

# MÔ HÌNH TƯƠNG TÁC ĐỐI TƯỢNG

**DINAMIC MODEL**

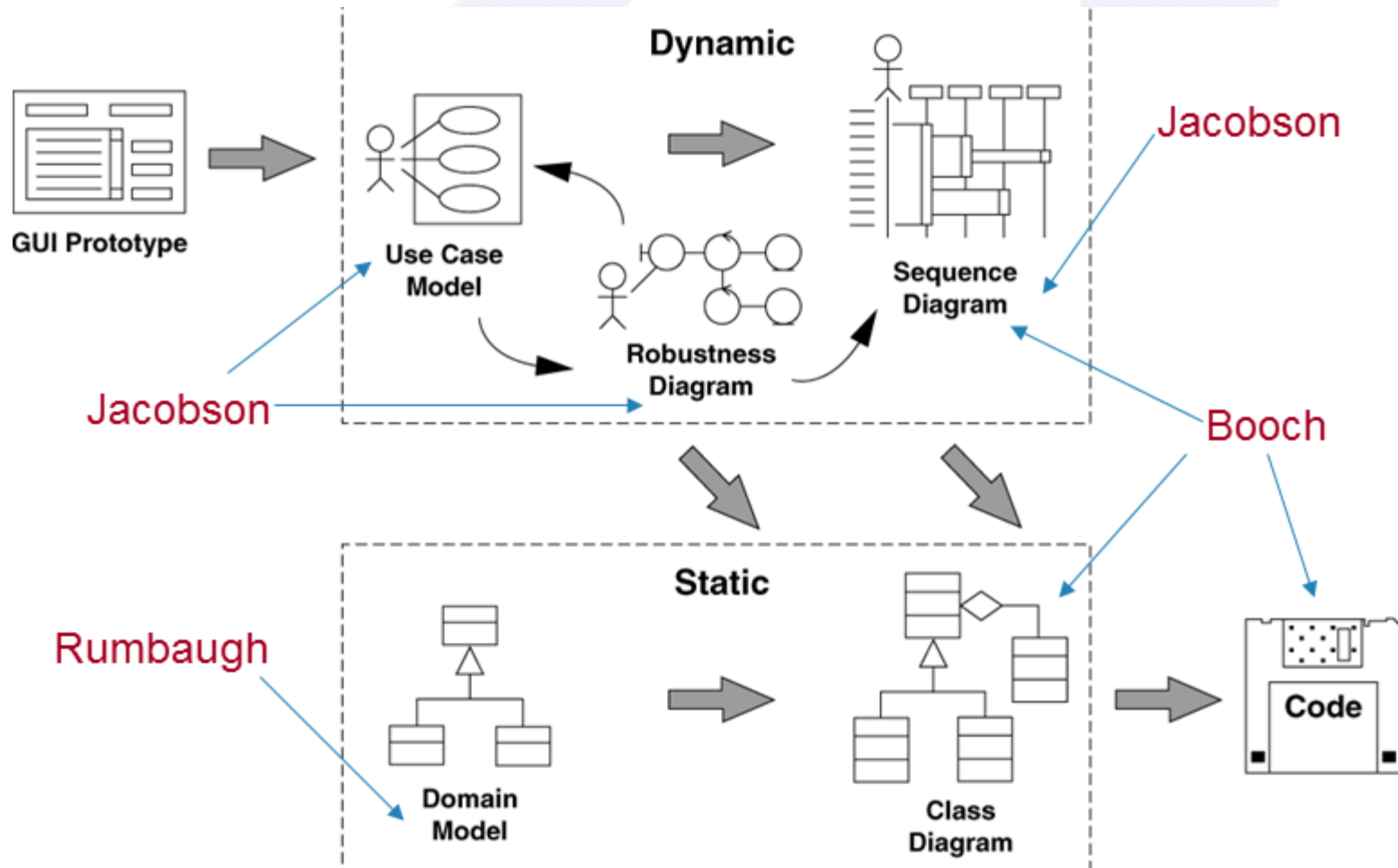


# **NỘI DUNG**

---

- 1. Khái niệm mô hình động**
- 2. Activity diagram**
- 3. Sequence diagram**
- 4. Collaboration diagram / Communication diagram**

# KHÁI NIỆM MÔ HÌNH ĐỘNG (DINAMIC MODEL)



# KHÁI NIỆM MÔ HÌNH ĐỘNG (DINAMIC MODEL)

- **Mô hình động (dynamic model).** để mô hình hóa sự hoạt động thật sự của một hệ thống và trình bày một hướng nhìn đối với hệ thống trong thời gian hệ thống hoạt động
- Hành vi của hệ thống được mô tả bằng mô hình động bao gồm:
  - Tương tác giữa các đối tượng: cộng tác hay trình tự
  - Trạng thái của đối tượng/lớp
  - Quá trình hoạt động của lớp/đối tượng







# TƯƠNG TÁC GIỮA CÁC ĐỐI TƯỢNG(1)

---

- Đối tượng tương tác (interaction) với nhau bằng cách gửi nhận các kích hoạt(stimulus)
- Actor cũng có thể gửi kích hoạt đến đối tượng
- Kích hoạt khiến một tác vụ thực thi, một đối tượng được tạo ra hay hủy đi, hoặc gây ra một tín hiệu.
- Thông điệp (message) là đặc tả của kích hoạt.

# TƯƠNG TÁC GIỮA CÁC ĐỐI TƯỢNG(1)

---

- Các loại thông điệp:
  - Đơn giản 
  - Đồng bộ 
  - Bất đồng bộ 
  - Trả về của gọi hàm 

# VAI TRÒ CỦA SƠ ĐỒ TƯƠNG TÁC

---

- UC mô tả chức năng của hệ thống, chỉ ra các actor có thể sử dụng hệ thống để làm gì (what), nhưng không chỉ ra hệ thống sẽ làm như thế nào.
- Chính các lớp và hành động (action) của các lớp sẽ thực thi các use case. Các hành động được thể hiện trong sơ đồ tương tác

# CÁC LOẠI BIỂU ĐỒ ĐỘNG

---

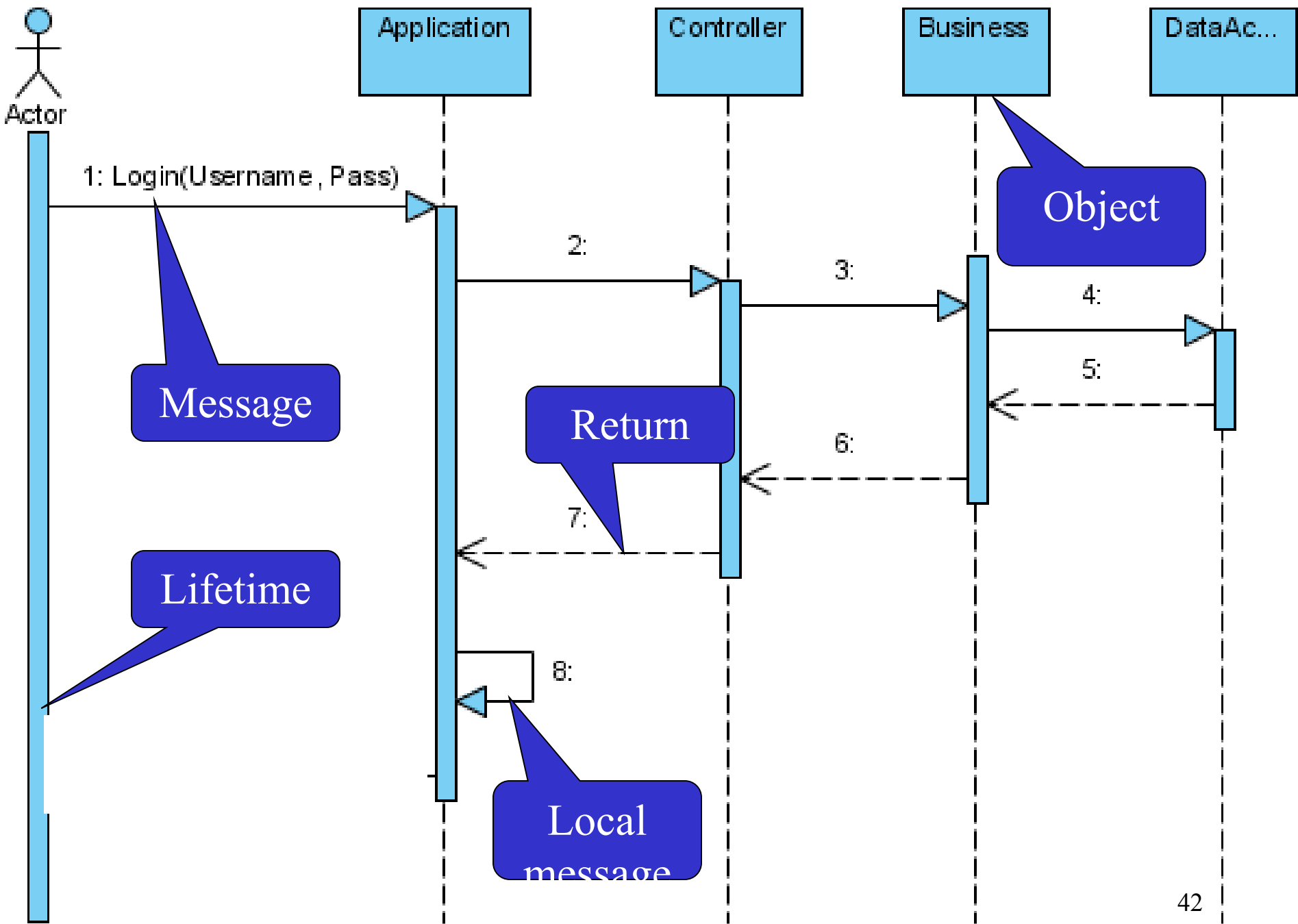
## **Bốn loại biểu đồ động trong UML**

- Sơ đồ hoạt động (activity diagram)
- Sơ đồ tuần tự (sequence diagram)
- Sơ đồ cộng tác (collaboration diagram)
- Sơ đồ trạng thái (status diagram)



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

- Sơ đồ tuần tự được sử dụng trong cả giai đoạn phân tích và thiết kế.
- Trong giai đoạn phân tích yêu cầu của bài toán, sơ đồ tuần tự được sử dụng để mô tả luồng sự kiện *theo thời gian* cấu trúc các hoạt động thực hiện một use case.
- Sơ đồ tuần tự biểu diễn chi tiết quan hệ *giao tiếp giữa các đối tượng* trong quá trình thực hiện use case
- Sơ đồ tuần tự có hai trục: trục nằm dọc chỉ thời gian, trục nằm ngang chỉ ra một tập hợp các đối tượng.
- Các mối liên kết không được thể hiện trong sơ đồ.
- Có sự liên kết chặt chẽ với sơ đồ lớp.



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

---

- **Ưu điểm**

- Biểu diễn rõ ràng trình tự các thông điệp tương tác giữa các đối tượng trong các trường hợp phức tạp

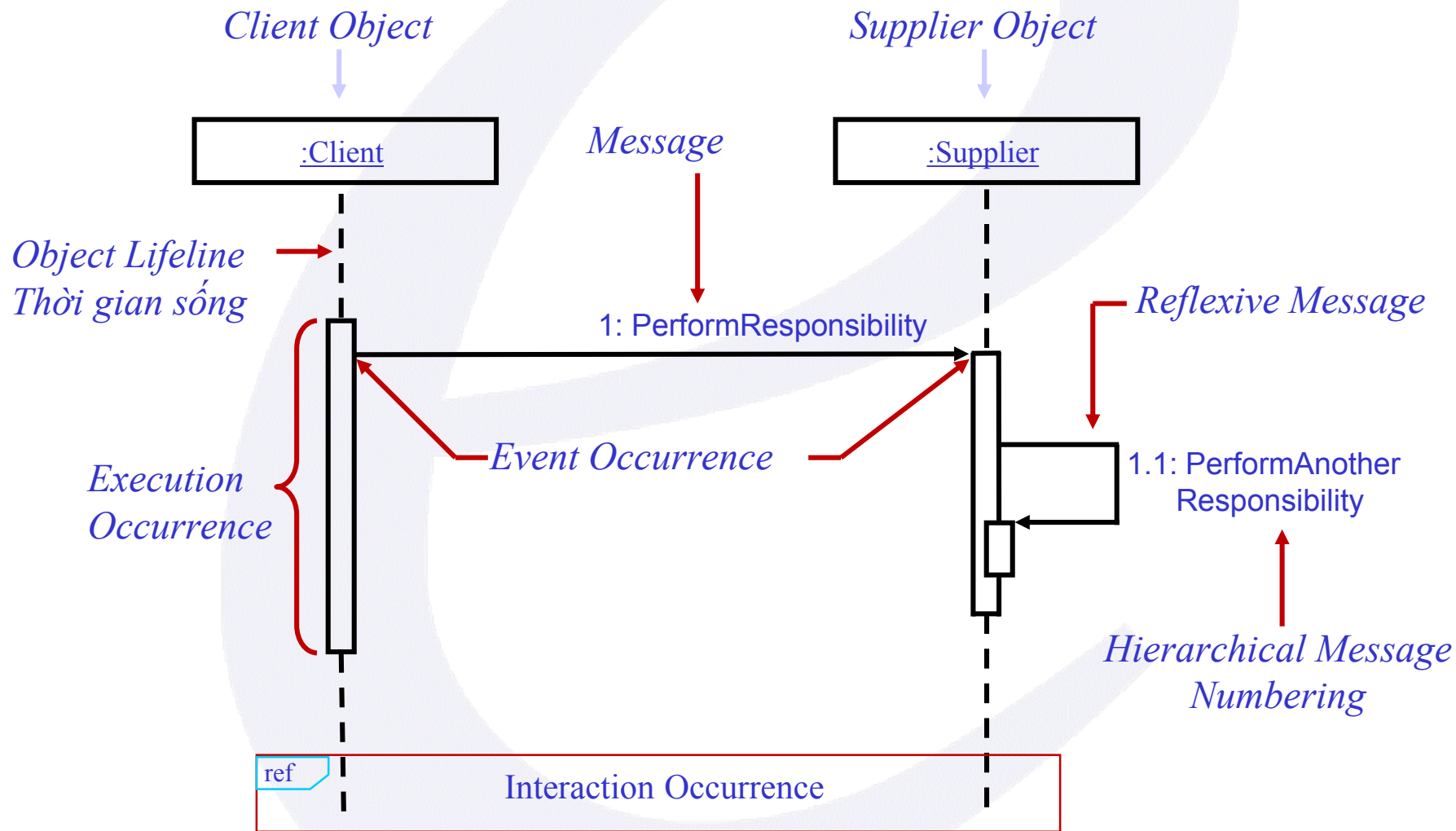
- **Nhược điểm**

- Chiếm không gian theo chiều ngang khi thêm đối tượng mới

- **Ứng dụng Sequence Diagram**

- ☐ Thiết kế các chức năng
- ☐ Kiểm chứng và bổ sung method cho các Class
- ☐ Sử dụng trong việc coding các chức năng

# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

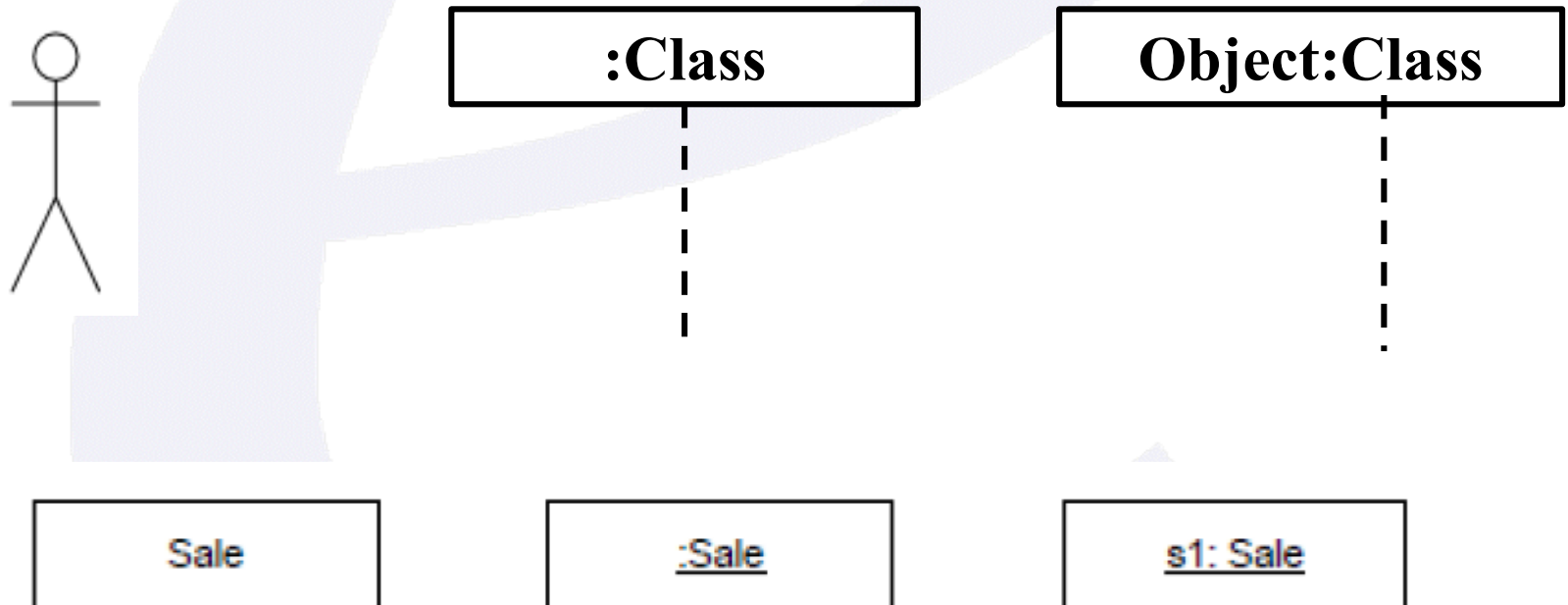


Các thuật ngữ dùng cho lược đồ tuần tự (sequence diagram)



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

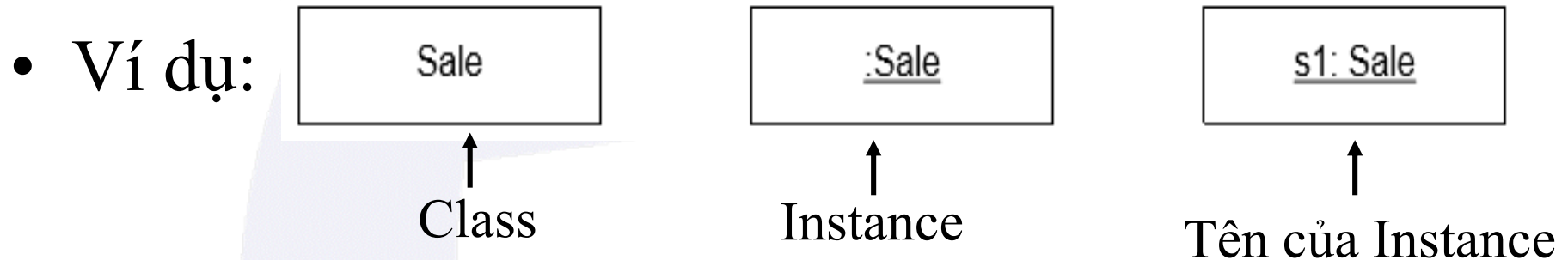
- **Đối tượng tham gia (Participant):** đối tượng thực hiện hành động trong sơ đồ trình tự.
  - **Ký hiệu trong UML**



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

- **Lớp (Classes) và thể hiện (Instances)**

- Trong UML, một *thể hiện (instances)* của một *lớp (class)* có ký hiệu giống như Lớp, nhưng *tên của thể hiện (instances)* được gạch chân.

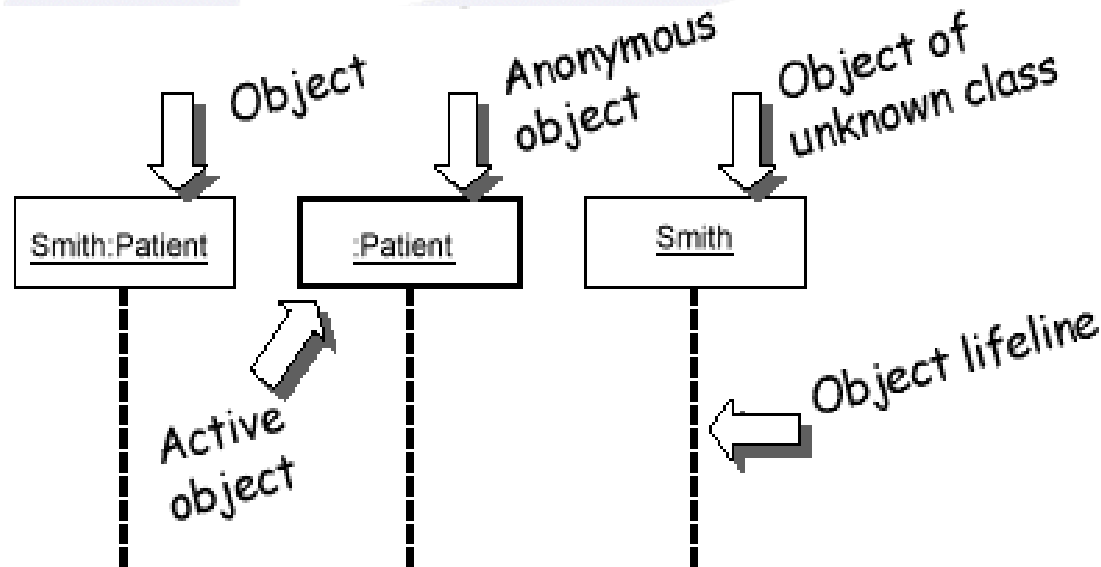


- *Tên của một Instance là duy nhất trong sơ đồ, nếu không đặt tên thì đặt dấu : trước tên Lớp và gạch chân*

# SƠ ĐỒ TRÌNH TỰ - SEQUENCE DIAGRAM

## Ánh xạ đối tượng vào lớp Mapping an Object to a Class

- Tất cả các đối tượng trong sơ đồ tuần tự cần được ánh xạ (map) vào một lớp nào đó.
- Có thể gán cho đối tượng thuộc 1 lớp đã được định nghĩa sẵn trong mô hình domain, hay gán cho nó 1 lớp mới

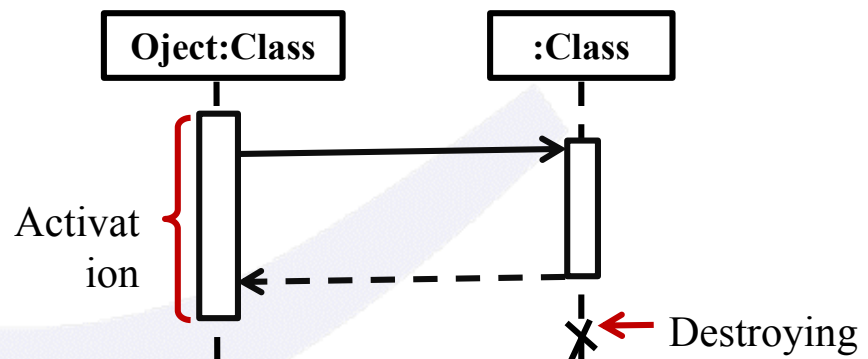


**Name syntax:** <objectname>:<classname>

# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

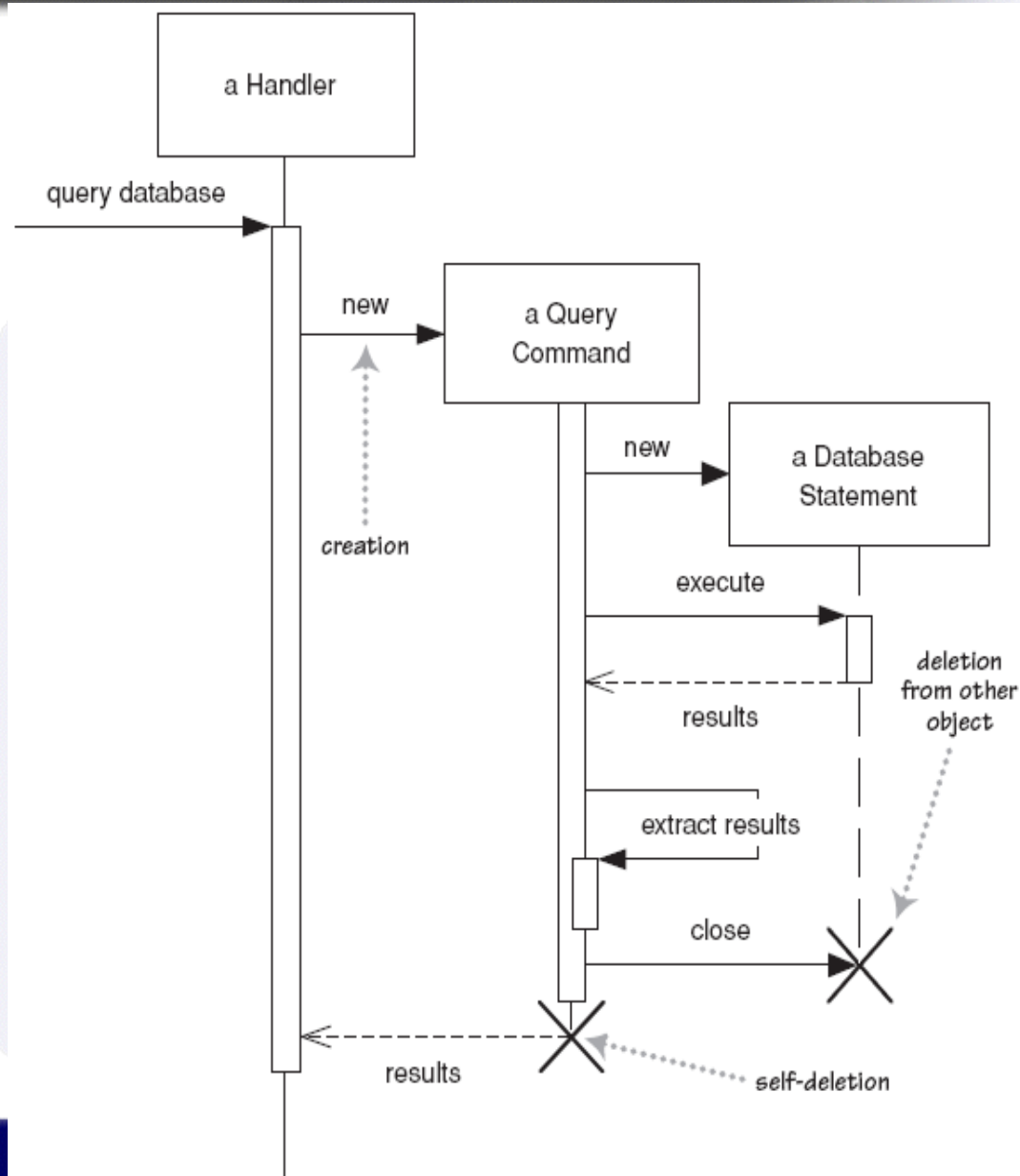
Các thành phần của Sequence diagram:

- **Lifeline (thời gian sống):** là một yếu tố được đặt tên đại diện cho một cá nhân tham gia trong sự tương tác. Biểu diễn thời gian sống của đối tượng trong sơ đồ tuần tự,
  - **Kích hoạt (Activation):** biểu diễn thời gian một đối tượng đang ở trạng thái hoạt động.
  - **Kết thúc đối tượng (Destroying):** đối tượng kết thúc sau khi hoàn tất hoạt động.
  - **Ký hiệu trong UML**





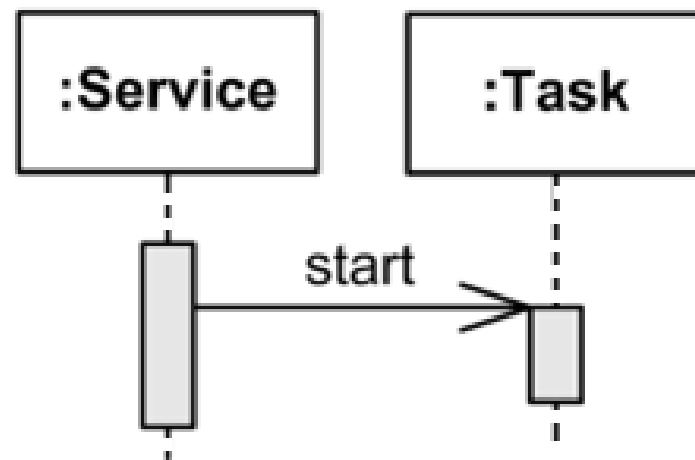


# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM




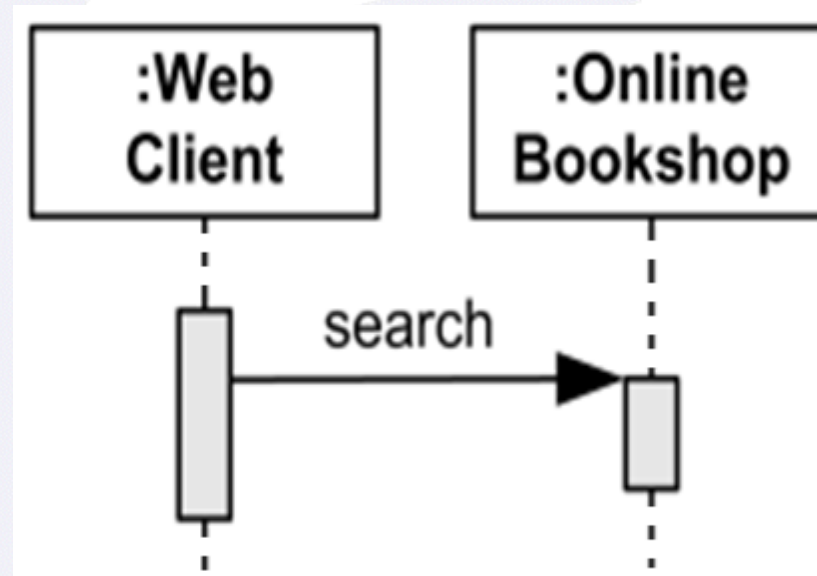
# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

- **Thông điệp (Messages):** biểu diễn giao tiếp giữa các đối tượng.
  - **Thông điệp không đồng bộ:** được gửi từ một đối tượng sẽ không chờ thông điệp trả về từ đối tượng nhận trước khi tiếp tục.
  - **Ký hiệu trong UML:** 
  - Ví dụ: 



