

---

# **Chương 4:**

## **Các Sơ Đồ UML Khác**

---

# Nội dung chính

- Sơ đồ đối tượng (Object diagram)
- Sơ đồ tương tác đối tượng (Object-Interaction diagram)
- Sơ đồ trạng thái (State diagram)
- Sơ đồ hoạt động (Activity diagram)
- Sơ đồ thành phần (Component diagram)
- Sơ đồ triển khai (Deployment diagram)
- Sơ đồ gói (Package diagram)

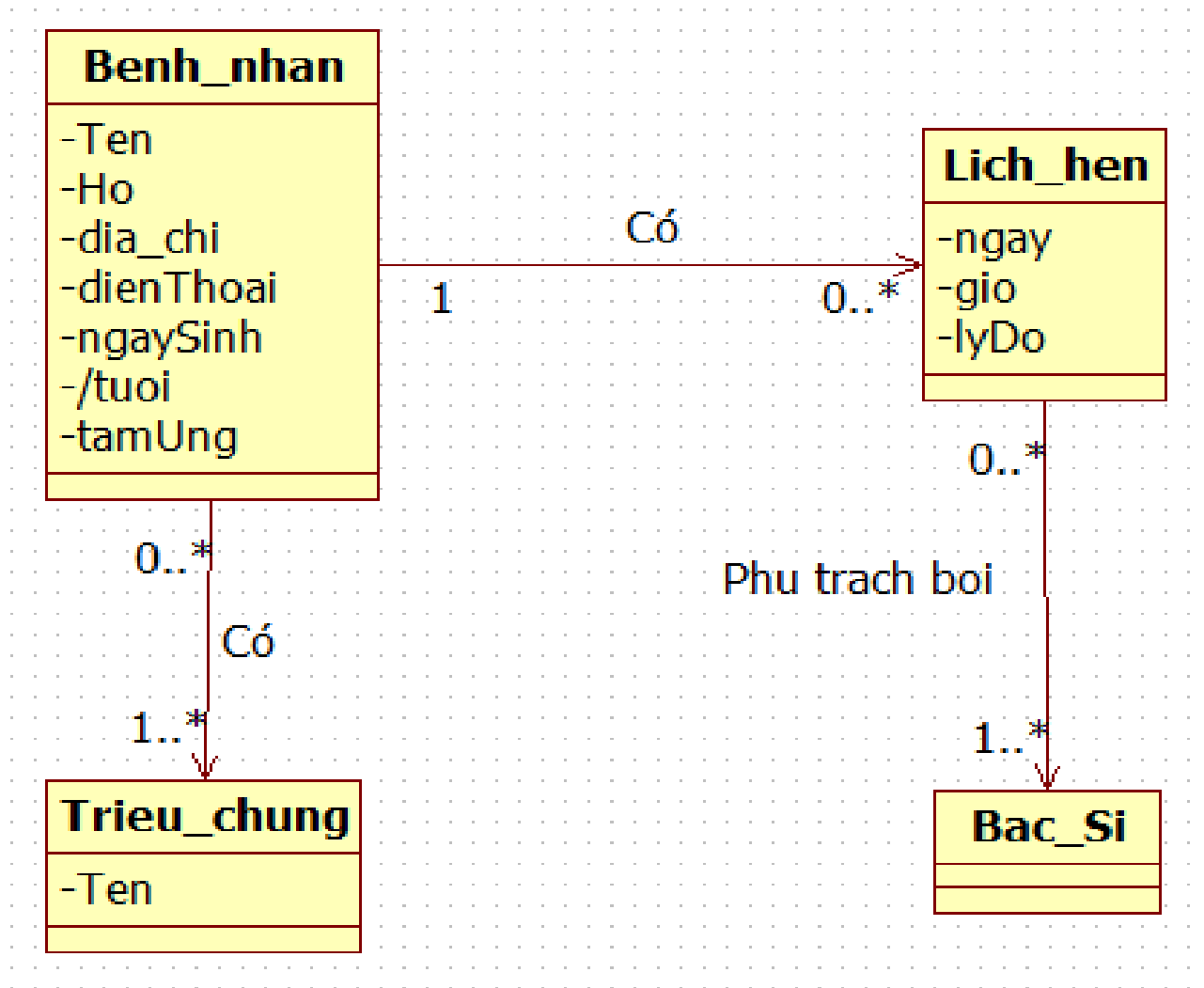
# Sơ đồ đối tượng (Object diagram)

---

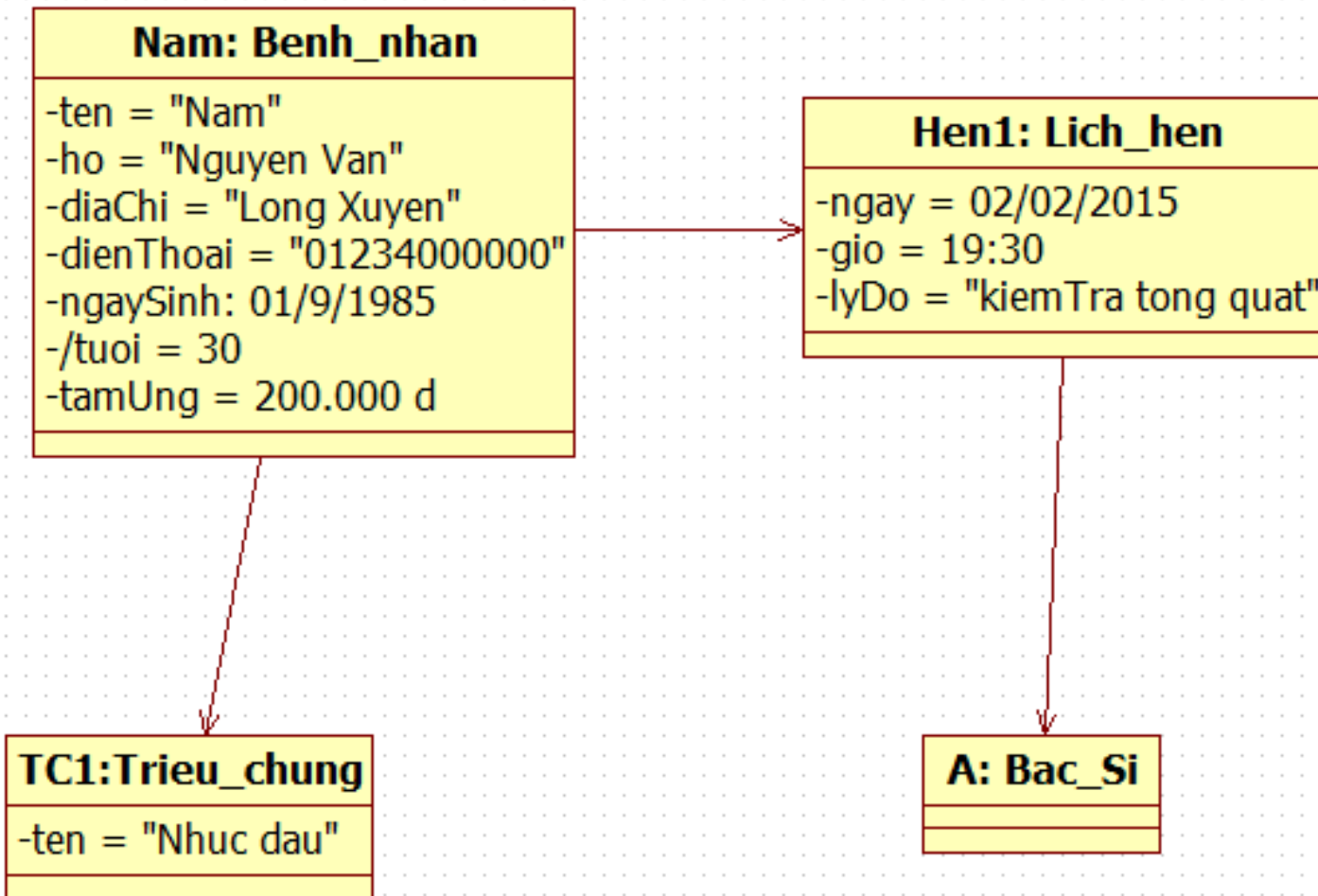
# Giới thiệu

- Là **một thể hiện cụ thể** của một phần hoặc toàn bộ sơ đồ lớp
- Thường được dùng khi muốn xem xét chi tiết của một lớp.
  - Thông qua đó có thể phát hiện ra những sai lầm khi thiết kế sơ đồ lớp.

# Từ sơ đồ lớp...



# ... đến sơ đồ đối tượng



# Sơ đồ tương tác đối tượng (Object-Interaction Diagrams)

---

Định nghĩa hành vi trong UML

# Giới thiệu

- Sơ đồ lớp biểu diễn quan hệ **tĩnh** giữa các lớp đối tượng
- Ngược lại, các sơ đồ tương tác biểu diễn mối quan hệ **động** giữa các đối tượng
- Thông thường, có rất nhiều mối quan hệ giữa các đối tượng:
  - Để đơn giản, quan hệ giữa các đối tượng thường được định nghĩa sử dụng các sơ đồ khác nhau cho mỗi *use case*



# Phân loại

- Có 2 loại sơ đồ tương tác đối tượng:
  - Sơ đồ tuần tự - sequence diagram
  - Sơ đồ cộng tác - collaboration diagram
- 2 loại sơ đồ này có thể sử dụng thay thế cho nhau, tuy nhiên mỗi loại sẽ có công dụng riêng của nó

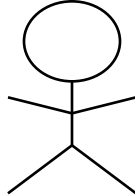

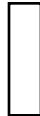
# Sơ đồ tuần tự vs Sơ đồ cộng tác

- Sơ đồ cộng tác thường được dùng để biểu diễn các tương tác đơn giản giữa các đối tượng
  - Sơ đồ cộng tác có thể dùng để định nghĩa các hành vi phức tạp nhưng sơ đồ có được thường rất phức tạp và khó theo dõi (difficult to follow)
  - Sơ đồ tuần tự đặc biệt thích hợp để định nghĩa các tương tác phức tạp dần theo thời gian

# Sơ đồ tuần tự

- Biểu diễn sự tương tác giữa các đối tượng tham gia vào một **tình huống xử lý** trong một usecase và các thông điệp được truyền giữa các đối tượng trong tình huống đó.
- Sơ đồ tuần tự được dùng trong các trường hợp sau:
  - Có rất nhiều tương tác trong một hành vi cụ thể
  - Trình tự của các tương tác khá phức tạp

# Các thành phần của sơ đồ tuần tự

Actor	
Object	<div><b>anObject:aClass</b></div>
Lineline	
Tiêu điểm (A focus of control)	
Message	<u>aMessage()</u> →
Hủy đối tượng	X

