thời gian và công sức
- Cho thiết kế tốt và chất lượng hệ thống cao

Câu hỏi và thảo luận