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

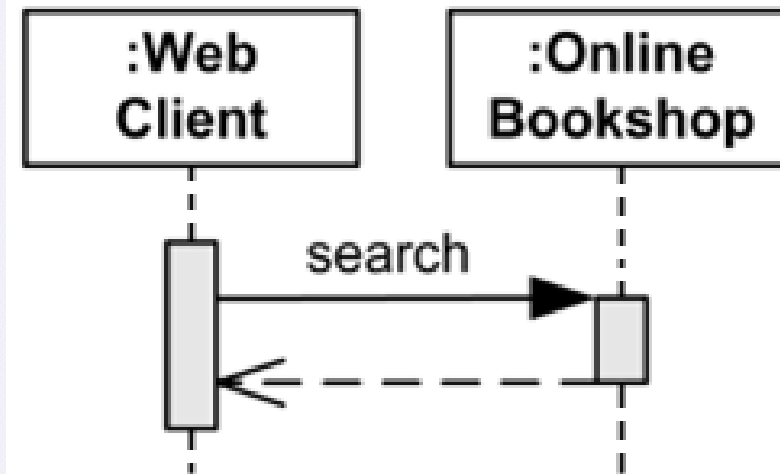
- **Thông điệp đồng bộ:** đối tượng gửi thông điệp chờ đến khi thông điệp được xử lý trước khi tiếp tục.
- **Ký hiệu trong UML:** 



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

- **Return Message**

- Thông điệp trả về kết quả cho đối tượng gửi.
- Ký hiệu trong UML ←-----

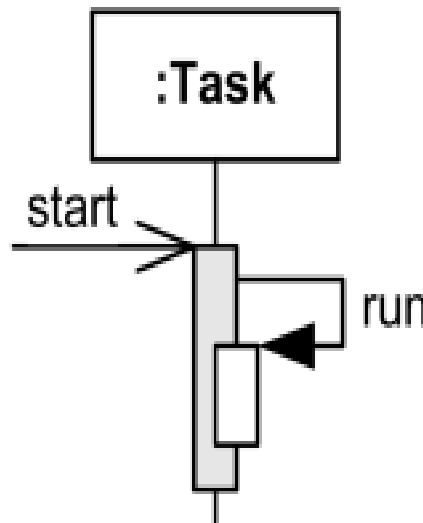




# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

- **Self Message**

- Một một cuộc gọi đệ quy của một hoạt động, hoặc một phương thức gọi một phương thức khác trên cùng một đối tượng.
- **Ký hiệu:**

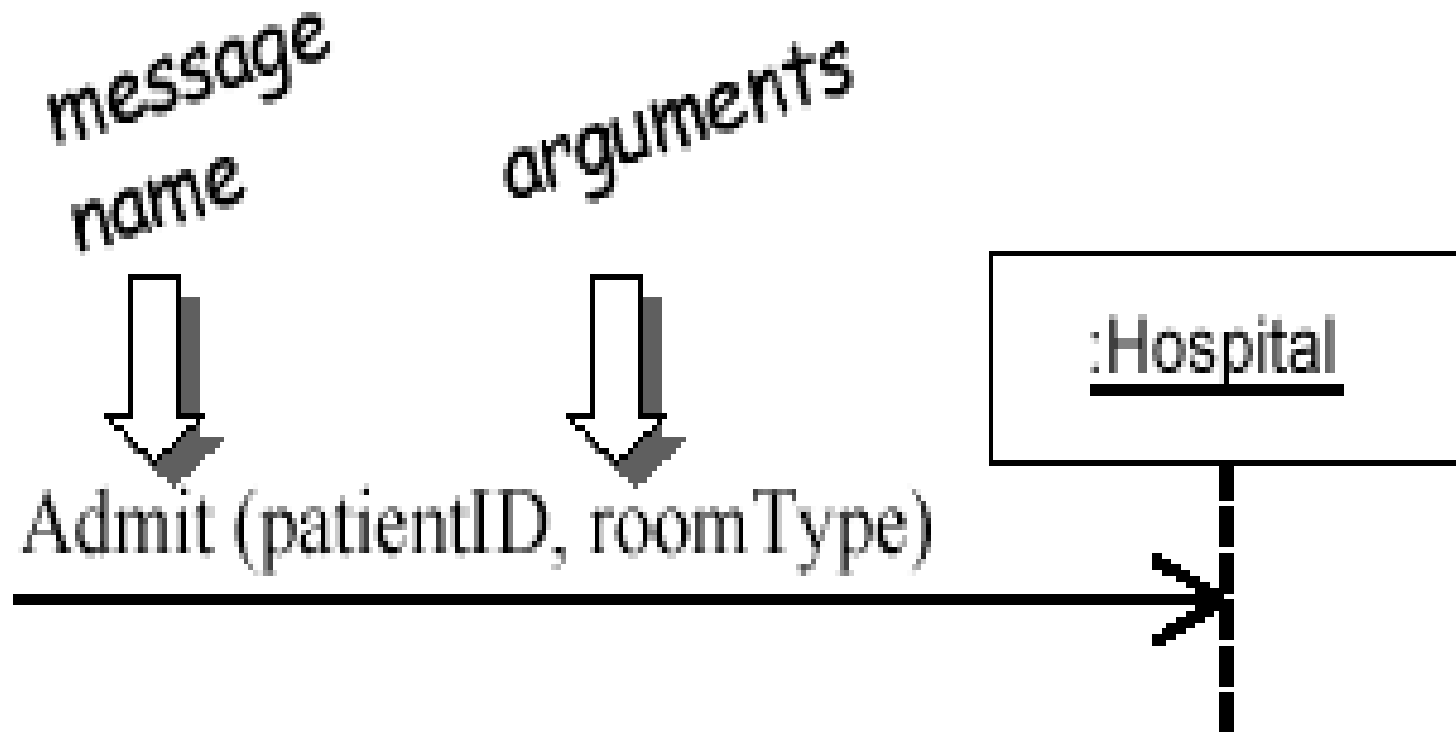


# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

## Thông điệp (Message)

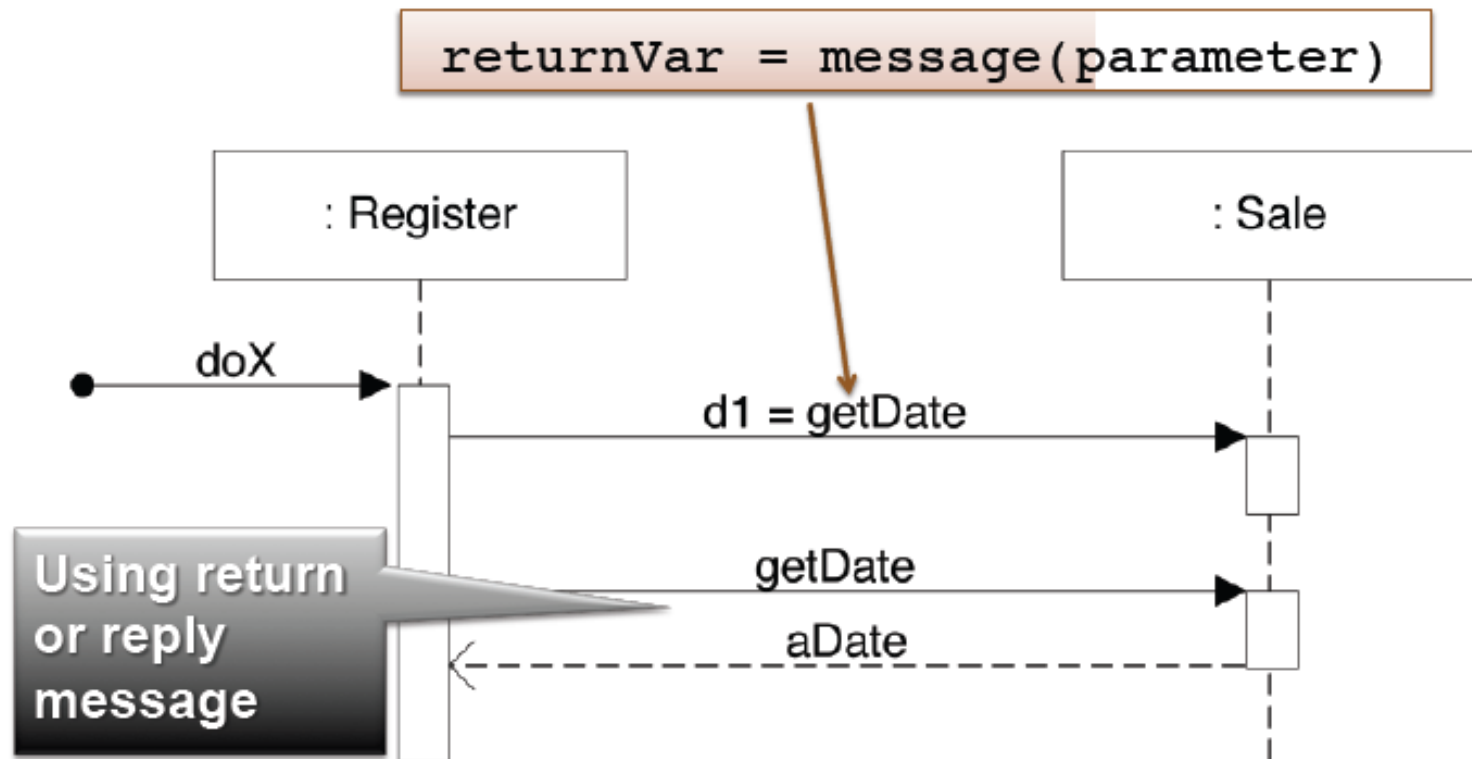
- Mỗi thông điệp đều có cú pháp như sau:
  - `return := message(parameter : parameterType) : returnType`
  - Parameter: là tham số của thông điệp,
  - returnType: loại của giá trị trả về (tùy chọn)
- Ví dụ :
  - `spec := getProductSpect(id)`
  - `spec := getProductSpect(id:ItemID)`
  - `spec := getProductSpect(id:ItemID): ProductSpect`

# BIỂU DIỄN THÔNG điệp



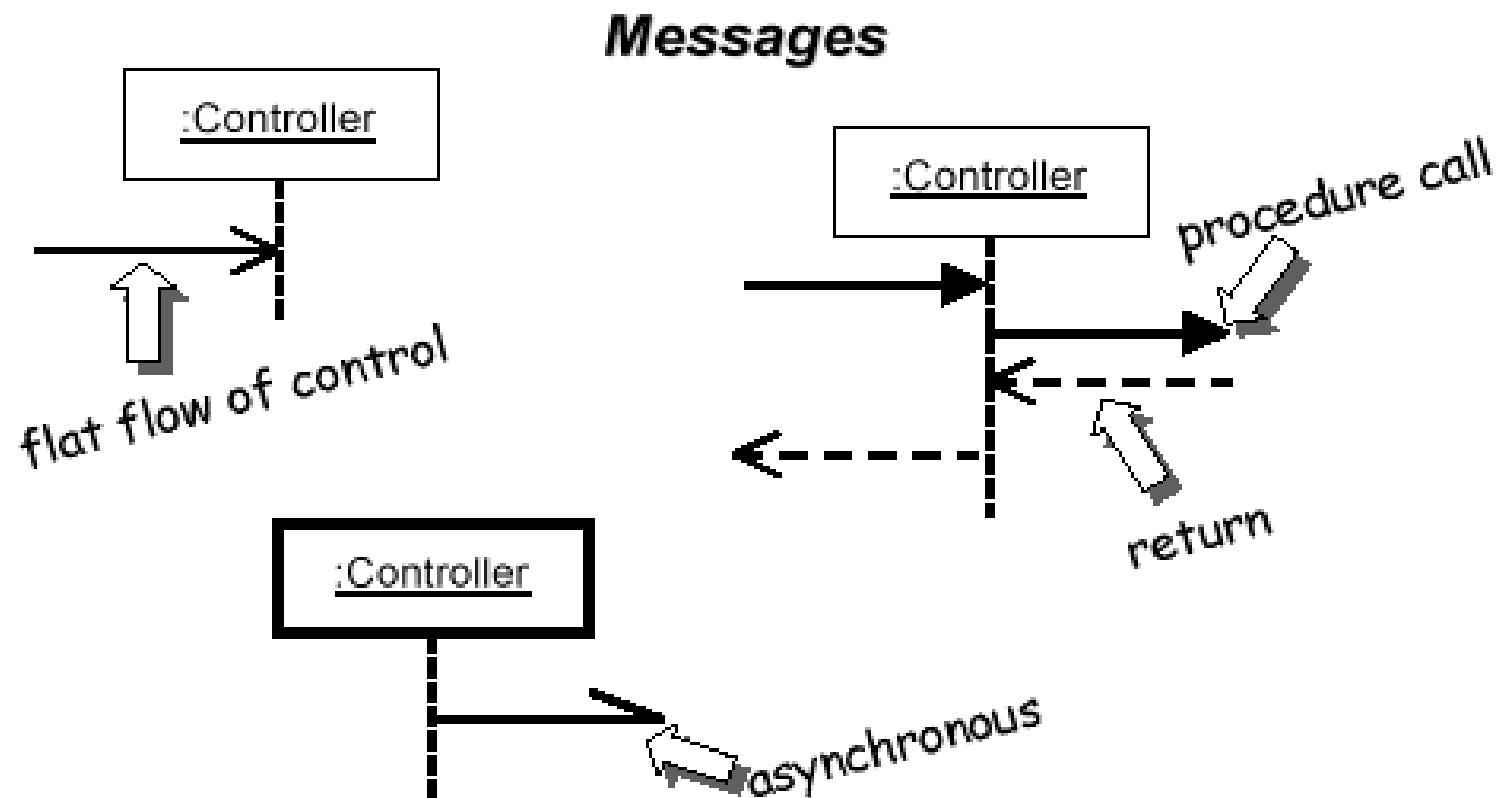
# SƠ ĐỒ TRÌNH TỰ - SEQUENCE DIAGRAM

## Two Ways of Illustrating Return Values



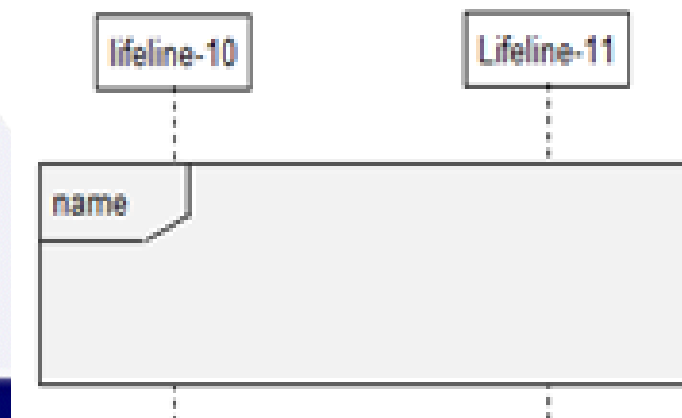


# BIỂU DIỄN THÔNG điệp



# ĐIỀU HƯỚNG GỌI METHODS

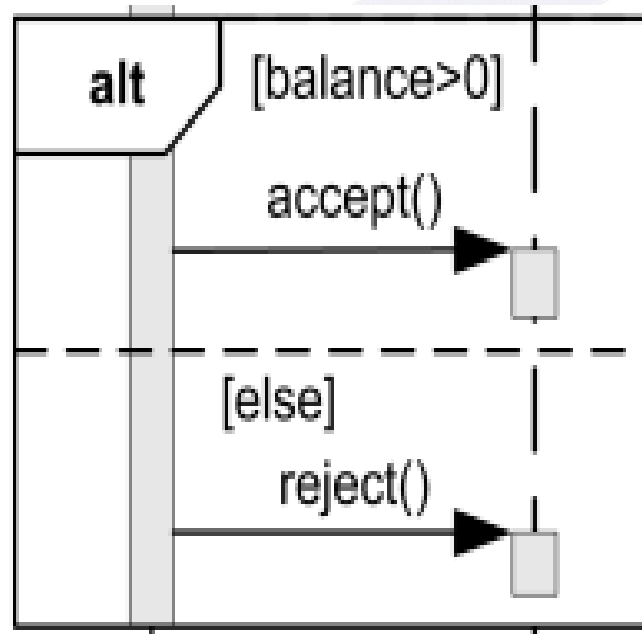
- **Frame:** một hộp biểu diễn một phần của sơ đồ tuần tự để thể hiện sự lựa chọn hoặc lặp
- hộp xung quanh một phần của biểu đồ trình tự để biết sự lựa chọn hoặc loop
  - **if** -> (opt) [condition]
  - **if/else** -> (alt) [condition], separated by horizon. Dashed line
  - **loop** -> (loop) [condition or items to loop over]