---

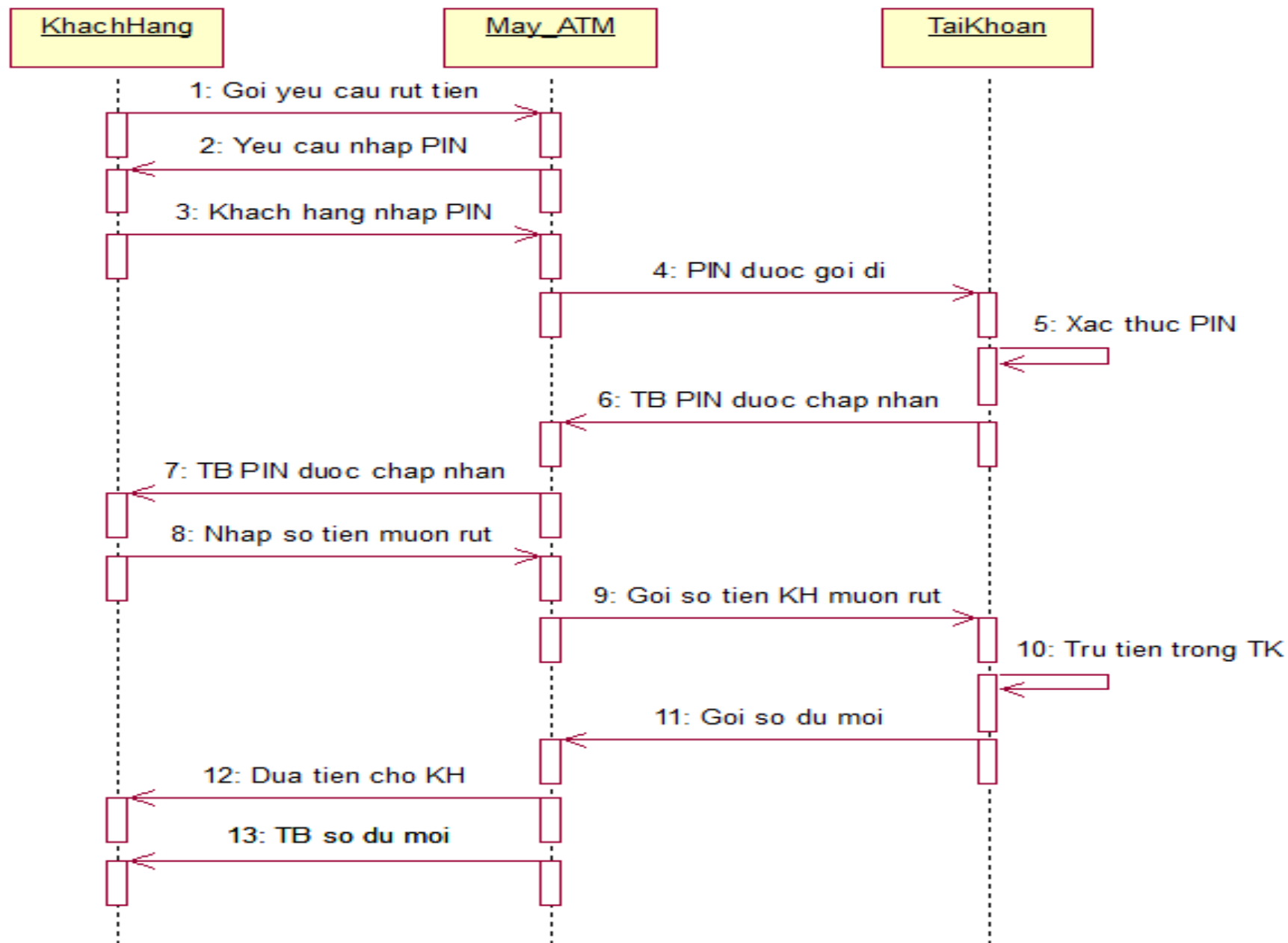
# Tạo một sơ đồ tuần tự

1. Xác định ngữ cảnh
2. Xác định các đối tượng tham gia
3. Thiết lập chu kỳ sống (lifecycle) cho mỗi đối tượng
4. Thêm các thông điệp vào
5. Đặt các tiêu điểm trên các lifeline
6. Xem xét lại sơ đồ

## Ví dụ: Cho một mô tả usecase gồm các luồng sự kiện chính sau:

- Khách hàng gửi yêu cầu rút tiền vào máy ATM.
- Máy ATM yêu cầu khách hàng nhập số PIN.
- PIN được gửi cho hệ thống để kiểm tra tài khoản
- Đối tượng tài khoản kiểm tra PIN và báo kết quả kiểm tra đến cho ATM
- ATM gửi kết quả kiểm tra này đến khách hàng
- Khách hàng nhập số tiền cần rút.
- ATM gửi số tiền cần rút đến cho tài khoản
- Đối tượng tài khoản trừ số tiền đó vào số dư trong tài khoản.
- Đối tượng tài khoản trả về số dư mới trong tài khoản cho máy ATM.
- Đối tượng ATM trả về mức tiền mới trong tài khoản cho khách hàng và dĩ nhiên, cả lượng tiền khách hàng đã yêu cầu được rút.

# Ví dụ

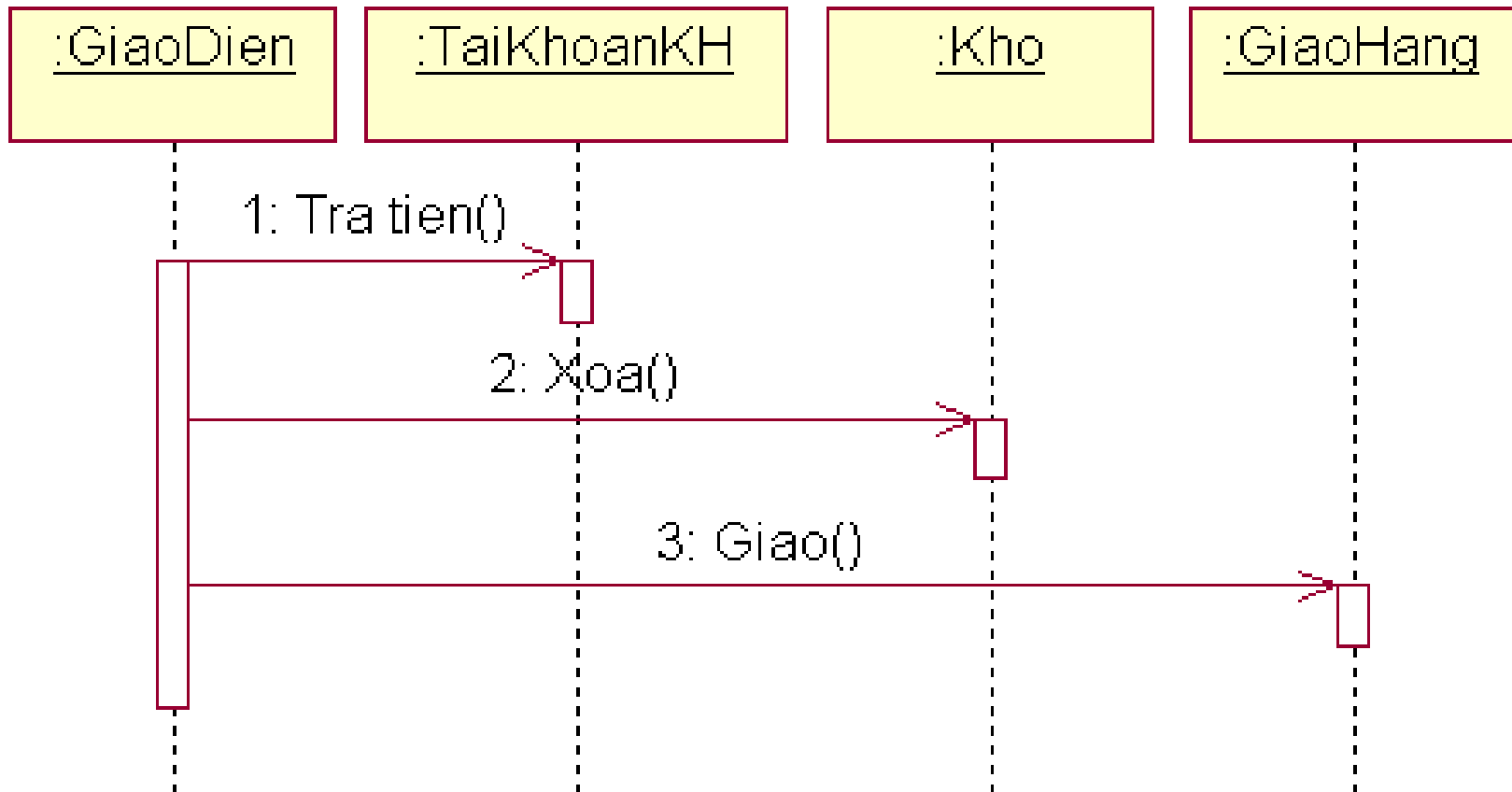


# Ví dụ

- Xét hành vi mua hàng gồm các bước sau:
  - ❑ Thu tiền từ tài khoản của khách hàng
  - ❑ Xóa sản phẩm tương ứng ra khỏi kho
  - ❑ Ra lệnh giao hàng cho khách hàng

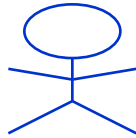



# Ví dụ



# Sơ đồ cộng tác

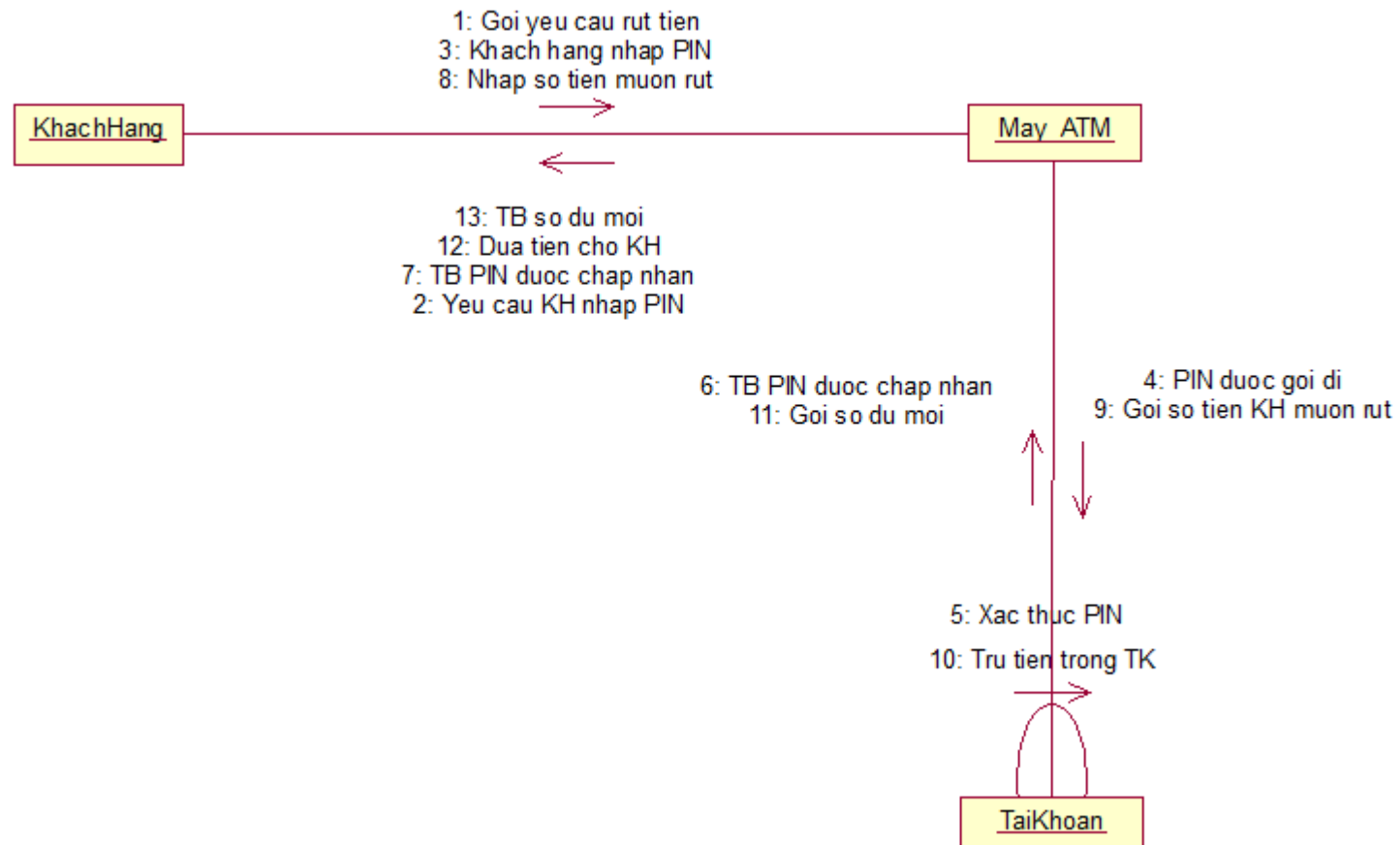
- Chỉ mô tả sự tương tác giữa các đối tượng mà không quan tâm đến trình tự thời gian xảy ra tương tác.
- Các thành phần:

Actor	
Object	<div>anObject:aClass</div>
Association	
Message	1: a message() →

# Các bước xây dựng sơ đồ cộng tác

1. Xác định ngữ cảnh
2. Xác định các đối tượng và các liên kết giữa các đối tượng
3. Vẽ các đối tượng và liên kết
4. Thêm các thông điệp
5. Kiểm tra lại sơ đồ

# Ví dụ: Xét luồng sự kiện trong UC ở slide 14



# Phân tích CRUD để xác định các tương tác giữa các đối tượng

- Xác định các quá trình tạo (**C**reate), đọc/tham chiếu (**R**ead/**R**eference), cập nhật (**U**psert) hoặc xóa (**D**estroy) các đối tượng dựa trên các UC.
- Mỗi tương tác giữa các đối tượng được biểu diễn bởi ma trận “CRUD”: một ma trận lớp-lớp trong đó mỗi ô biểu diễn cho sự tương tác giữa các thể hiện của các lớp.

# Ví dụ

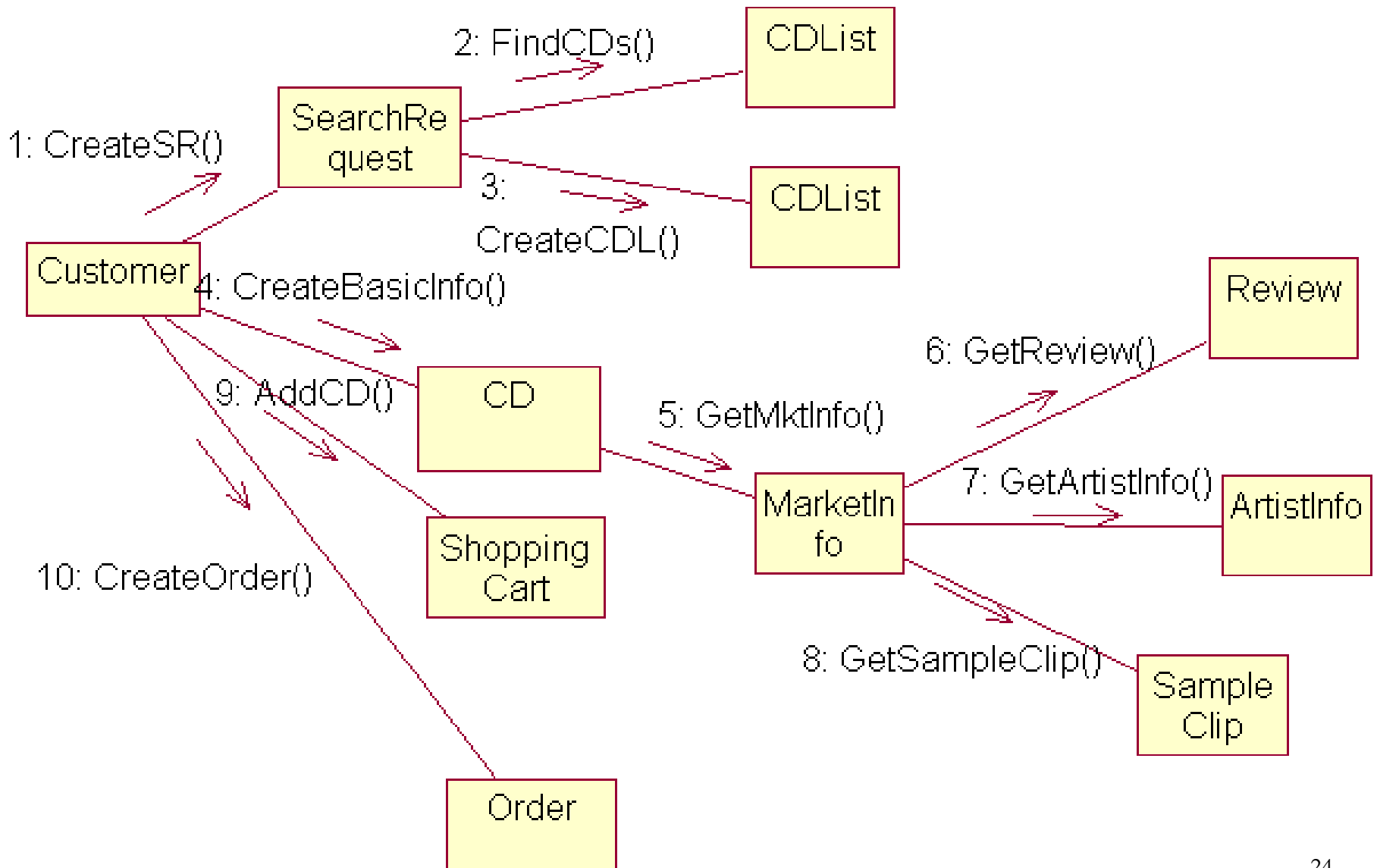
Cho một UC có các luồng công việc chính sau:

1. **Khách hàng** gửi một yêu cầu tìm kiếm cho hệ thống.
2. Hệ thống trả về cho **khách hàng** một danh sách các CD tìm được.
3. **Khách hàng** chọn một CD trong danh sách để tìm thêm thông tin.
4. Hệ thống cung cấp cho **khách hàng** các thông tin cơ bản và tóm tắt nội dung của CD.
5. **Khách hàng** gọi **UC đặt hàng**.
6. **Khách hàng** lặp lại các bước 3-5 để chọn thêm CD.
7. **Khách hàng** gọi **UC thanh toán**.
8. **Khách hàng** rời khỏi website.

# Ví dụ: Phân tích CRUD

	Khách hàng	Yêu cầu tìm kiếm	Danh sách CD	CD	Thông tin tiếp thị	Đánh giá	Tác giả	Clip mẫu	Giỏ hàng	Đơn đặt hàng
Khách hàng		C							U	C
Yêu cầu tìm kiếm			C							
Danh sách CD										
CD					R					
Thông tin tiếp thị						U	U	U		
Đánh giá										
Tác giả										
.....										

# Ví dụ:





# Sơ đồ tương tác đối tượng: Tóm tắt

- Sơ đồ tương tác đối tượng cho phép mô tả hành vi của các đối tượng trong hệ thống ở mức thấp.
- Để mô tả hành vi của toàn bộ hệ thống, có thể cần nhiều sơ đồ tương tác
- Loại sơ đồ tương tác nào sẽ được sử dụng phụ thuộc vào độ phức tạp của tương tác.

# Sơ đồ trạng thái (Statechart diagrams)

Biểu diễn trực quan sự biến đổi của đối tượng theo thời gian (Visualizing the Evolution of Objects Over Time)

# Giới thiệu

- Các đối tượng có cả hành vi và trạng thái. Với các đối tượng phức tạp, đặc biệt là các đối tượng mà hành vi phụ thuộc vào trạng thái của chúng, thì người phát triển hệ thống khó có thể hiểu được hết các trạng thái của chúng nếu chỉ dựa vào sơ đồ lớp và sơ đồ đối tượng.
- Lập trình viên cần phải biết cách thức đối tượng hoạt động bởi họ phải hiện thực những hành vi này trong phần mềm.

# Giới thiệu

- Sơ đồ trạng thái đảm bảo rằng lập trình viên không phải “đoán mò” đối tượng sẽ hoạt động như thế nào.
- Sơ đồ trạng thái được sử dụng để biểu diễn các trạng thái mà một đối tượng có thể có và sự di chuyển giữa các trạng thái đó.
- Với một bức tranh rõ ràng về hành vi của đối tượng, khả năng nhóm lập trình viên sẽ tạo ra một hệ thống thỏa mãn yêu cầu càng dễ trở thành hiện thực.

# Các thành phần của sơ đồ trạng thái

## ■ Trạng thái

- ❑ **Biểu diễn** tình trạng hiện có của đối tượng (có ý nghĩa trong một đoạn thời gian)
- ❑ Ví dụ: Đối tượng *bệnh nhân* có thể có những trạng thái “*mới*” hoặc “*hiện tại*” hoặc “*cũ*”
- ❑ Ký hiệu: Gồm 2 phần:
  - Tên
  - Các thao tác

Tên trạng thái
entry/ entry action do/ activity exit/ exit action

# Các thành phần của sơ đồ trạng thái

- Các sự kiện có thể xuất hiện trong một trạng thái
  - Đi vào (Entry) – Các hành động thực hiện khi đi vào trạng thái
  - Thực hiện (Do) – Các hành động thực hiện bên trong trạng thái
  - Đi ra (Exit) – Các hành động thực hiện khi đi ra khỏi trạng thái

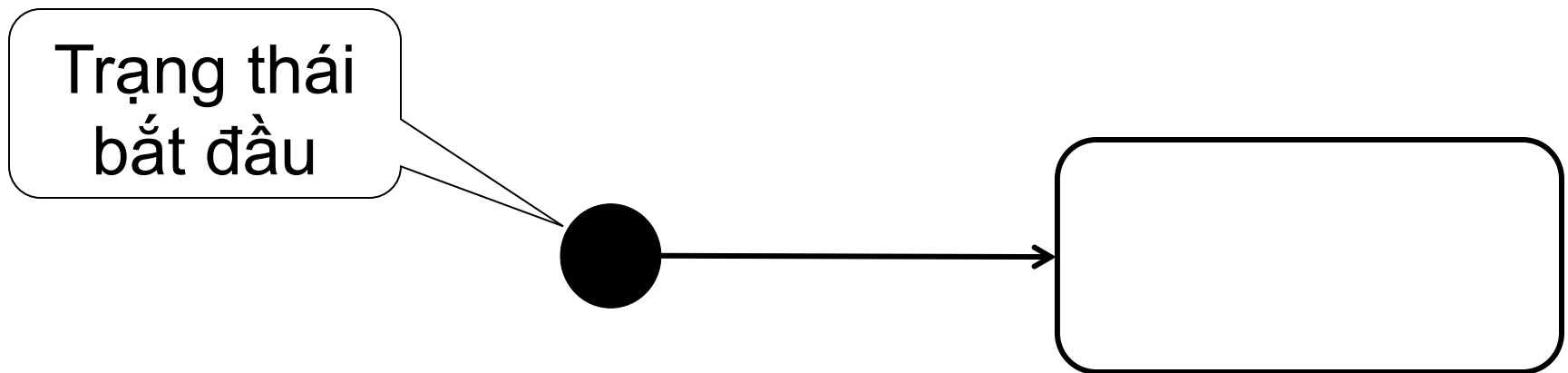
## Faxing

entry / Key in fax number  
do / Add datestamp and timestamp  
do / Add phone number  
do / Add owner  
do / Pull page through  
do / Paginate  
exit / Complete transmission

# Các thành phần của sơ đồ trạng thái

## ■ Trạng thái bắt đầu

- ❑ Là trạng thái khi mới được khởi tạo của đối tượng
- ❑ Bắt buộc phải có
- ❑ Chỉ có thể có 1 trạng thái bắt đầu

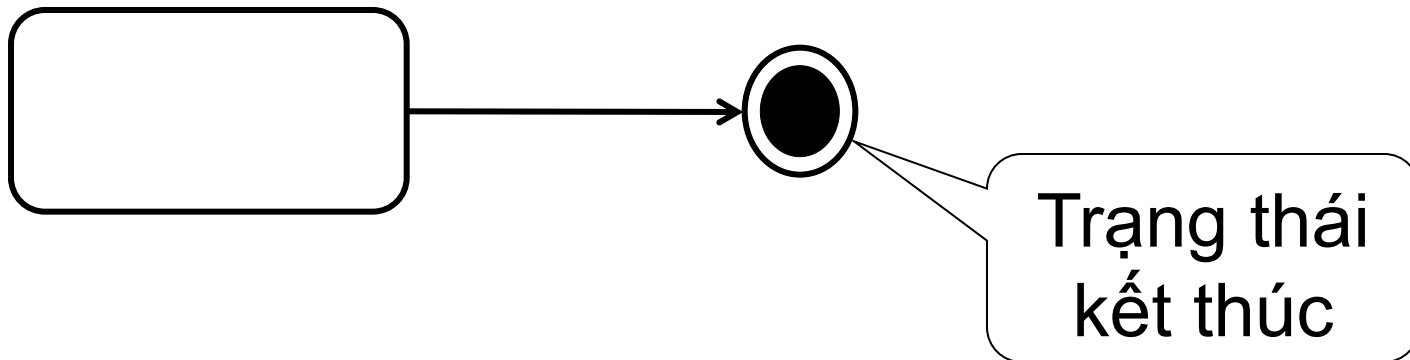




# Các thành phần của sơ đồ trạng thái

## ■ Trạng thái kết thúc

- ❑ Chỉ vị trí kết thúc đời sống của đối tượng
- ❑ Không nhất thiết phải thể hiện
- ❑ Có thể có nhiều trạng thái kết thúc