# ĐIỀU HƯỚNG GỌI METHODS

- **Alt**

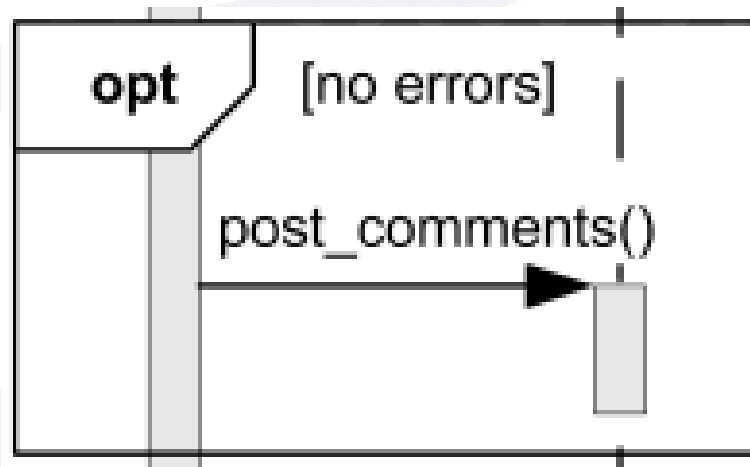
- Biểu diễn cho một sự lựa chọn hoặc thay thế của hành vi.
- Ví dụ:



# ĐIỀU HƯỚNG GỌI METHODS

- **Option**

- Đại diện cho một sự lựa chọn của hành vi mà một trong hai (duy nhất) toán hạng sẽ xảy ra hoặc không có gì xảy ra.
- Ví dụ:

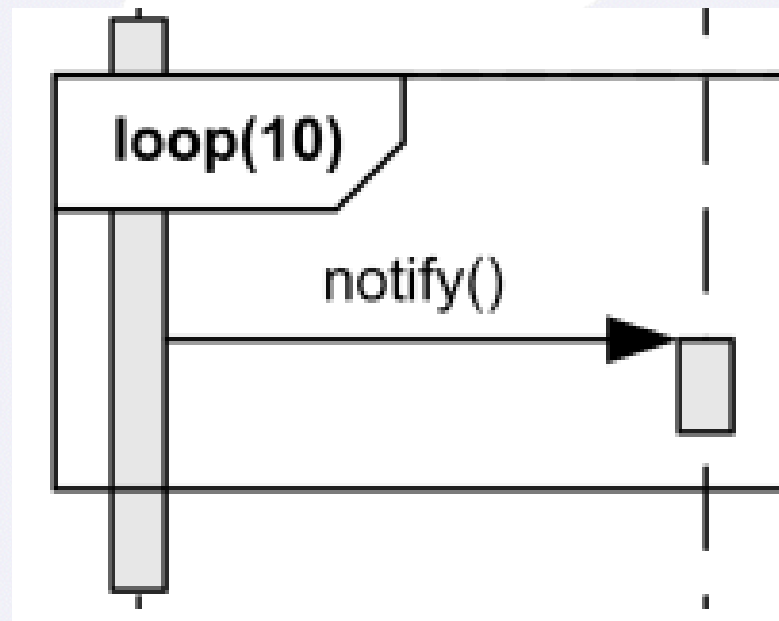




# ĐIỀU HƯỚNG GỌI METHODS

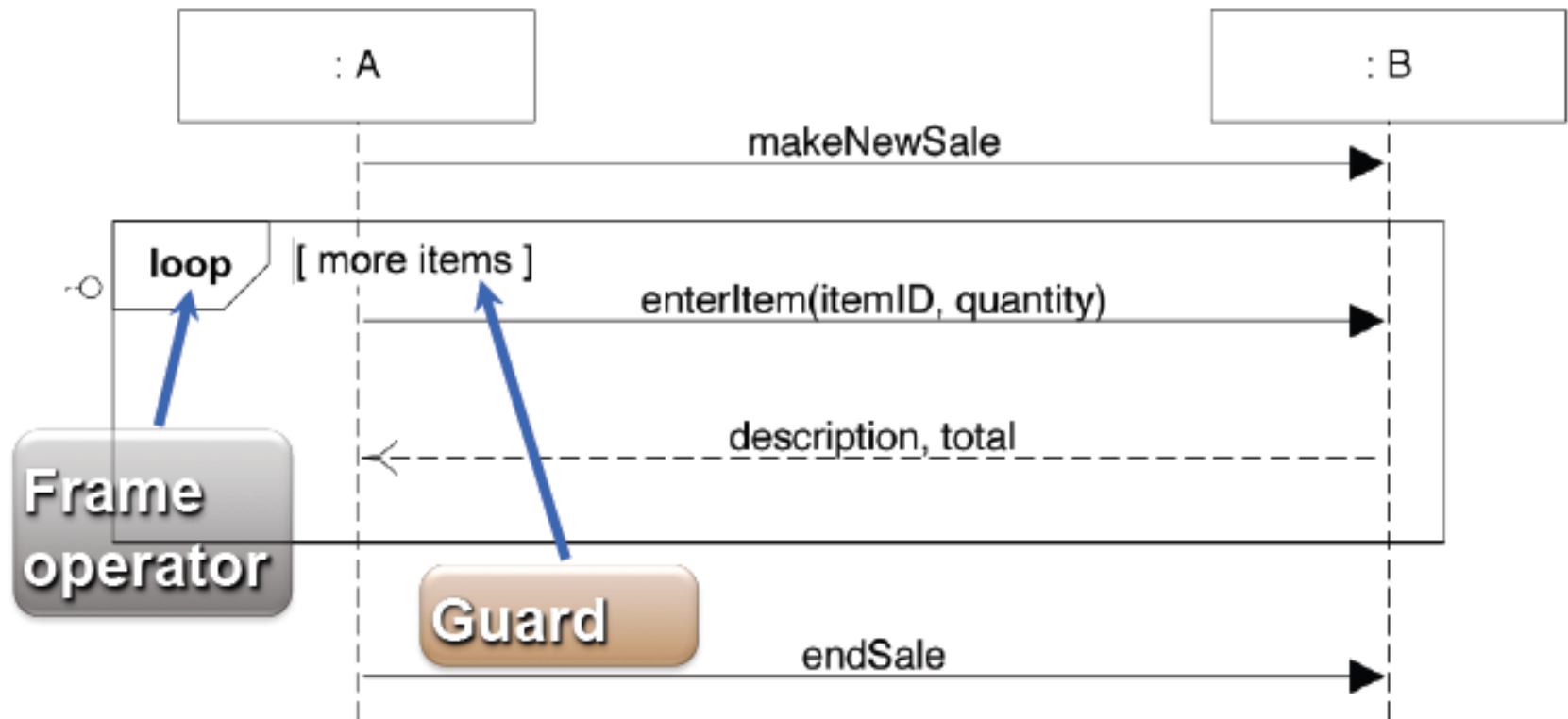
- **Loop**

- Vòng lặp sẽ được thực hiện chính xác số lần quy định.

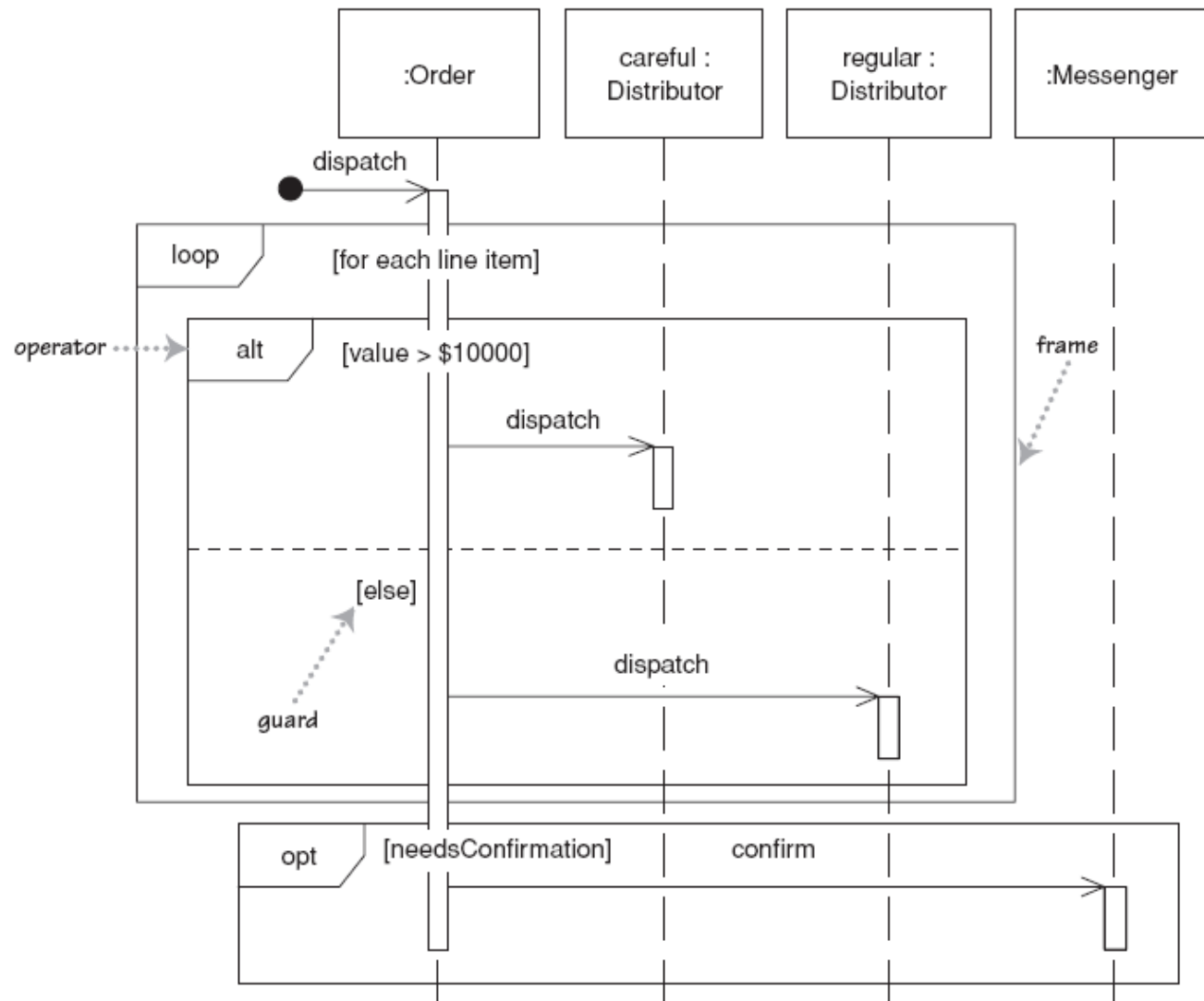


# SƠ ĐỒ TRÌNH TỰ - SEQUENCE DIAGRAM

## Interaction Frame



# SELECTION AND LOOPS



# LINKING SEQUENCE DIAGRAMS

- If one diagram is too large or refers to another, indicate with:
  - an unfinished arrow and comment,
  - or a "ref" frame that names the other diagram
- when would this occur in our system?