# Các thành phần của sơ đồ trạng thái

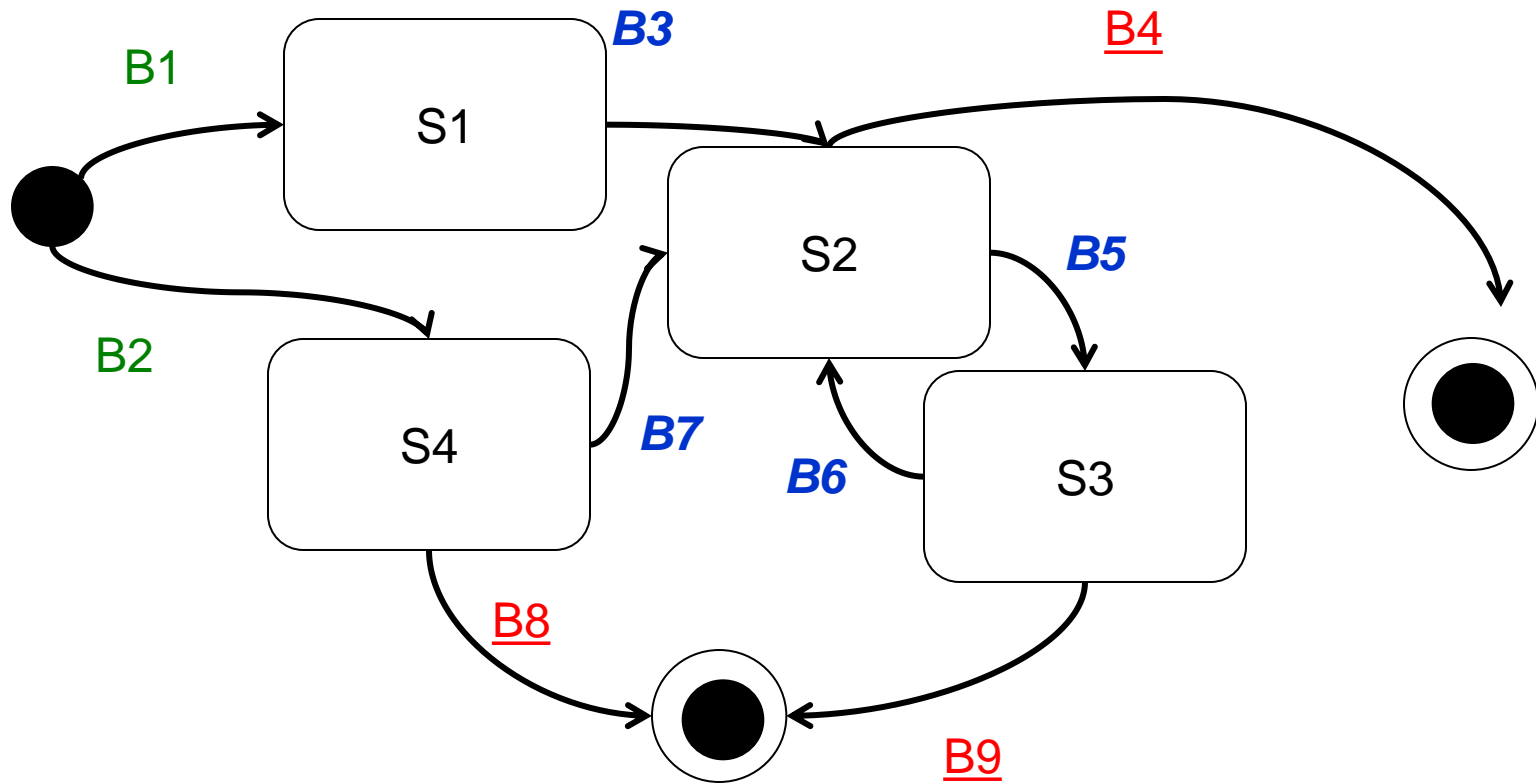
## ■ Biến cố

- ❑ Là các sự kiện xảy ra làm cho đối tượng chuyển sang trạng thái khác
- ❑ Ví dụ: Đối tượng *đơn hàng* có các sự kiện *tạo*, *gởi đơn hàng*, ....
- ❑ Được sử dụng để đặt nhãn một chuyển trạng thái.
- ❑ Các loại biến cố:
  - Sinh
  - Hoạt động
  - Tạm dừng
  - Mất

B1, B2: biến cố sinh

B4, B8, B9: biến cố mất

***B3, B5, B6, B7: biến cố hoạt động***

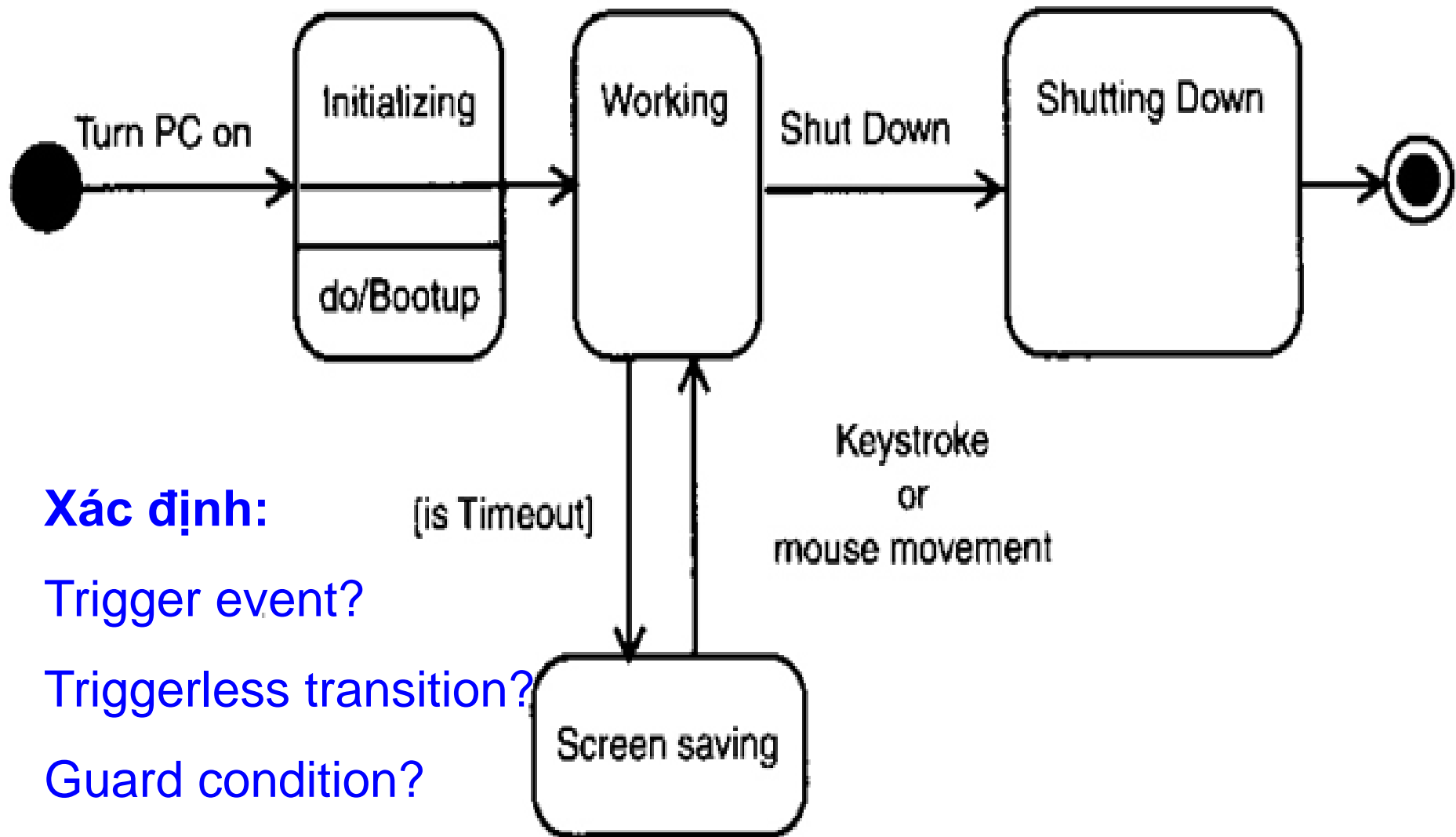


# Các thành phần của sơ đồ trạng thái

## ■ Chuyển trạng thái

- ❑ Cho biết một đối tượng chuyển từ trạng thái A sang trạng thái B
- ❑ Được mô tả bằng một mũi tên bắt đầu từ trạng thái A hướng đến trạng thái B
- ❑ Một chuyển trạng thái xảy ra do phản ứng lại:
  - Một sự kiện kích hoạt (trigger event).
  - Một hoạt động trong một trạng thái (triggerless transition).
  - Một điều kiện cụ thể (guard condition).

# Ví dụ



**Xác định:**

Trigger event?

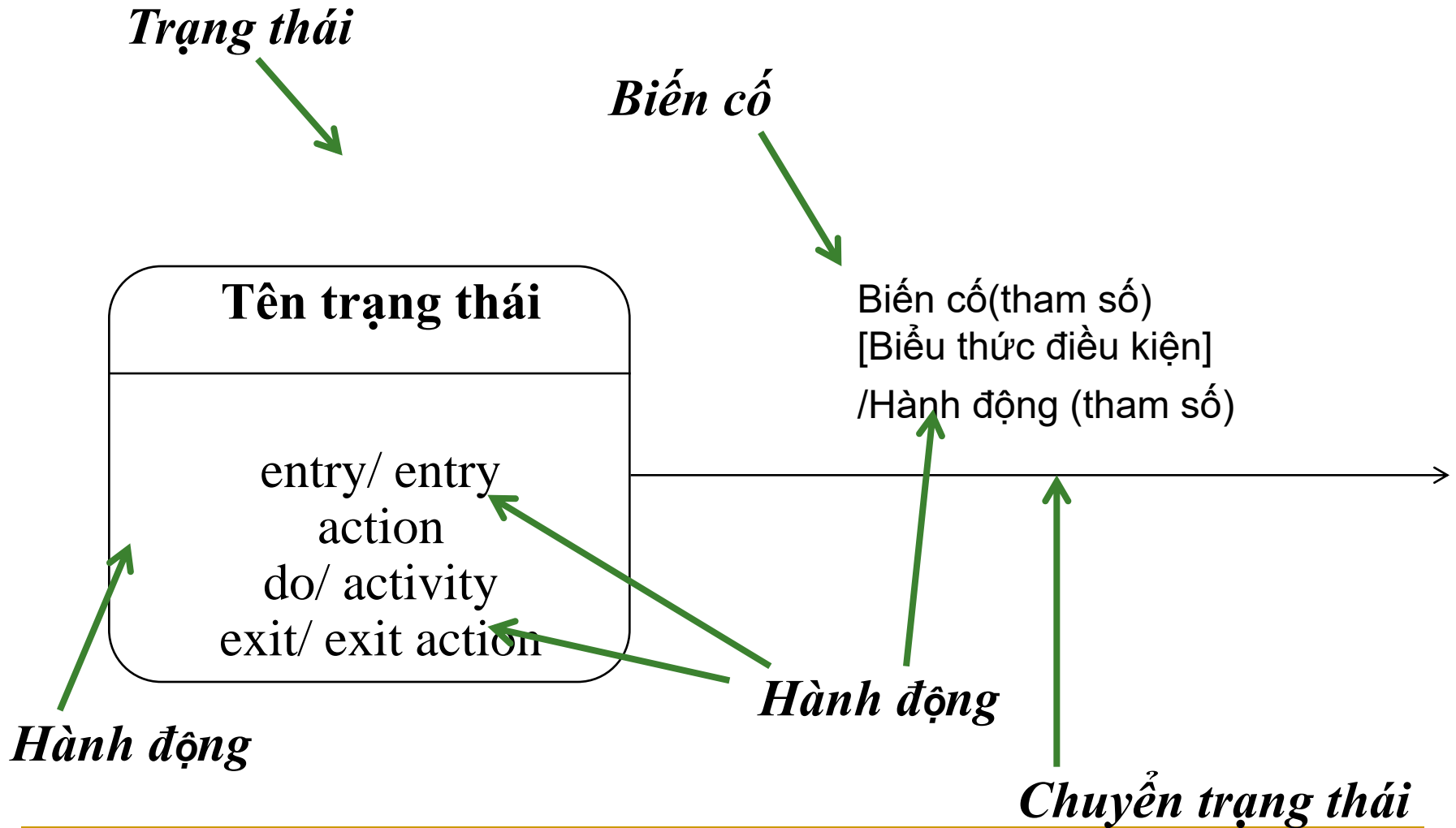
Triggerless transition?

Guard condition?