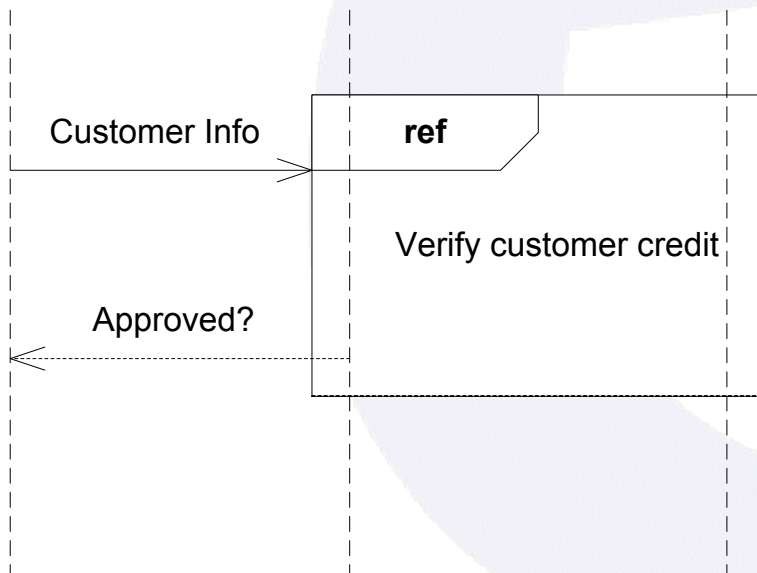


Diagram 1

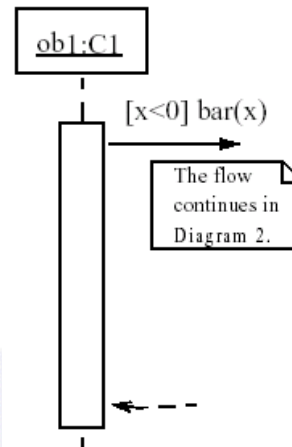
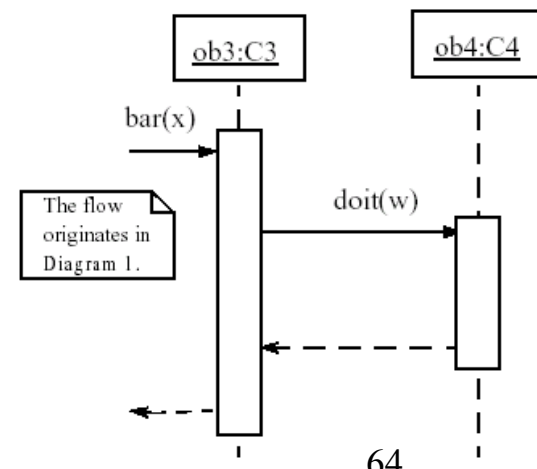
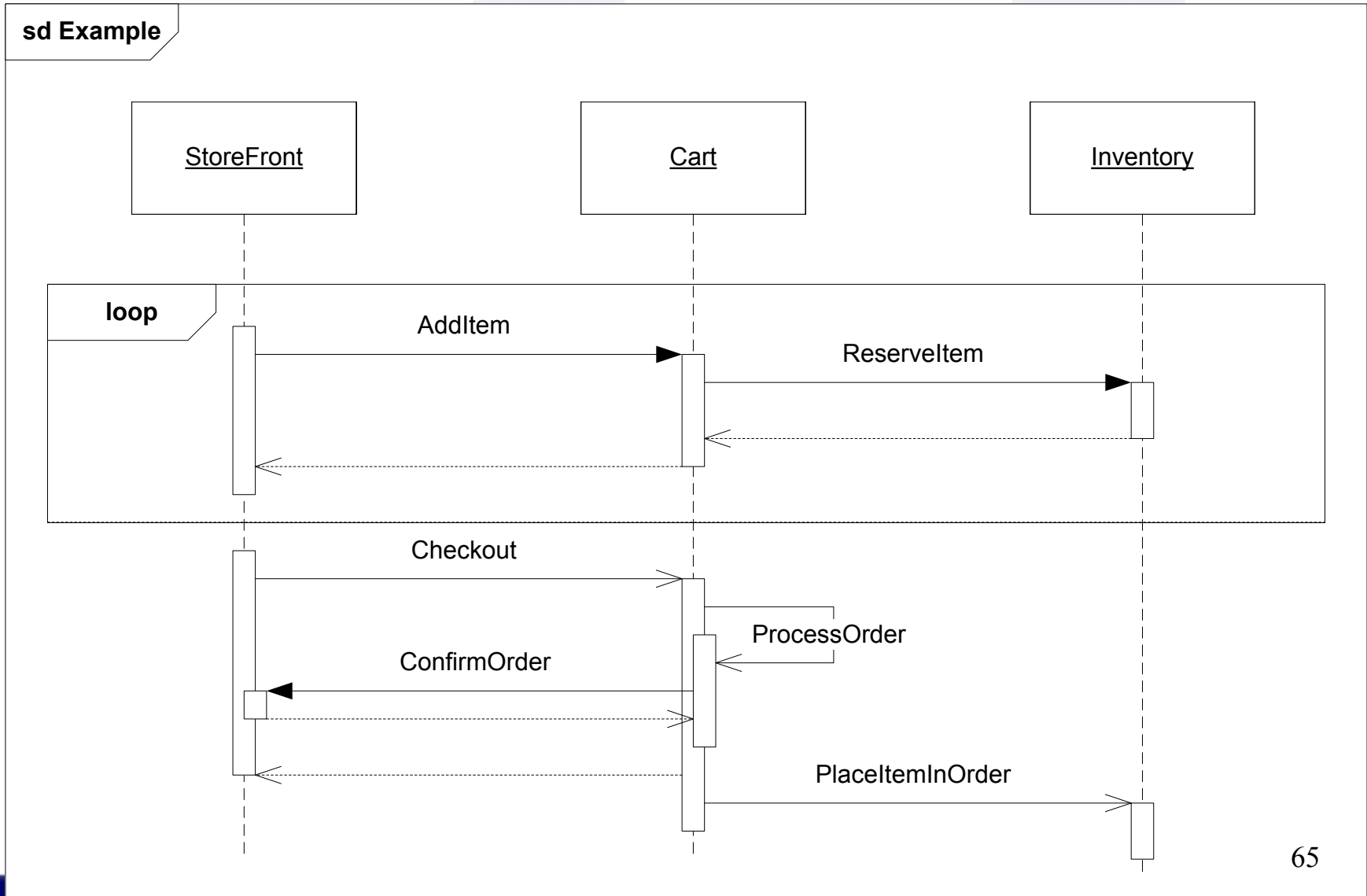


Diagram 2





# VÍ DỤ LINKING SEQUENCE DIAGRAMS



# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

---

## Xây dựng Sequence Diagram

- **Bước 1:** Xác định chức năng cần thiết kế. Bạn dựa vào Use Case Diagram để xác định xem chức năng nào cần thiết kế.
- **Bước 2:** Dựa vào Activity Diagram để xác định các bước thực hiện theo nghiệp vụ.
- **Bước 3:** Đối chiếu với Class Diagram để xác định lớp trong hệ thống tham gia vào nghiệp vụ.
- **Bước 4:** Vẽ Sequence Diagram
- **Bước 5:** Cập nhật lại bản vẽ Class Diagram

# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

---

Khi vẽ sơ đồ tuần tự, cần chú ý:

- Sự kiện được biểu diễn bằng các đường thẳng nằm ngang.
- Đối tượng bằng các đường nằm dọc.
- Thời gian được thể hiện bằng đường thẳng nằm dọc bắt đầu từ trên sơ đồ. Điều đó có nghĩa là các sự kiện cần phải được thể hiện theo đúng thứ tự mà chúng xảy ra, vẽ từ trên xuống dưới.

# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

**Xây dựng Sequence Diagram cho hệ thống eCommerce**

***Bước 1: Xác định các Use Case cần thiết kế***

- Các Use Case sau cần thiết kế:
  - ☐ Xem sản phẩm theo chủng loại
  - ☐ Thêm sản phẩm theo nhà cung cấp
  - ☐ Thêm giỏ hàng
  - ☐ Chat
  - ☐ Quản lý đơn hàng
  - ☐ Thanh toán
  - ☐ Theo dõi chuyển hàng
  - ☐ Đăng nhập
- Tiếp theo, chúng ta sẽ thiết kế cho chức năng “**Xem sản phẩm theo chủng loại**”.



# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

**Xây dựng Sequence Diagram cho hệ thống eCommerce**

***Bước 2: Xem Activity Diagram cho Use Case này chúng ta xác định các bước sau:***

- ☐ Người dùng chọn loại sản phẩm
- ☐ Hệ thống sẽ lọc lấy loại sản phẩm tương ứng, sau đó lấy giá, lấy khuyến mãi và hiển thị lên màn hình.
- ☐ Người dùng xem sản phẩm

# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

**Xây dựng Sequence Diagram cho hệ thống eCommerce**

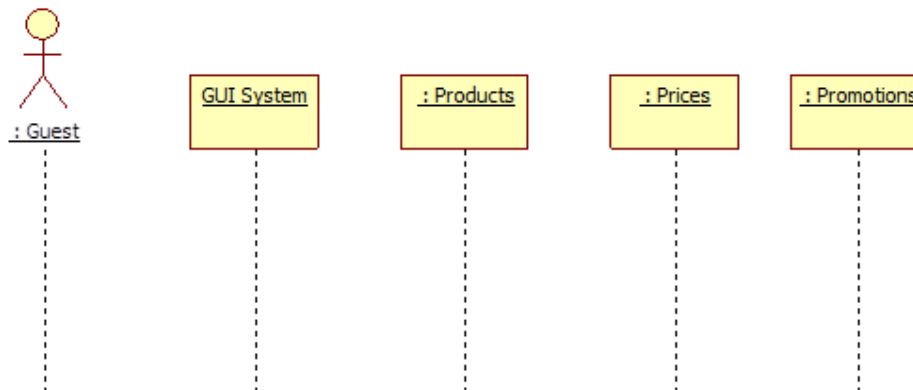
***Bước 3: Đối chiếu với Class Diagram ta xác định các đối tượng thực hiện như sau:***

- **Người dùng:** chọn loại sản phẩm qua giao diện
- **Giao diện:** sẽ lấy danh sách sản phẩm tương ứng từ **Products**
- **Giao diện:** lấy giá của từng sản phẩm từ Class **Prices** và Promotion Amount từ lớp **Promotions**
- **Giao diện:** tổng hợp danh sách và hiển thị
- **Người dùng:** Xem sản phẩm

# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

## *Bước 4: Vẽ sequence Diagram*

- Xác định các lớp tham gia vào hệ thống gồm: người dùng (Guest), Giao diện (GUI System), Sản phẩm (Products), Giá (Prices), Khuyến mãi (Promotions).
- Trong đó GUI System để sử dụng chung cho giao diện, bạn có thể sử dụng cụ thể trang Web nào nếu bạn đã có Mockup (thiết kế chi tiết của giao diện).

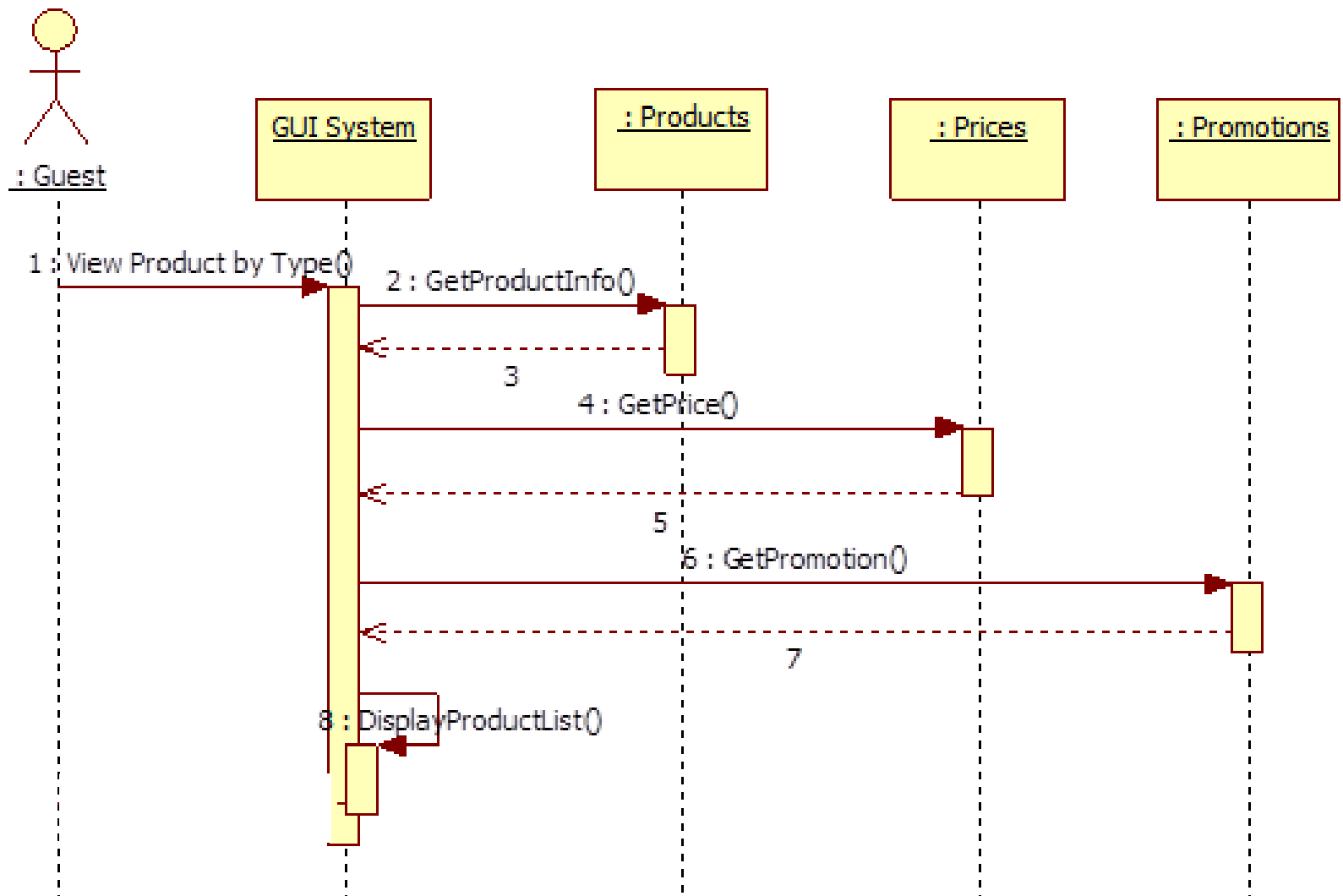


# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

Các bước thực hiện của Use Case này như sau:

- **Guest** gửi yêu cầu xem sản phẩm lên giao diện kèm theo chủng loại
- **GUI system**: gửi yêu cầu lấy danh sách các sản phẩm tương ứng với chủng loại cho lớp sản phẩm và nhận lại danh sách.
- **GUI system**: gửi yêu cầu lấy Giá cho từng sản phẩm từ Prices
- **GUI system**: gửi yêu cầu lấy khuyến mãi cho từng sản phẩm từ Promotions và nhận lại kết quả
- **GUI system**: ghép lại danh sách và hiển thị lên browser và trả về cho Guest

# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM





# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

## *Bước 5: Kiểm tra và cập nhật bản vẽ Class Diagram*

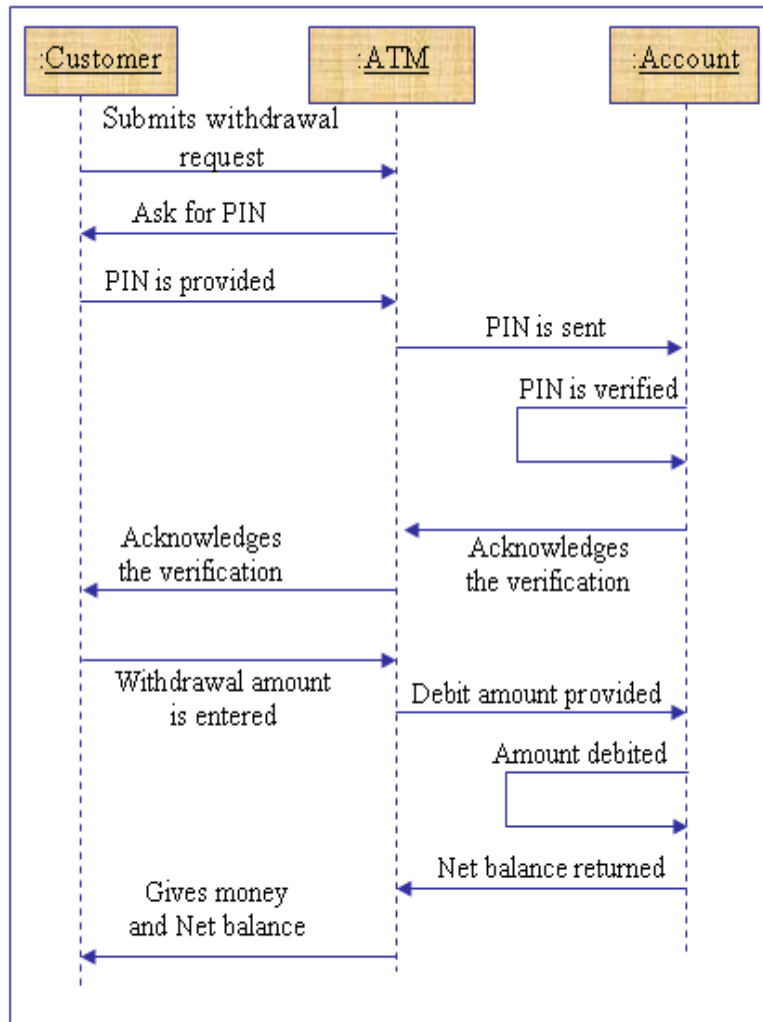
- Chúng ta nhận thấy để thực hiện được bản vẽ trên chúng ta cần bổ sung các phương thức cho các lớp như sau:
- **Products class:** bổ sung phương thức **GetProductInfo(Product Type):** trả về thông tin sản phẩm có loại được truyền vào. Việc này các đối tượng của lớp Products hoàn toàn làm được vì họ đã có thuộc tính ProductType nên họ có thể trả về được thông tin này.
- **Prices:** bổ sung phương thức **GetPrice(ProductID):** UnitPrice. Sau khi lấy được ProductID từ Products, GUI gọi phương thức này để lấy giá của sản phẩm từ lớp giá. Các đối tượng từ lớp Prices hoàn toàn đáp ứng điều này.

# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

## *Bước 5: Kiểm tra và cập nhật bản vẽ Class Diagram*

- **Promotions:** tương tự bổ sung phương thức **GetPromotion(ProductID)**.
- **GUI System(View Product Page):** bổ sung phương thức
- **DisplayProductList(List of product)** để hiển thị danh sách lên sản phẩm. Ngoài ra, bạn cần có thêm một phương thức **ViewProductbyType(ProductType)** để mô tả chính hoạt động này khi người dùng kích chọn.

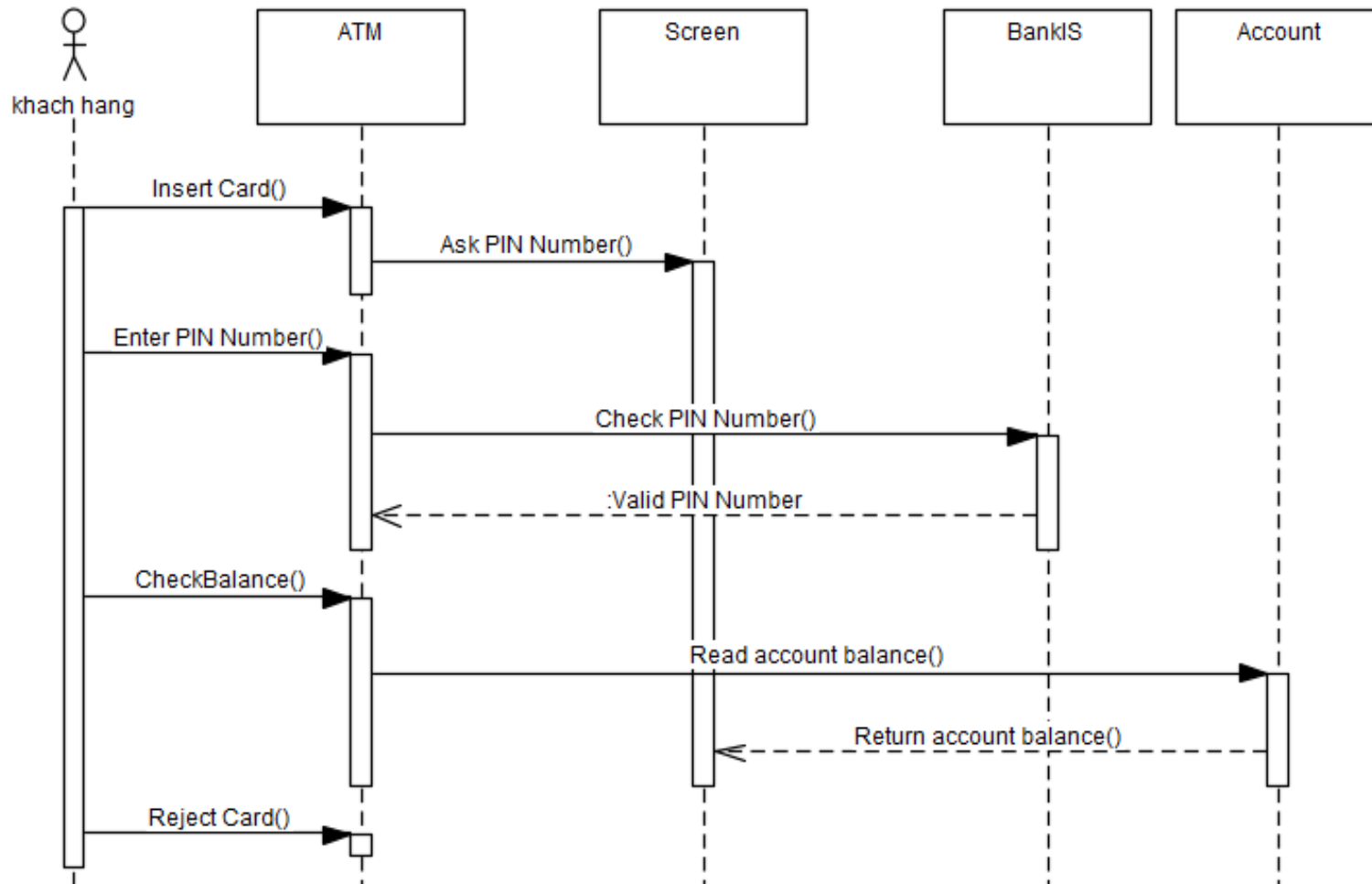
# CÁC VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM



- Có ba lớp tham gia cảnh kịch này: khách hàng, máy ATM và tài khoản.
- Khách hàng đưa thẻ tiền vào máy ATM
- Đối tượng máy ATM yêu cầu khách hàng cung cấp mã số
- Mã số được gửi cho hệ thống để kiểm tra tài khoản
- Đối tượng tài khoản kiểm tra mã số 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 mức tiền trong tài khoản. Tại thời điểm này, chúng ta thấy có một mũi tên quay trở lại chỉ vào đối tượng tài khoản. Ý nghĩa của nó là đối tượng tài khoản xử lý yêu cầu này trong nội bộ đối tượng và không gửi sự kiện đó ra ngoài.
- Đối tượng tài khoản trả về mức tiền 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.

Hình 6.4- Biểu đồ cảnh kịch rút tiền mặt tại máy ATM

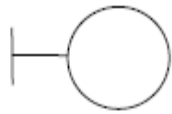
# CÁC VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCE DIAGRAM



# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM



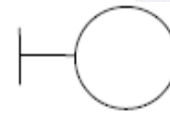
: Student



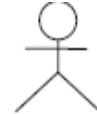
RegisterForCoursesForm



RegistrationController



CourseCatalogSystem



Course Catalog

1: // create schedule()

2: // get course offerings()

3: // get course offerings(forSemester)

4: // get course offerings()

5: // display course offerings()

6: // display blank schedule()

Create a new  
schedule

A list of the available  
course offerings for this  
semester are displayed

A blank schedule  
is displayed for the  
students to select  
offerings

ref

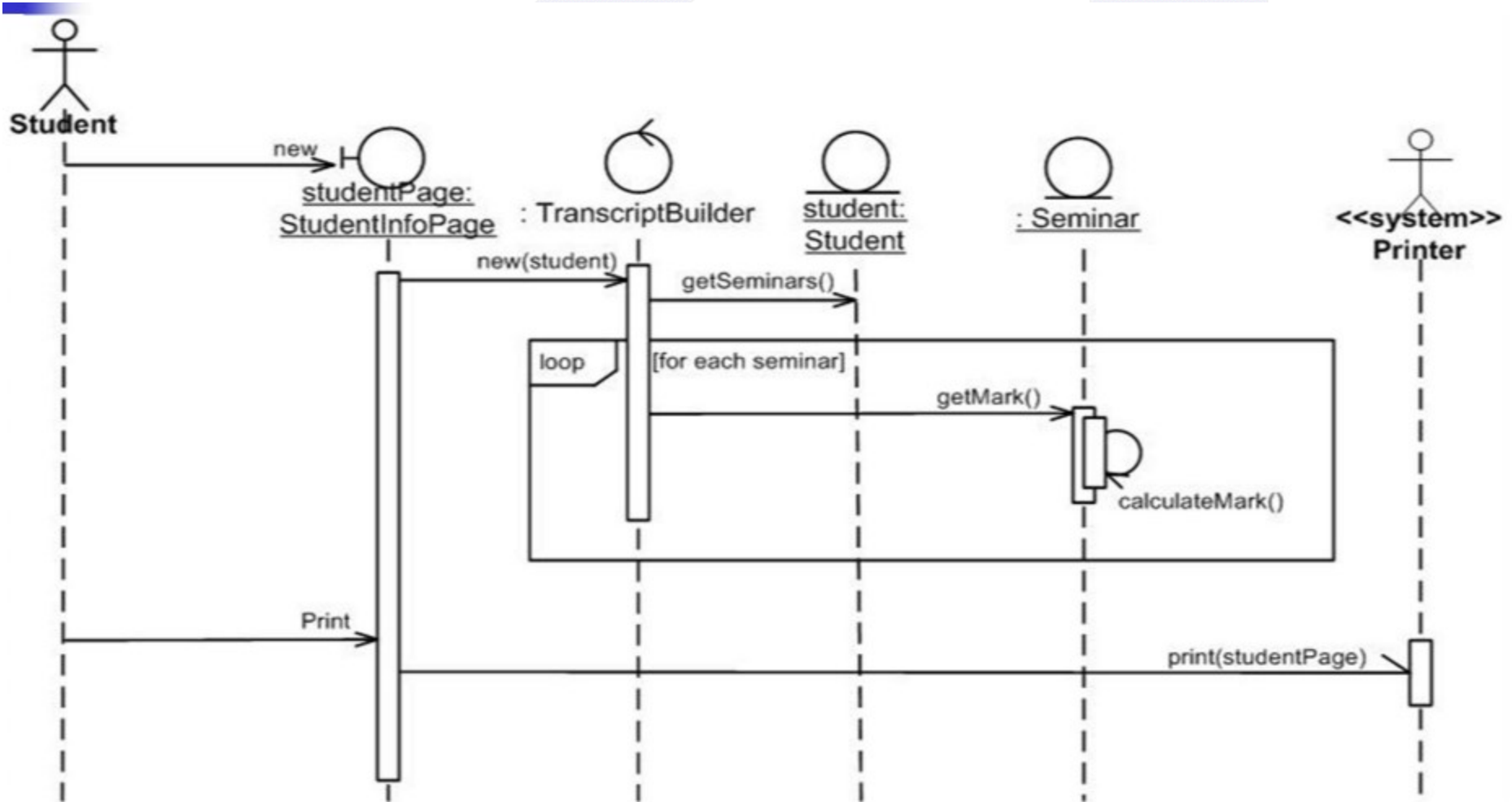
Select Offerings

ref

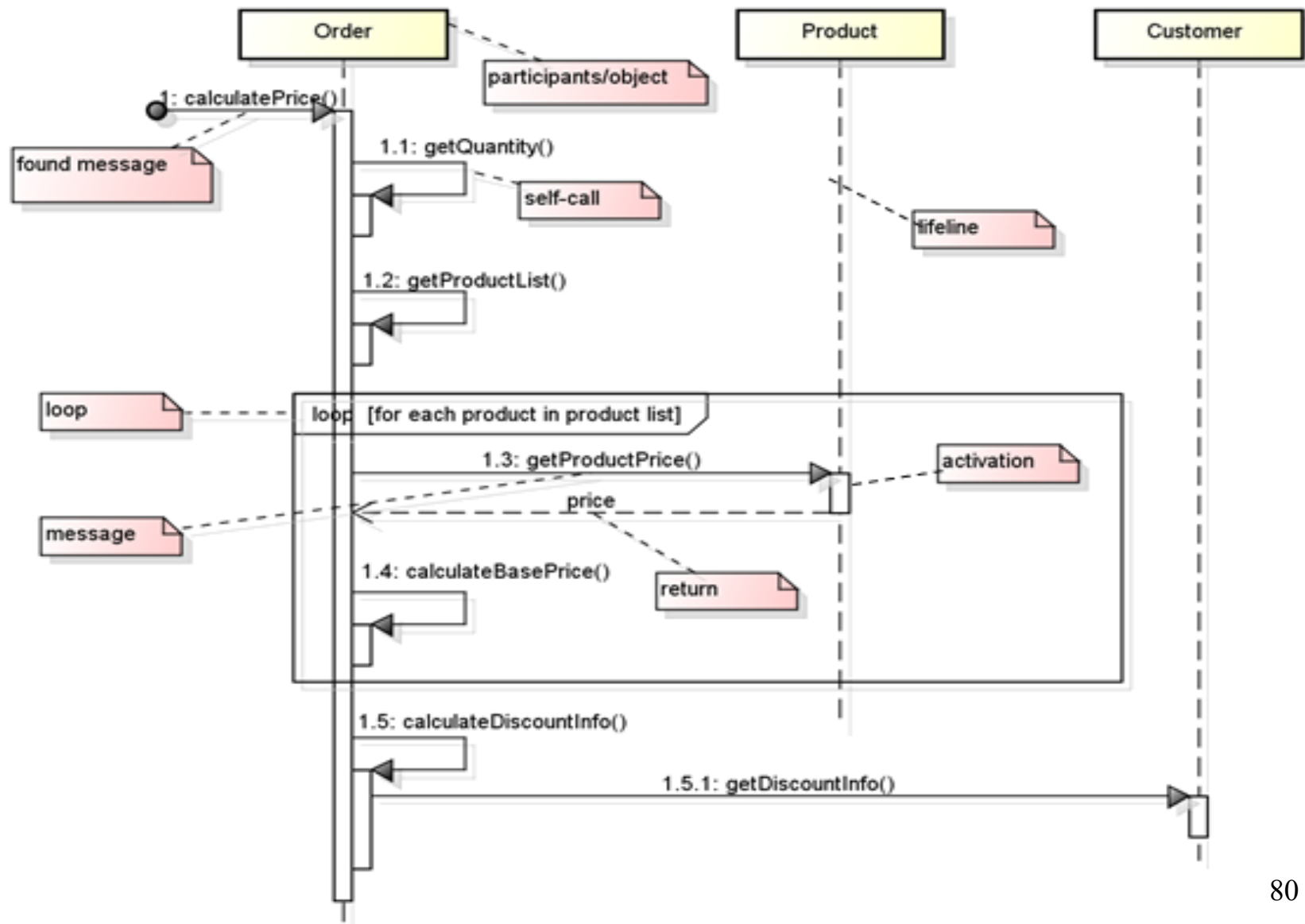
Submit Schedule



# VÍ DỤ SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM



# SƠ ĐỒ TRÌNH TỰ - SEQUENCE DIAGRAM



# Ví dụ: Use case rút tiền của hệ thống ATM

Use case: **Withdraw**

Actor: **Client**

Purpose: **To allow the client to withdraw money**

Pre-conditions: **User** already logged-in

Post-conditions: **Amount** is deducted from user's **account**

Main flow:

- 1) **Client** initiates this usecase by selecting 'withdraw'
- 2) System displays all the **accounts** related to the **Card** and prompts to select any one.
- 3) Client selects one account.
- 4) System prompts for the **amount** (fast **cash** or ...)
- 5) Client enter amount

...