# Sơ đồ trạng thái: Định nghĩa

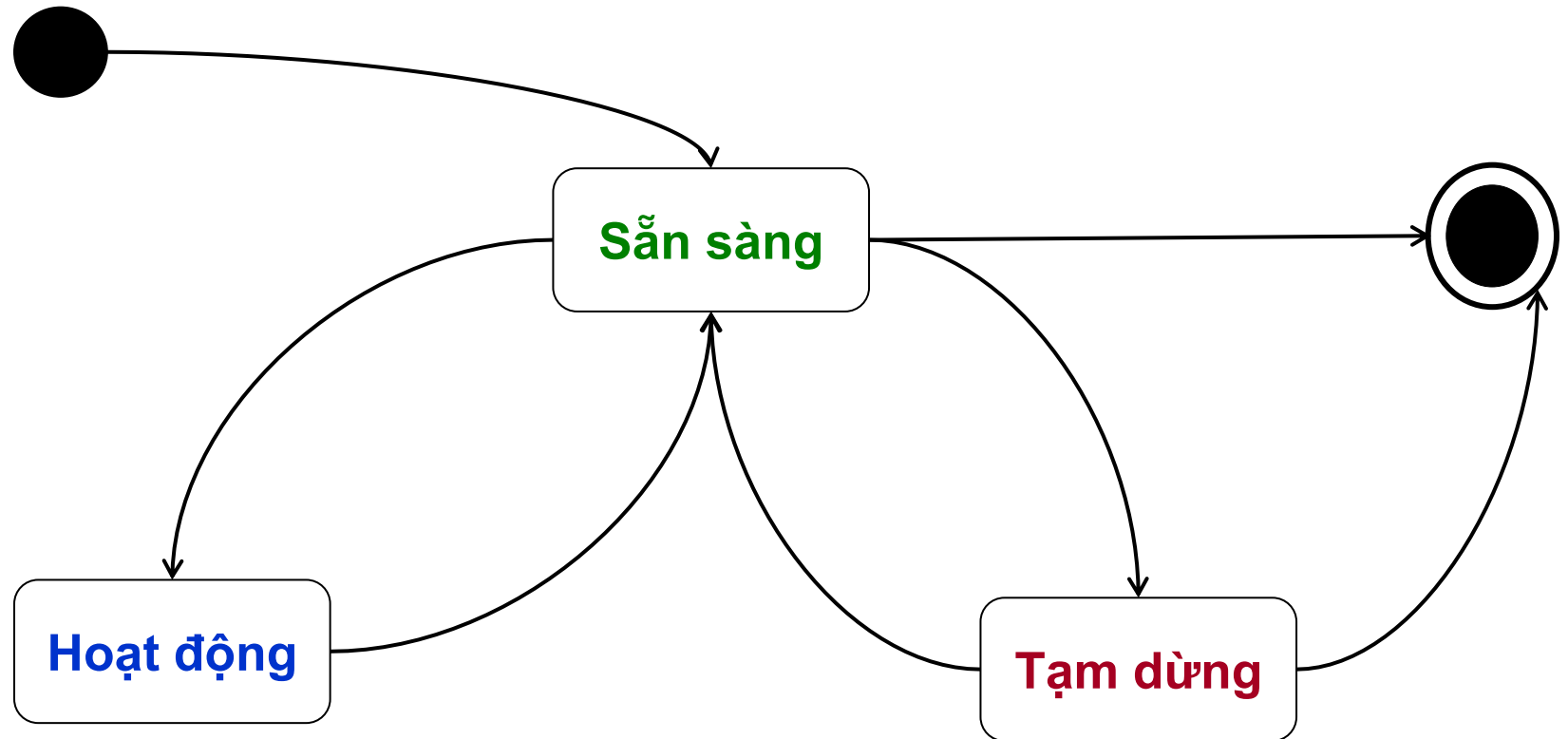
- Là đồ thị có hướng với các nút là các trạng thái nối với nhau bởi các cung mô tả việc chuyển đổi trạng thái.
- Mô tả chu trình sống của các đối tượng chính từ khi sinh ra, hoạt động & mất đi.
- Mỗi đối tượng có thể có nhiều sơ đồ trạng thái theo các góc nhìn khác nhau.

# Sơ đồ trạng thái: Ký hiệu



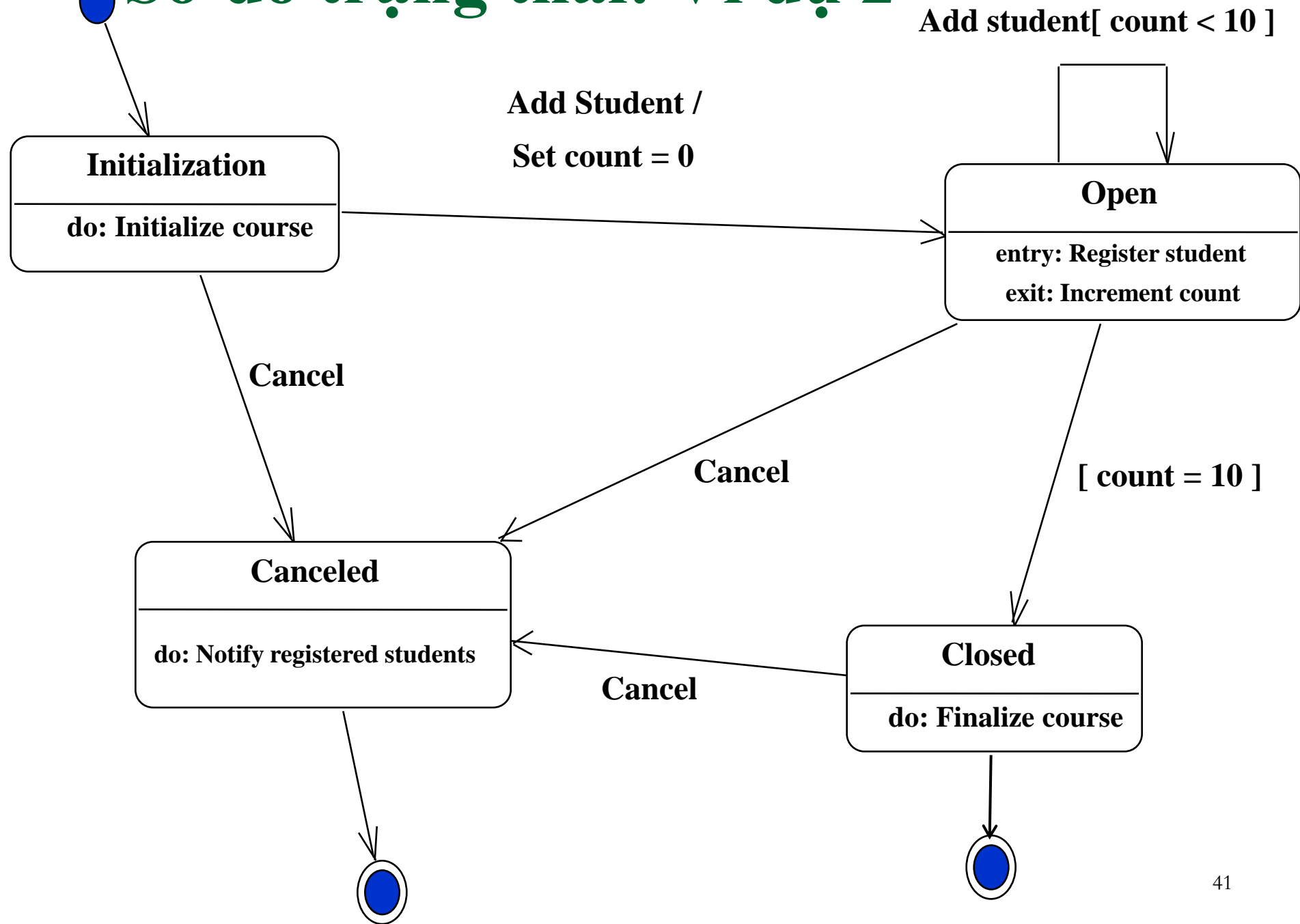
# Sơ đồ trạng thái: Ví dụ 1

- Sơ đồ trạng thái của 1 đối tượng có dạng:





# Sơ đồ trạng thái: Ví dụ 2



# Cách xây dựng sơ đồ trạng thái

- Được xây dựng từ sơ đồ lớp và các tài liệu đặc tả thu được từ quá trình thu tập và phân tích yêu cầu
- Chỉ cần vẽ sơ đồ trạng thái cho các đối tượng phức tạp và/hoặc có nhiều trạng thái
- Các bước thực hiện:
  - Xác định ngữ cảnh;
  - Xác định các trạng thái: bắt đầu, kết thúc và trung gian
  - Xác định quá trình hoạt động của đối tượng từ khi sinh ra đến khi mất đi cùng với các sự kiện/hành động/điều kiện làm cho đối tượng chuyển trạng thái
  - Kiểm tra lại sơ đồ

---

# Lưu ý

- Không phải tất cả các lớp đối tượng đều cần sơ đồ trạng thái
- Sơ đồ trạng thái thường sử dụng cho các đối tượng giao diện và điều khiển

# Bài tập

- Vẽ sơ đồ trạng thái cho đối tượng đèn giao thông ở một ngã tư.
- Vẽ sơ đồ trạng thái cho đối tượng **hóa đơn** (ở bài tập Quản lý cửa hàng sách), **chuyến du lịch** và **vé** (ở bài tập Quản lý tour du lịch).

# Sơ đồ hoạt động (Activity diagram)

Tham khảo:

<http://www.ibm.com/developerworks/rational/library/2802.html>

Thay thế cho các lưu đồ (flow chart) và sơ đồ dòng dữ liệu (Data flow diagram) trong phân tích theo hướng cấu trúc/chức năng

# Giới thiệu

- Sơ đồ hoạt động được dùng để:
  - ❑ Mô tả các hoạt động diễn ra đồng thời
  - ❑ Phân tích một UC
  - ❑ Tìm hiểu các luồng công việc xuyên qua nhiều UC
  - ❑ Mô hình hóa các ứng dụng đa luồng (song song)
- Nó chỉ ra:
  - ❑ Các bước trong luồng công việc
  - ❑ Các điểm quyết định
  - ❑ Ai có trách nhiệm thực hiện từng bước
  - ❑ Các đối tượng ảnh hưởng đến luồng công việc

# Các thành phần

- Hoạt động (activity)
  - Là phần tử đơn giản biểu diễn một hoạt động
  - Tên: Động từ + danh từ
  - Ký hiệu:

**Make appointment**

# Các thành phần




- Nút đối tượng (object node)
  - Biểu diễn một đối tượng được kết nối với các luồng đối tượng
  - Tên của nút là tên lớp của đối tượng
- Luồng điều khiển (control flow)
  - Biểu diễn chuỗi hoạt động
- Luồng đối tượng (object flow)
  - Biểu diễn luồng của 1 đối tượng từ hoạt động này sang hoạt động khác

**Class name**



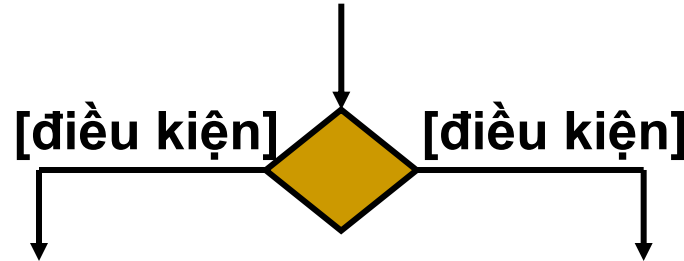


# Các thành phần

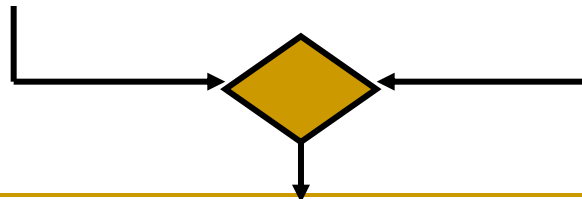
- Nút khởi đầu (initial node) 
  - Biểu diễn điểm bắt đầu của tập hợp các hoạt động
- Nút kết thúc hoạt động (final activity node) 
  - Kết thúc toàn bộ các luồng điều khiển và luồng đối tượng
- Nút kết thúc luồng (final flow node) 
  - Kết thúc một luồng điều khiển hoặc luồng đối tượng

# Các thành phần

- Nút lựa chọn (decision node)
  - Kiểm tra điều kiện để đảm bảo luồng điều khiển hoặc luồng đối tượng chỉ đi theo 1 đường



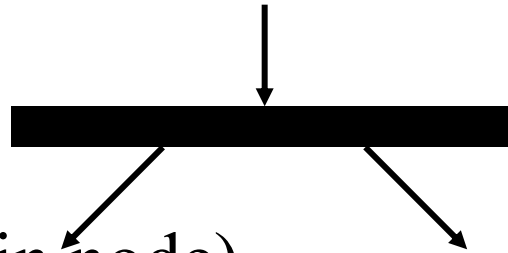
- Nút gộp (merge node)
  - Gộp các nhánh được tạo ra bởi nút lựa chọn



# Các thành phần

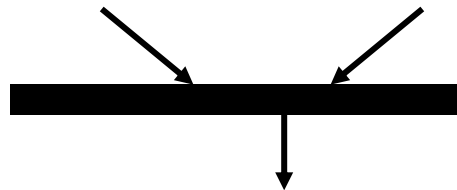
- Nút chia (fork node)

- Chia hoạt động thành các luồng hoạt động song song



- Nút kết hợp (join node)

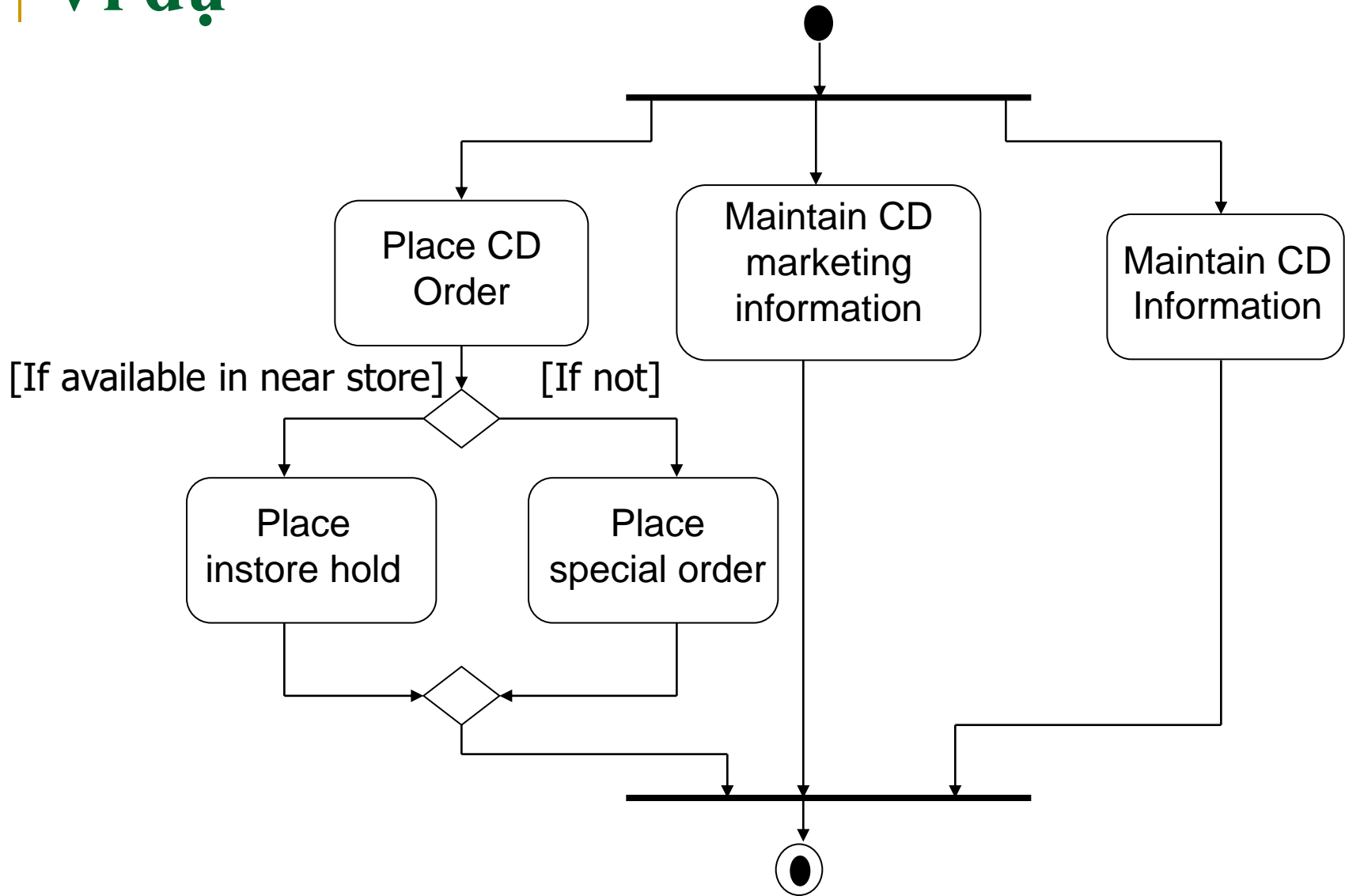
- Dùng để kết hợp các luồng hoạt động song song



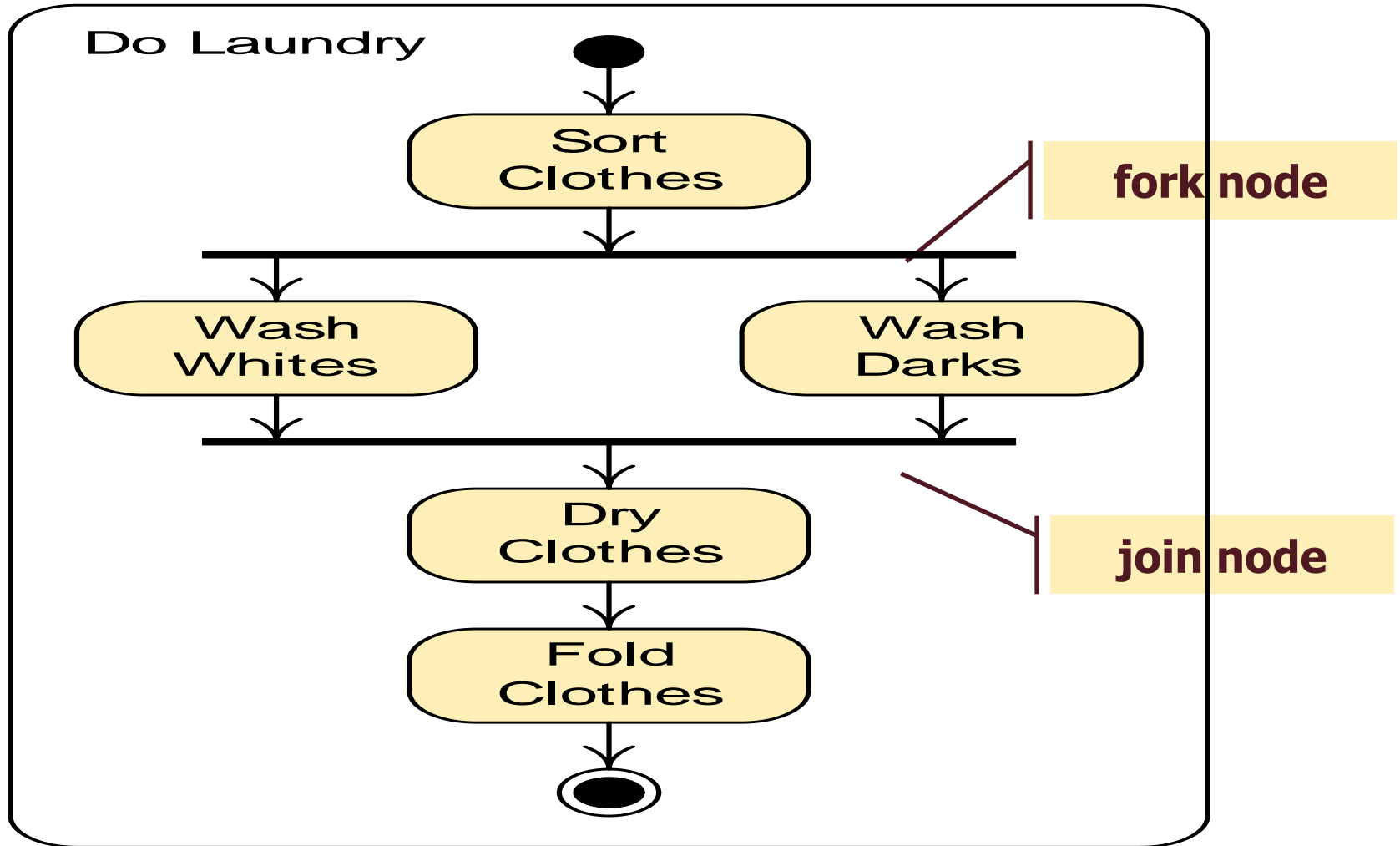
# Xây dựng sơ đồ hoạt động

1. Xác định phạm vi và ngữ cảnh của quá trình nghiệp vụ để đặt tên cho sơ đồ hoạt động.
2. Xác định các hoạt động, luồng điều khiển, luồng đối tượng diễn ra giữa các hoạt động.
3. Xác định các quyết định lựa chọn trong quá trình kinh doanh
4. Xác định khả năng thực hiện song song của các hoạt động (nếu có).
5. Vẽ sơ đồ hoạt động

# Ví dụ



# Ví dụ



# Bài tập

- Vẽ sơ đồ hoạt động cho các hoạt động:
  - Mua hàng trên một website bán hàng trực tuyến
  - Đăng ký môn học trên website đăng ký học phần trực tuyến

# So sánh sơ đồ hoạt động và sơ đồ trạng thái

- Sơ đồ hoạt động mô tả hành vi của một lớp đáp ứng lại các xử lý **bên trong** hệ thống.
- Sơ đồ trạng thái mô tả các trạng thái của một đối tượng đáp ứng lại các sự kiện **bên ngoài** hệ thống.



# Sơ đồ thành phần (Component diagram)

Biểu diễn các thành phần phần mềm của hệ thống và sự phụ thuộc giữa chúng.

# Giới thiệu

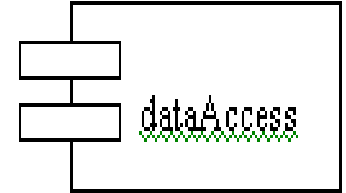
- Các sơ đồ thành phần cung cấp cho người thiết kế một định dạng tự nhiên để bắt đầu mô hình hóa một giải pháp.
- Sơ đồ thành phần cho phép người thiết kế xác định các chức năng cần có của một hệ thống đang được thực hiện bởi thành phần nào. Thông qua đó bảo đảm rằng hệ thống cuối cùng sẽ được người dùng chấp nhận.
- Sơ đồ thành phần là công cụ trao đổi thông tin hữu ích giữa các nhóm.

# Giới thiệu

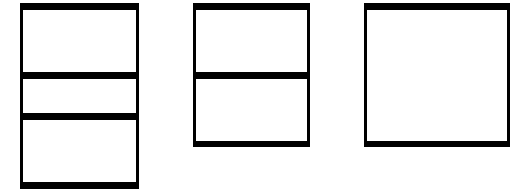
- **Đối với các nhà phát triển:** Sơ đồ thành phần cung cấp cho họ một khung nhìn về mặt kiến trúc (ở mức cao) của hệ thống mà họ sẽ xây dựng, giúp cho các nhà phát triển đưa ra lộ trình thực hiện và quyết định về phân bổ nhiệm vụ và cải tiến các kỹ năng cần thiết.
- **Đối với người quản trị hệ thống:** Sơ đồ thành phần giúp họ có được một khung nhìn sớm về những thành phần PM logic sẽ hoạt động trên hệ thống của mình.

# Cấu tạo

- Thành phần (Component)
  - Các đối tượng tương tác bên trong hệ thống



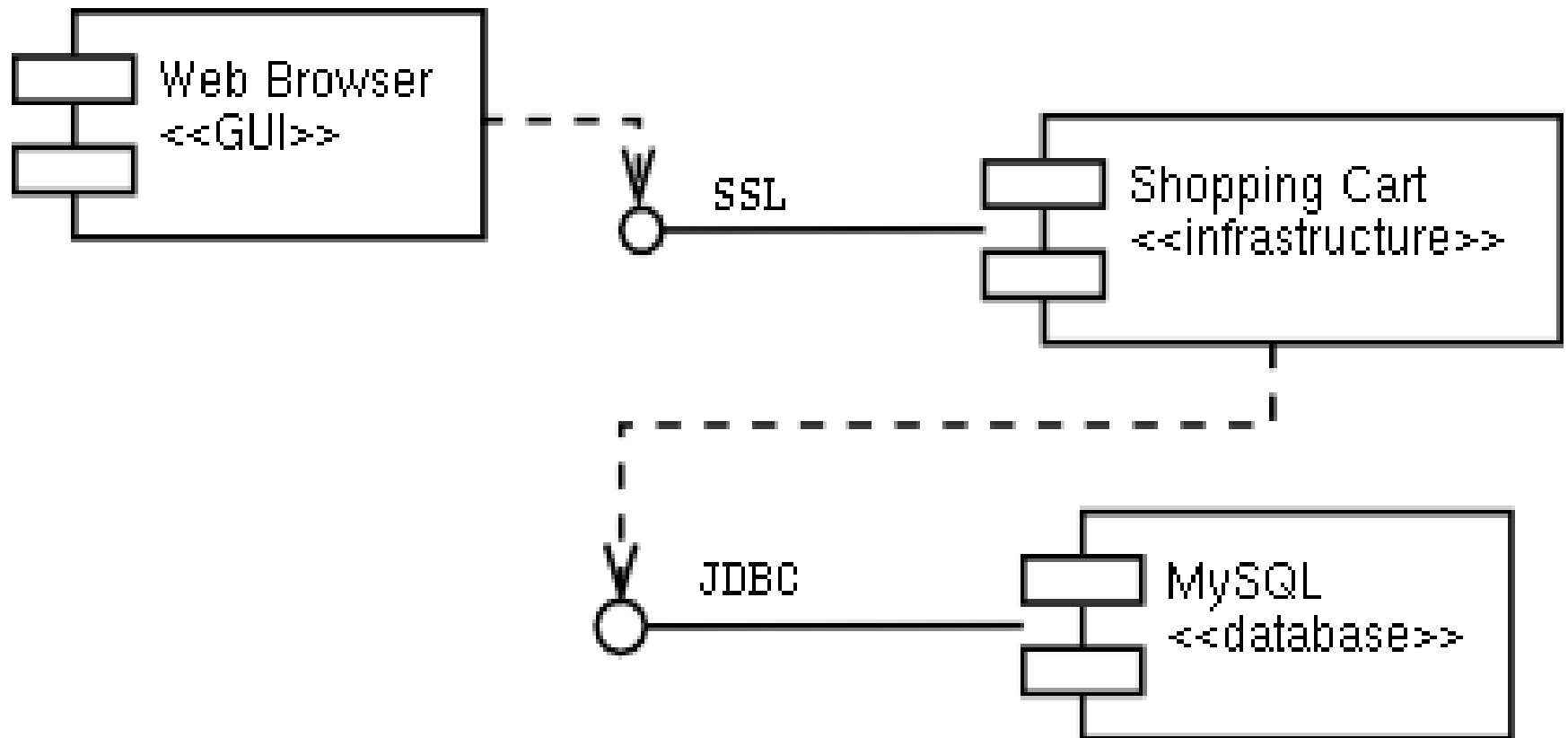
- Class/Interface/Object



- Quan hệ (Relation/Association)