Alternative flows:

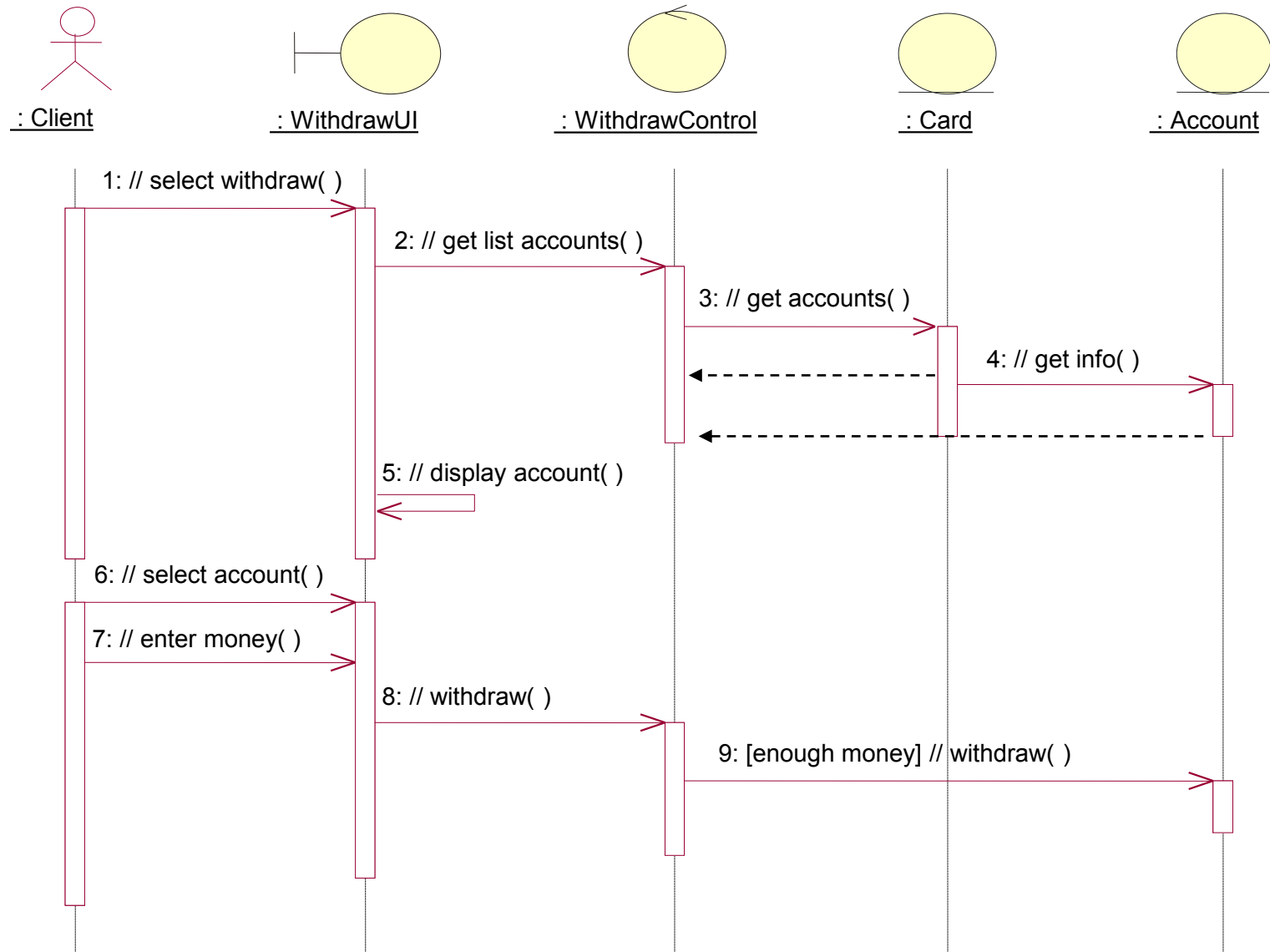
- 2) & 3) System selects the only one available account

Identify Nouns



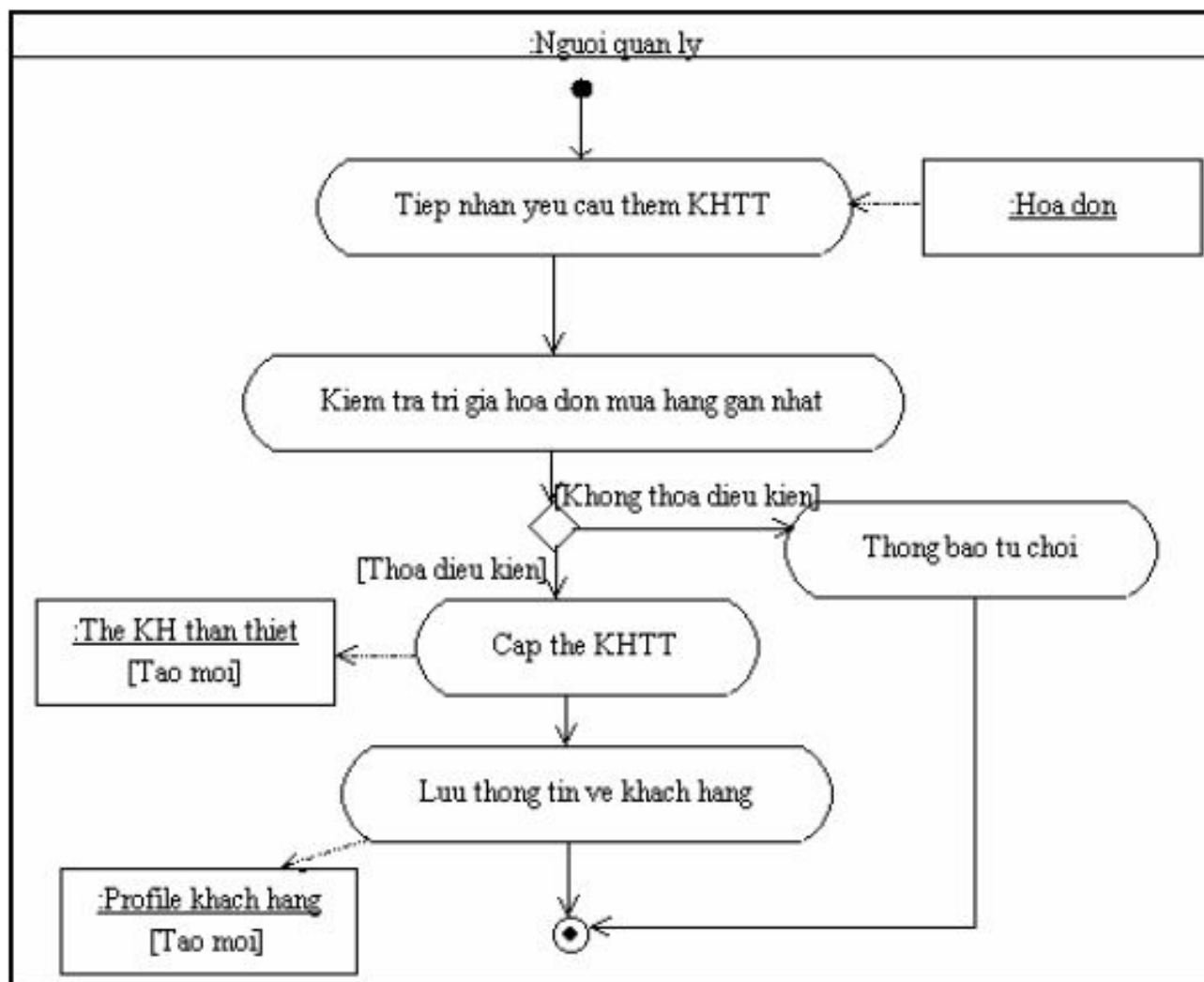
**Client**  
**User**  
**Account**  
**Amount**  
**Money**  
**Card**  
**Cash**

# sơ tuần tự – Use case rút tiền (tt)



# VÍ DỤ ACTIVITY DIAGRAM

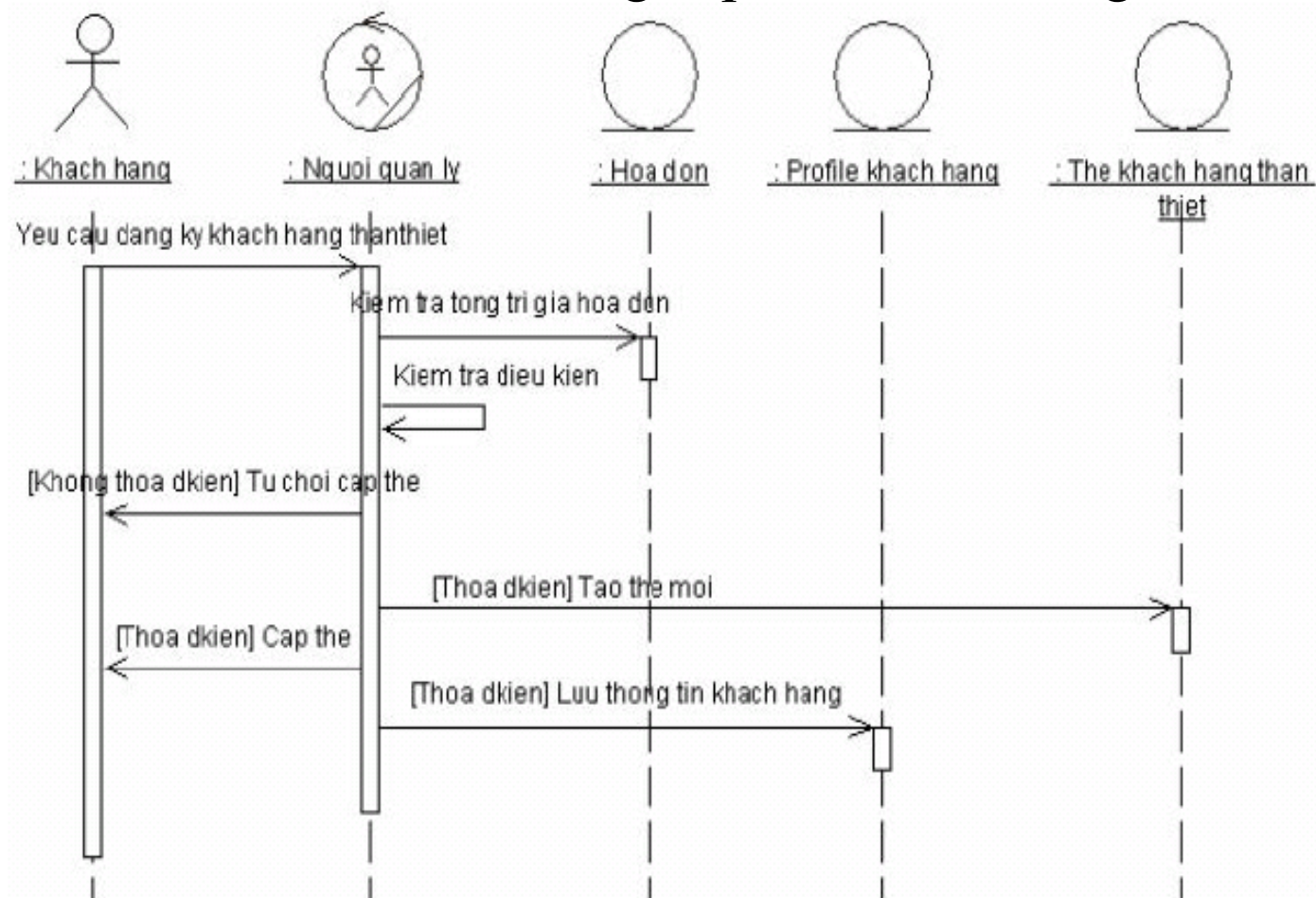
- sơ hoạt động hiện thực hoá use case Quản lý khách hàng thân thiết





# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

Vẽ sơ trình tự cho hoạt động cấp thẻ khách hàng thành viên



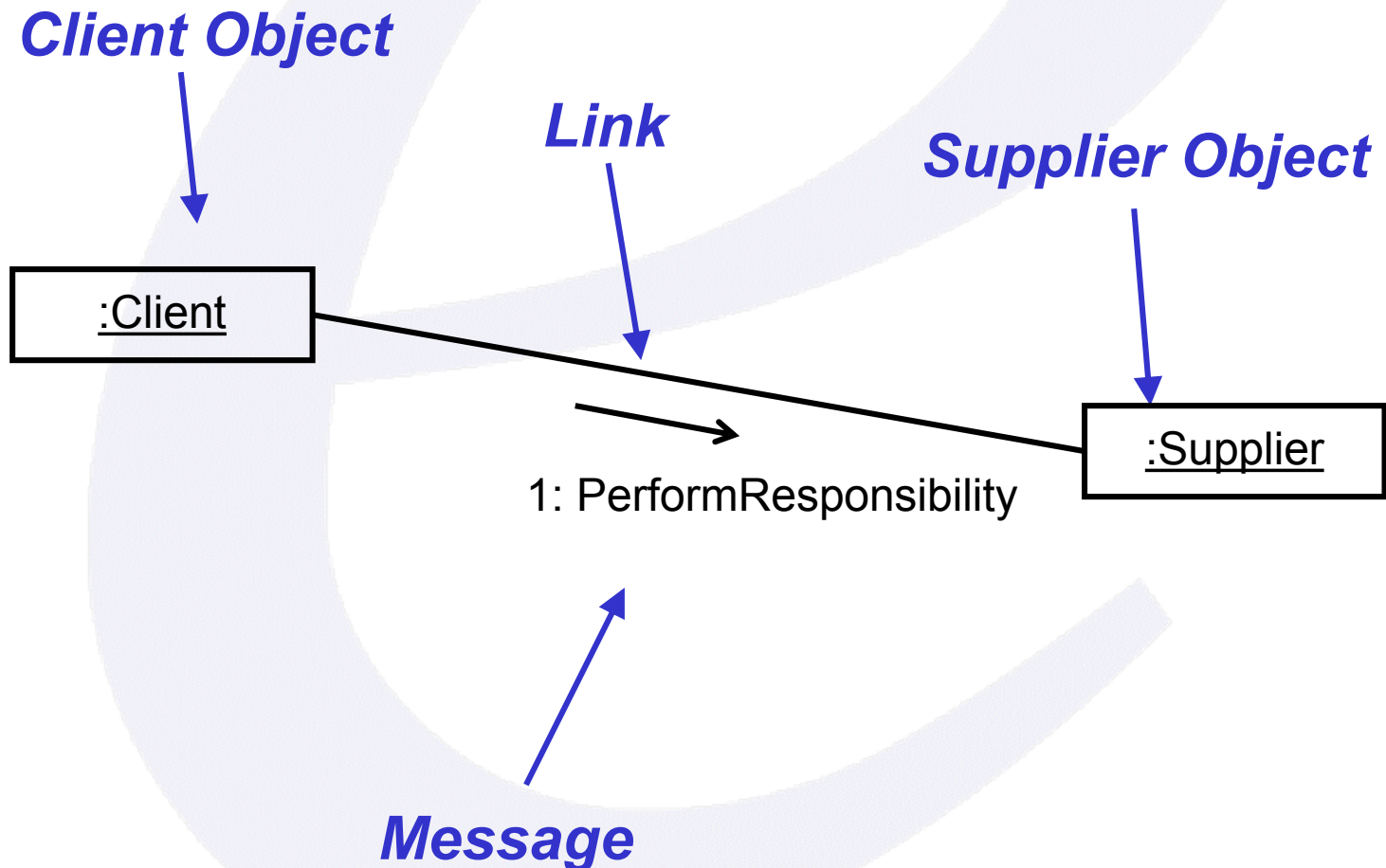
# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

**Sơ đồ cộng tác:** miêu tả giữa actor và các đối tượng hệ thống tương tác với nhau ra sao, vị trí của đối tượng không quan trọng nhưng trọng điểm trong một biểu đồ cộng tác là sự kiện. Tập trung vào sự kiện có nghĩa là chú ý đặc biệt đến mối quan hệ (nối kết) giữa các đối tượng, và vì thế mà phải thể hiện chúng một cách rõ ràng trong biểu đồ.