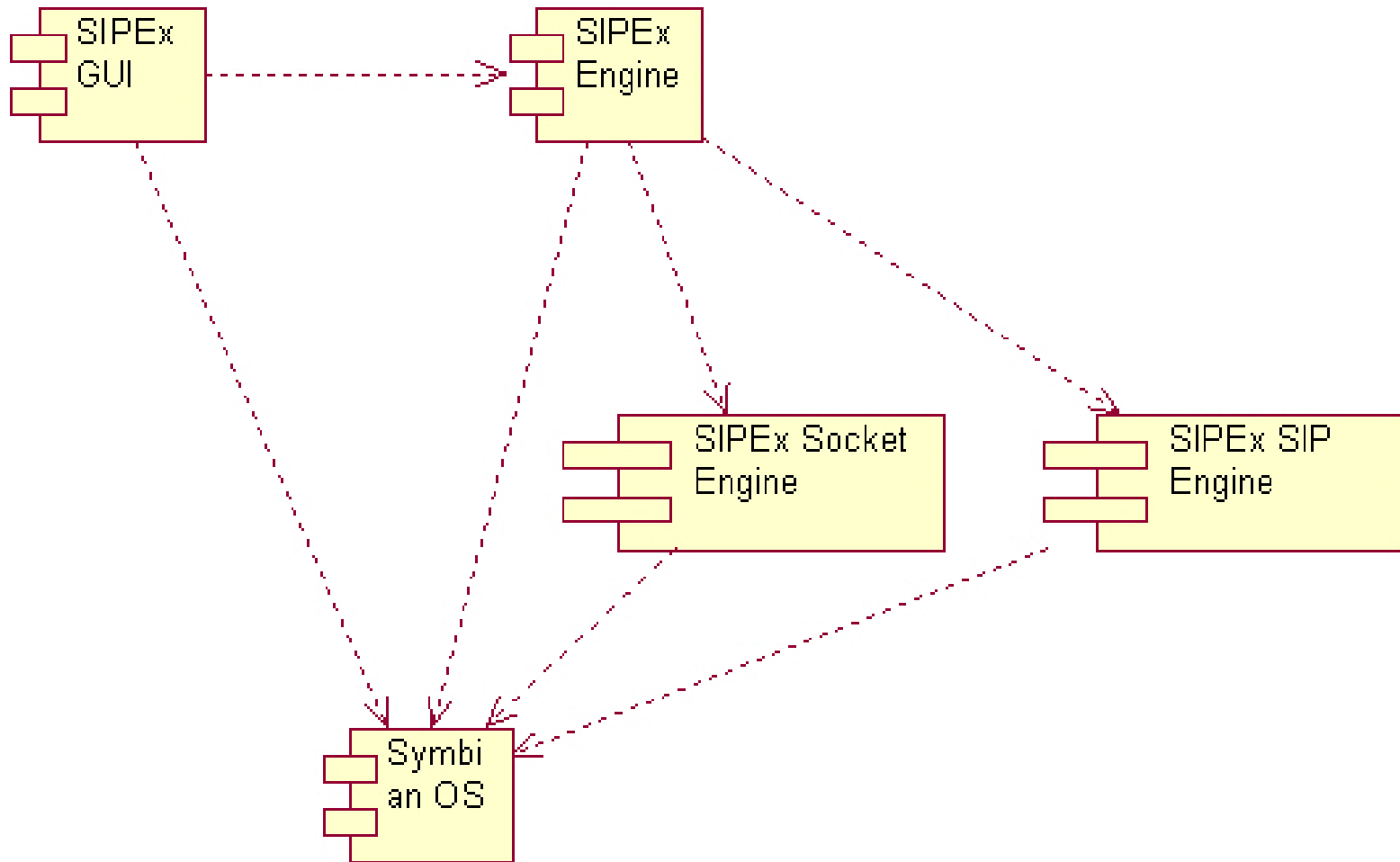
# Ví dụ



SSL = Secure Sockets Layer

JDBC = Java Database Connectivity

# Ví dụ



# Sơ đồ triển khai

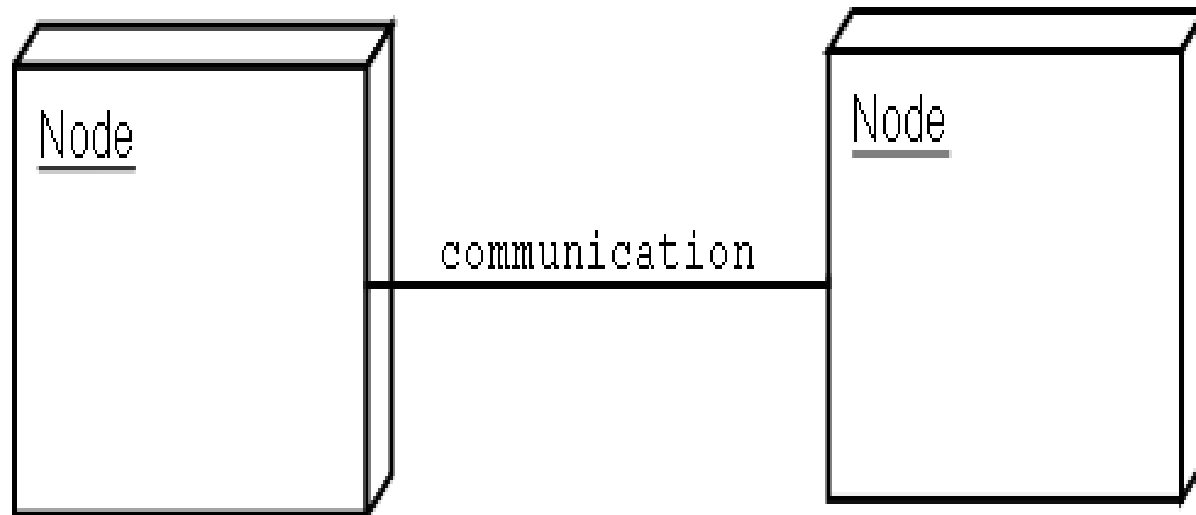
## (Deployment diagram)

# Giới thiệu

- Sơ đồ triển khai dùng để vẽ khung nhìn tĩnh của các nút xử lý trong hệ thống và các thành phần vận hành trên các nút này.
- Nói cách khác, sơ đồ triển khai dùng để biểu diễn các phần cứng của hệ thống, các phần mềm được cài đặt trên đó và các thành phần trung gian (middleware) dùng để kết nối các hệ thống lại với nhau.
- Cần phải tạo ra sơ đồ này cho các ứng dụng được triển khai trên nhiều thiết bị phần cứng khác nhau.



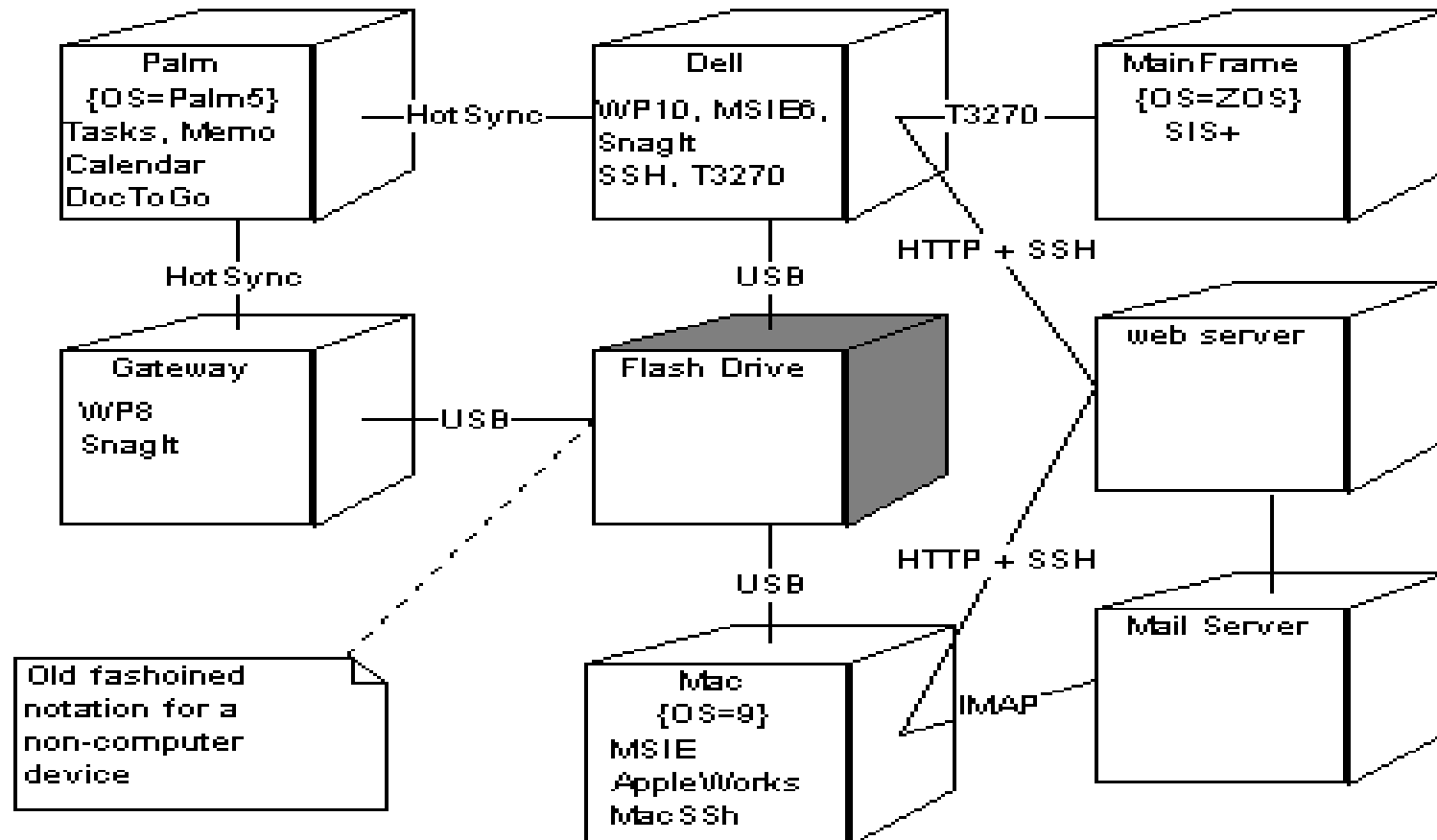
# Cấu tạo



- Các nút biểu diễn các thành phần xử lý (computational elements) như processor, server....)

# Ví dụ

UML Deployment Diagram analysing current hardware



Rational Rose Student Edn.

# Sơ đồ gói (Package diagram)

---

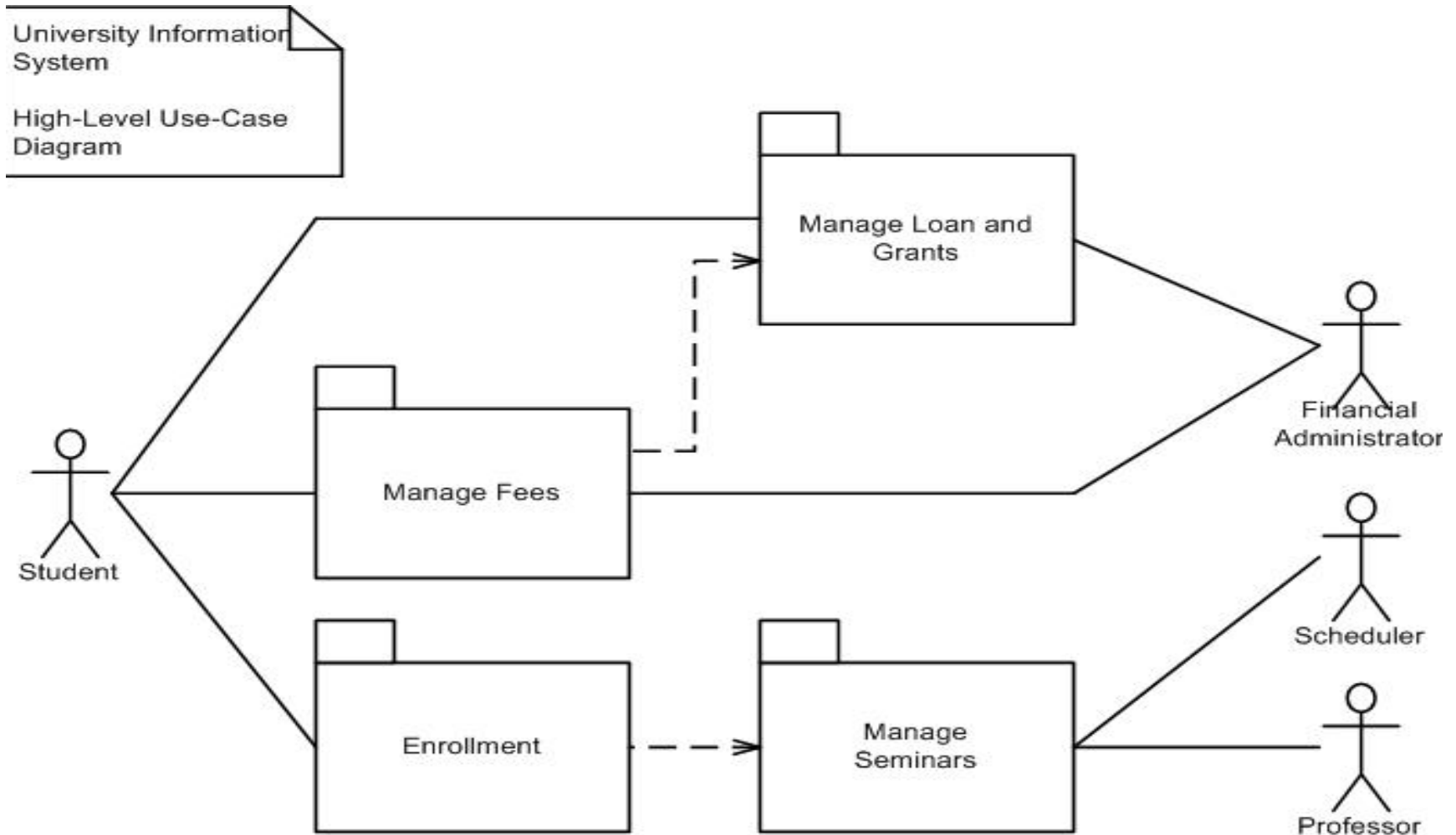
# Giới thiệu

- Dùng để sắp xếp các thành phần của các sơ đồ UML thành các nhóm, giúp cho sơ đồ đơn giản và dễ hiểu hơn.
- Các gói cũng giống như các folder và có thể dùng cho bất cứ sơ đồ UML nào. Tuy nhiên chỉ sử dụng phổ biến trong sơ đồ usecase và sơ đồ lớp vì các sơ đồ này thường phức tạp.
- Một sơ đồ có thể được mô tả ở nhiều mức khác nhau

# Tạo gói trong sơ đồ UC

- Nguyên tắc
  - Các UC mở rộng (extending) và bao hàm (included) thuộc vào cùng một gói với các UC có liên quan.
  - Các UC có cùng mục tiêu và do một actor thực hiện sẽ thuộc cùng một gói.

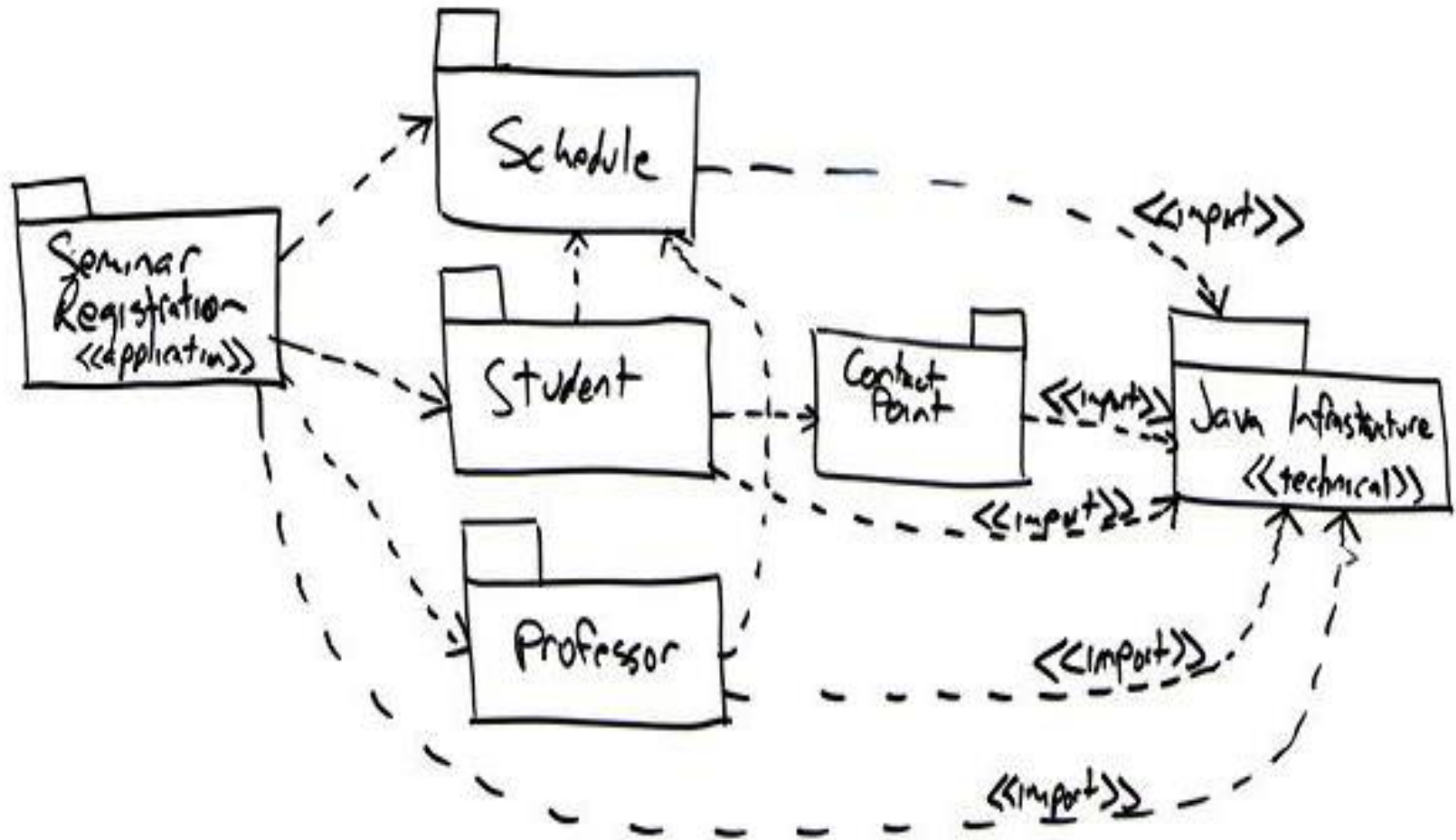
# Tạo gói trong sơ đồ UC: Ví dụ



# Tạo gói trong sơ đồ lớp

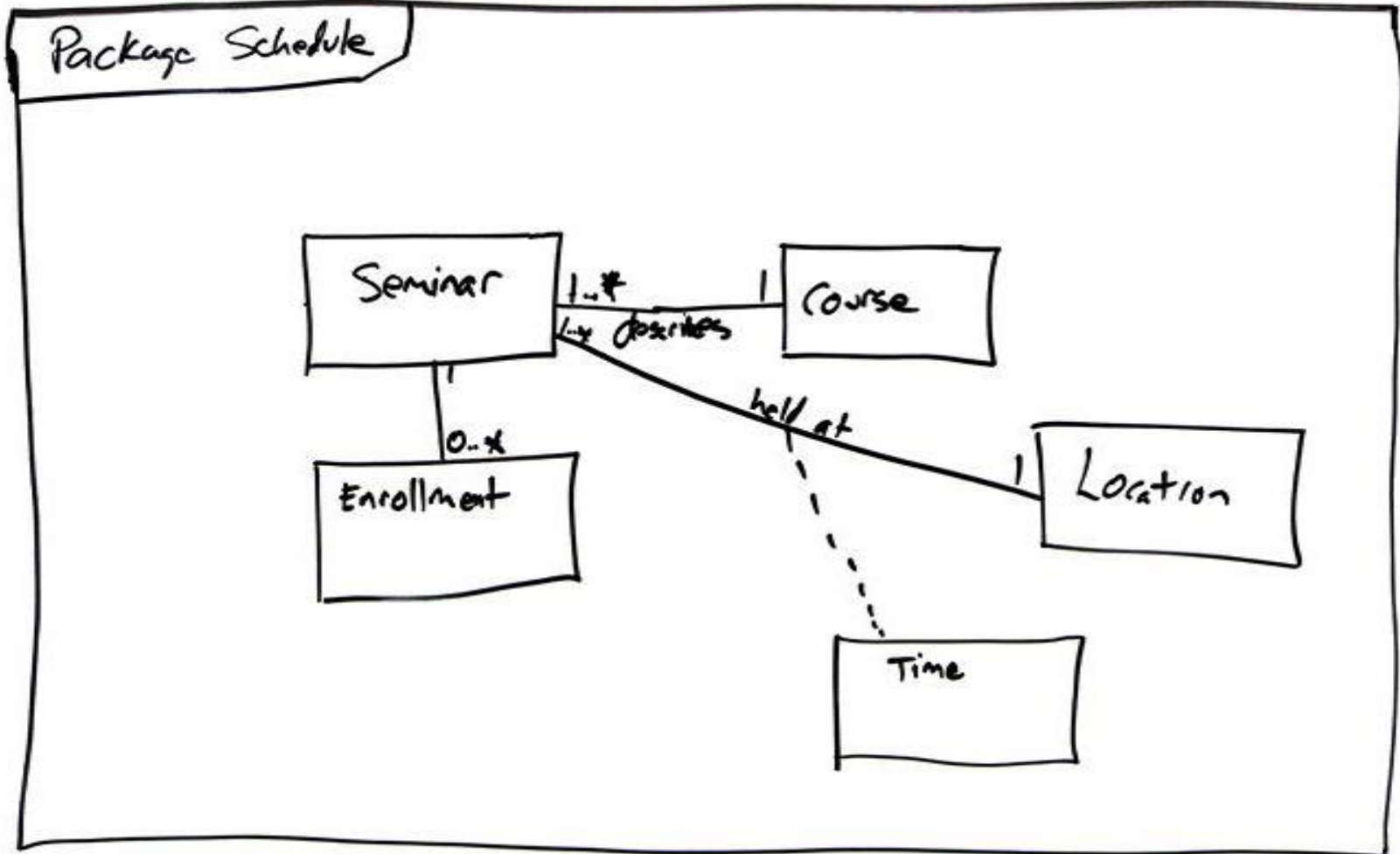
- Nguyên tắc:
  - ❑ Các lớp có quan hệ kế thừa với nhau thường thuộc vào cùng một gói.
  - ❑ Các lớp có quan hệ hợp thành với một lớp khác thường thuộc vào cùng một lớp.
  - ❑ Các lớp có tương tác với nhau thể hiện trên sơ đồ tuần tự hoặc sơ đồ cộng tác thường thuộc cùng một gói.

# Tạo gói trong sơ đồ lớp: Ví dụ





# Tạo gói trong sơ đồ lớp: Ví dụ



---

# Tóm tắt

- Sơ đồ đối tượng
- Sơ đồ tương tác đối tượng
- Sơ đồ trạng thái
- Sơ đồ hoạt động
- Sơ đồ thành phần
- Sơ đồ triển khai
- Sơ đồ gói