- sơ đồ cộng tác thường được dùng để biểu diễn một kịch bản khai thác (scenario) của một use case
- Có thể tạo nhiều collaboration diagram cho một usecase.
- Có thể xác định được các lớp đối tượng và mối liên hệ giữa các lớp từ collaboration diagram

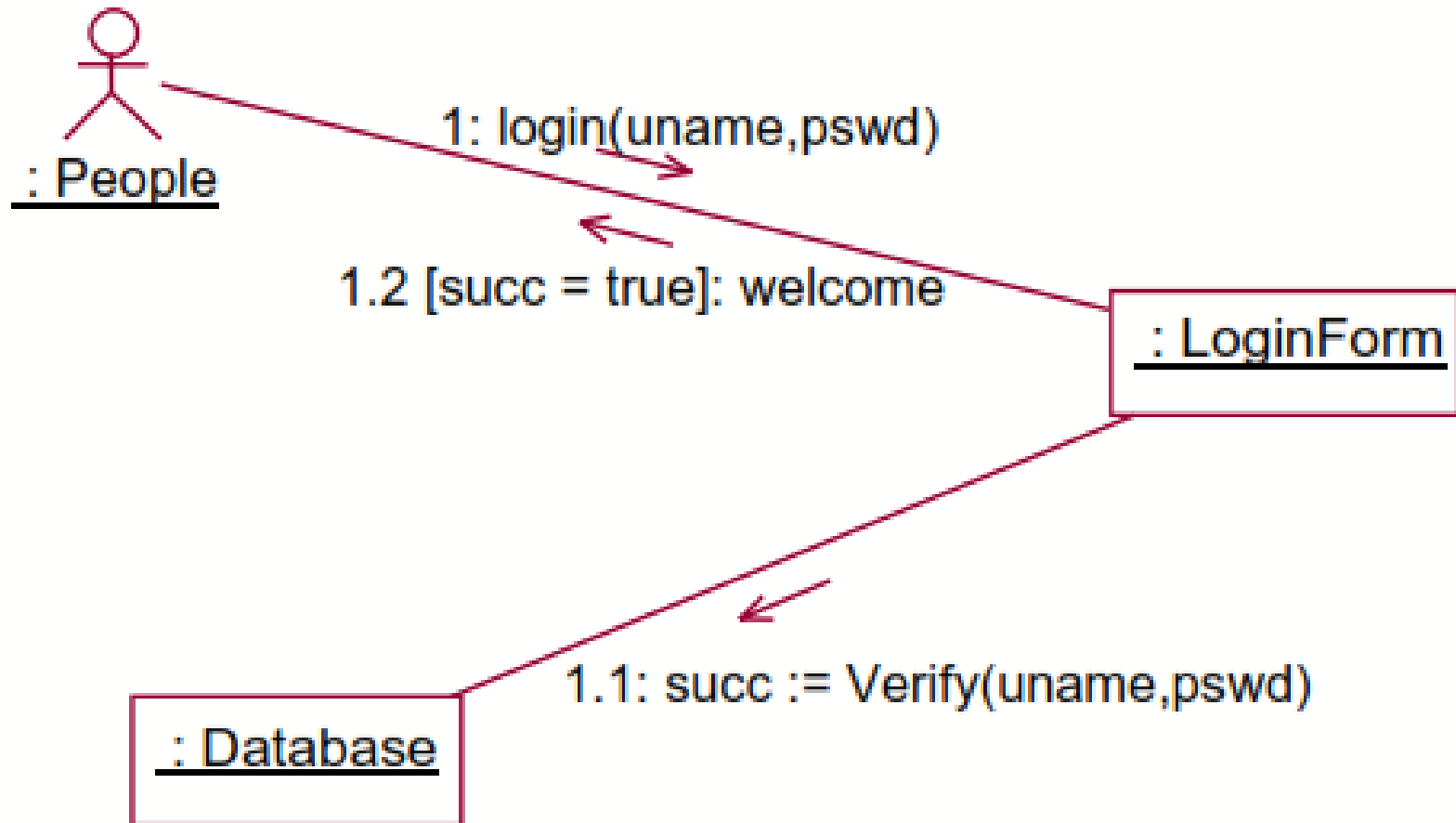
# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

- Các thành phần trong sơ đồ cộng tác:

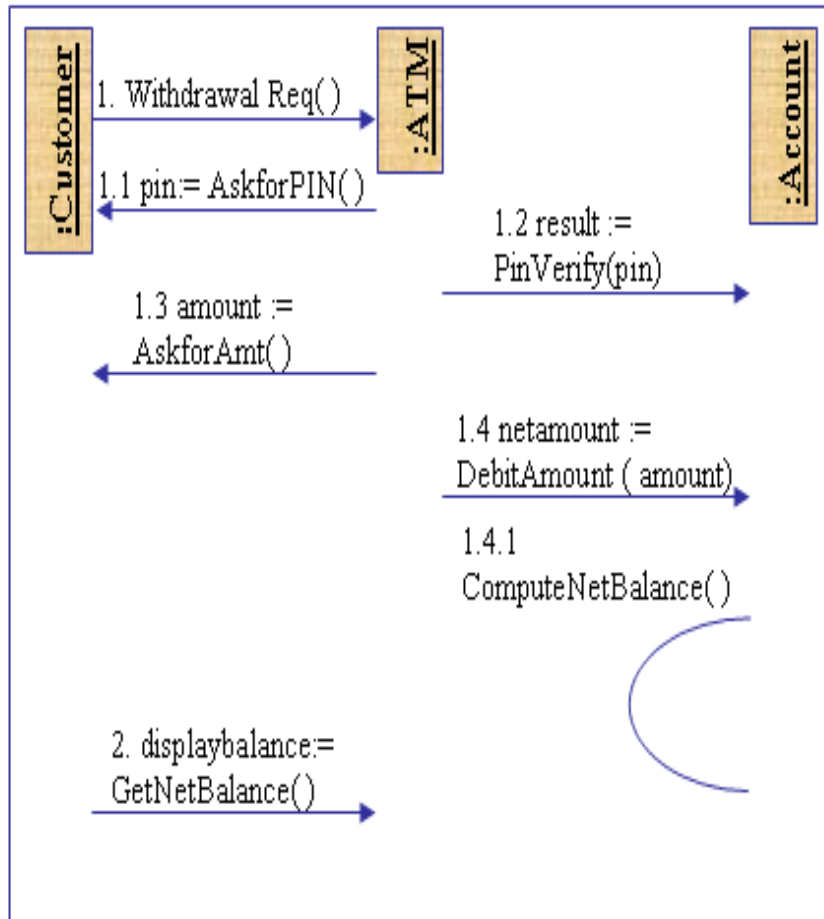


# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

- Ví dụ: sơ đồ cộng tác mức cụ thể cho use case của login của hệ thống đăng ký môn học tín chỉ qua web



# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)



- Đầu tiên thủ tục WithdrawalReq() được gọi từ lớp khách hàng. Đó là lệnh gọi số 1.
- Bước tiếp theo trong tuần tự là hàm AskForPin(), số 1.1, được gọi từ lớp ATM. Thông điệp trong biểu đồ được viết dưới dạng pin:= AskForPin(), thể hiện rằng "giá trị trả về" của hàm này chính là mã số mà lớp khách hàng sẽ cung cấp.
- Hình cung bên lớp tài khoản biểu thị rằng hàm ComputeNetBalance() được gọi trong nội bộ lớp tài khoản và nó xử lý cục bộ. Thường thì nó sẽ là một thủ tục riêng (private) của lớp.

6.5- Một biểu đồ cộng tác của kịch cảnh rút tiền ở máy



# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

---

Các thành phần có trong sơ đồ cộng tác:

- Actor
- Object
- Message
- Instance link

# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

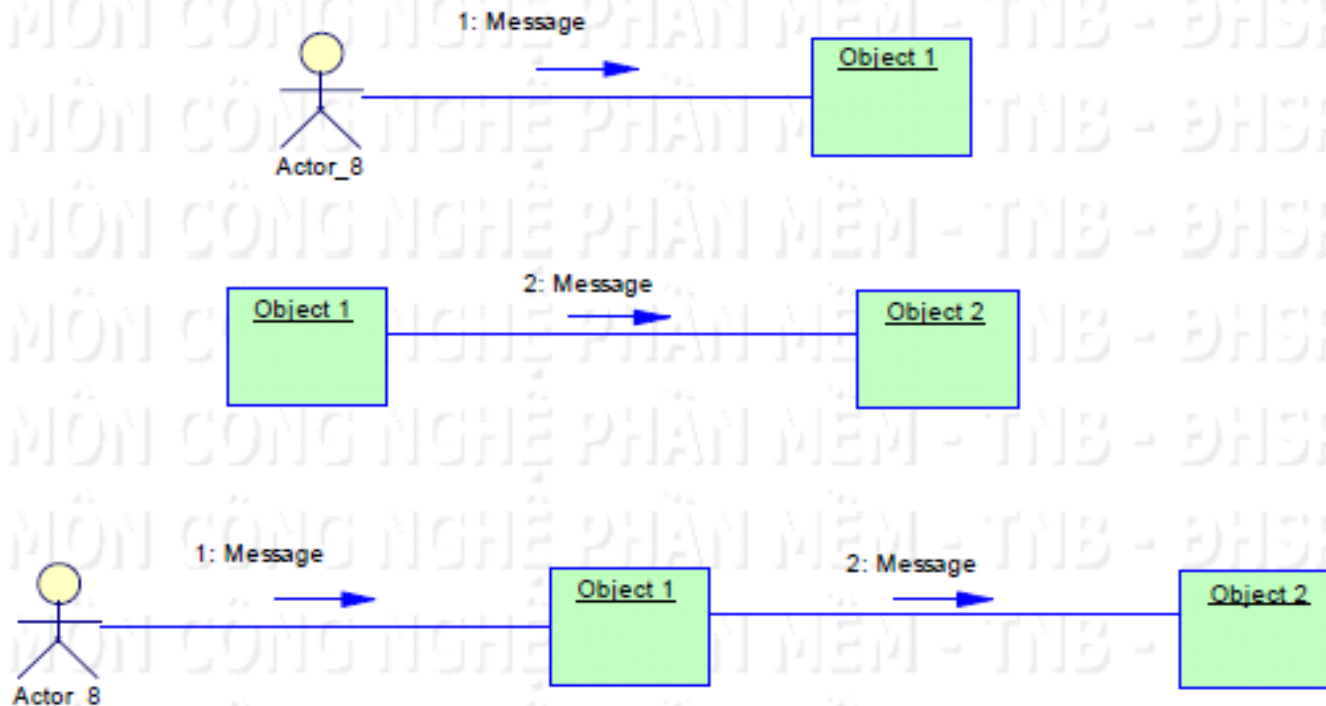
---

- Actor:
  - Tác nhân bên ngoài tương tác với hệ thống
- Object
  - Đối tượng tham gia quá trình tương tác giữa người dùng và hệ thống
- Message
  - Thông điệp mô tả tương tác giữa các đối tượng
  - Thông điệp được gửi từ đối tượng này sang đối tượng khác
  - Thông điệp có thể là 1 yêu cầu thực thi hệ thống, lời gọi hàm khởi tạo đối tượng, hủy đối tượng, cập nhật đối tượng,...

# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

- Message

– Thông điệp được biểu diễn trong Collaboration như sau:



# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

---

Thuộc tính của thông điệp:

- Action
- Control Flow
- Operation
- Arguments
- Return Value
- Predecessor list
- Condition



# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

---

## Thuộc tính của thông điệp:

- Action

- Create: đối tượng gửi thông điệp gọi hàm khởi tạo đối tượng nhận thông điệp
- Destroy: đối tượng gửi thông điệp gọi hàm hủy đối tượng nhận thông điệp
- Self Destroy: đối tượng gửi thông điệp sẽ bị hủy sau khi gửi thông điệp đến đối tượng nhận.



# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

---

Thuộc tính của thông điệp:

- Control Flow
  - Undefined: không được định nghĩa
  - Asynchronous: thông điệp không đồng bộ, đối tượng gửi thông điệp không cần đợi kết quả trả về từ đối tượng nhận, các thông điệp có thể thực hiện đồng thời
  - Procedure Call: thông điệp đồng bộ, đối tượng gửi thông điệp phải đợi kết quả trả về từ đối tượng nhận
  - Return: thông điệp return thường liên kết với thông điệp loại "Procedure call"

# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

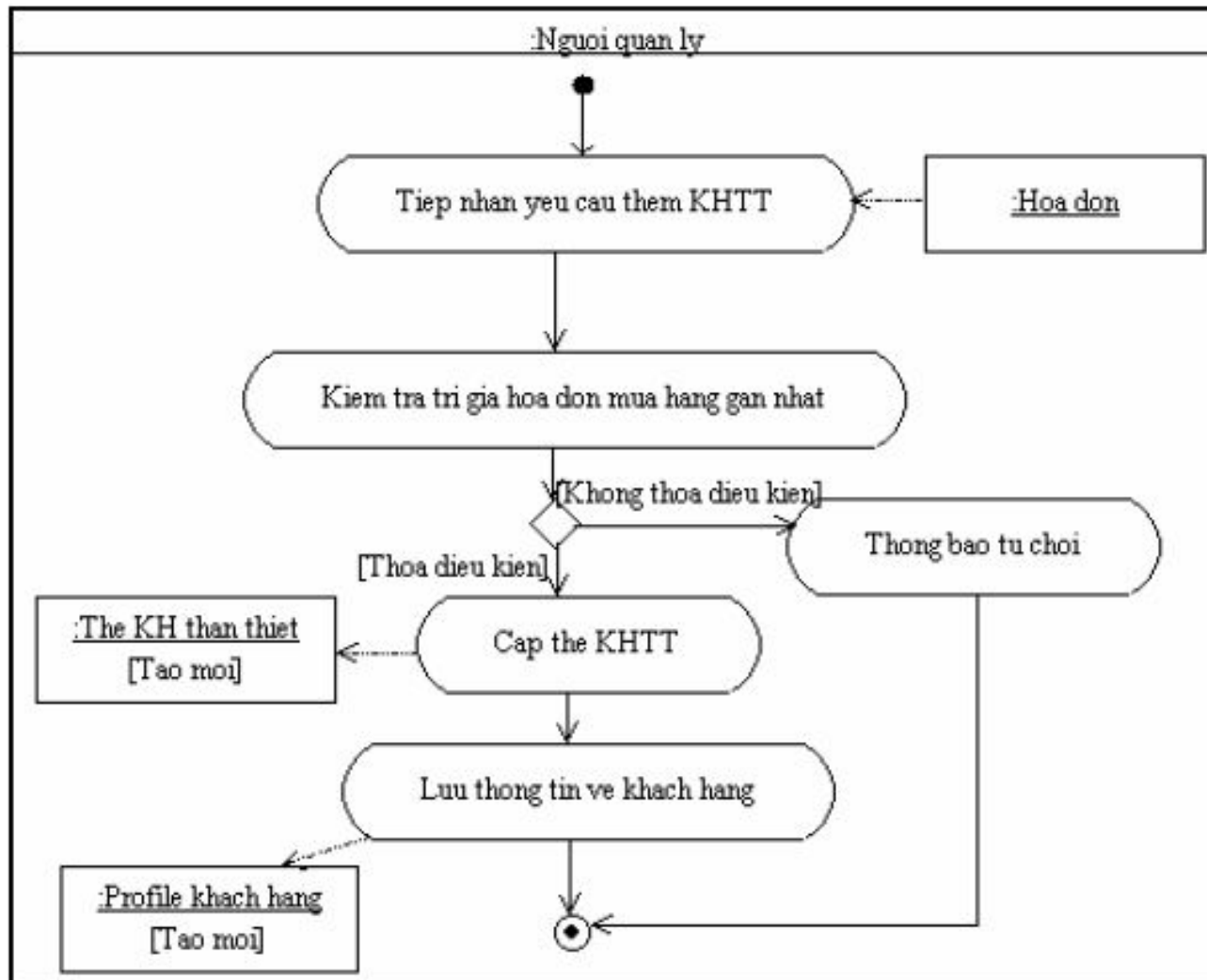
---

Thuộc tính của thông điệp:

- Operation
  - Nếu đối tượng trong sơ đồ là một thể hiện của một lớp đối tượng (class), chúng ta có thể chọn operation của class để liên kết message
  - Không thể liên kết return message với operation
- Arguments
- Return Value
- Predecessor list
- Condition

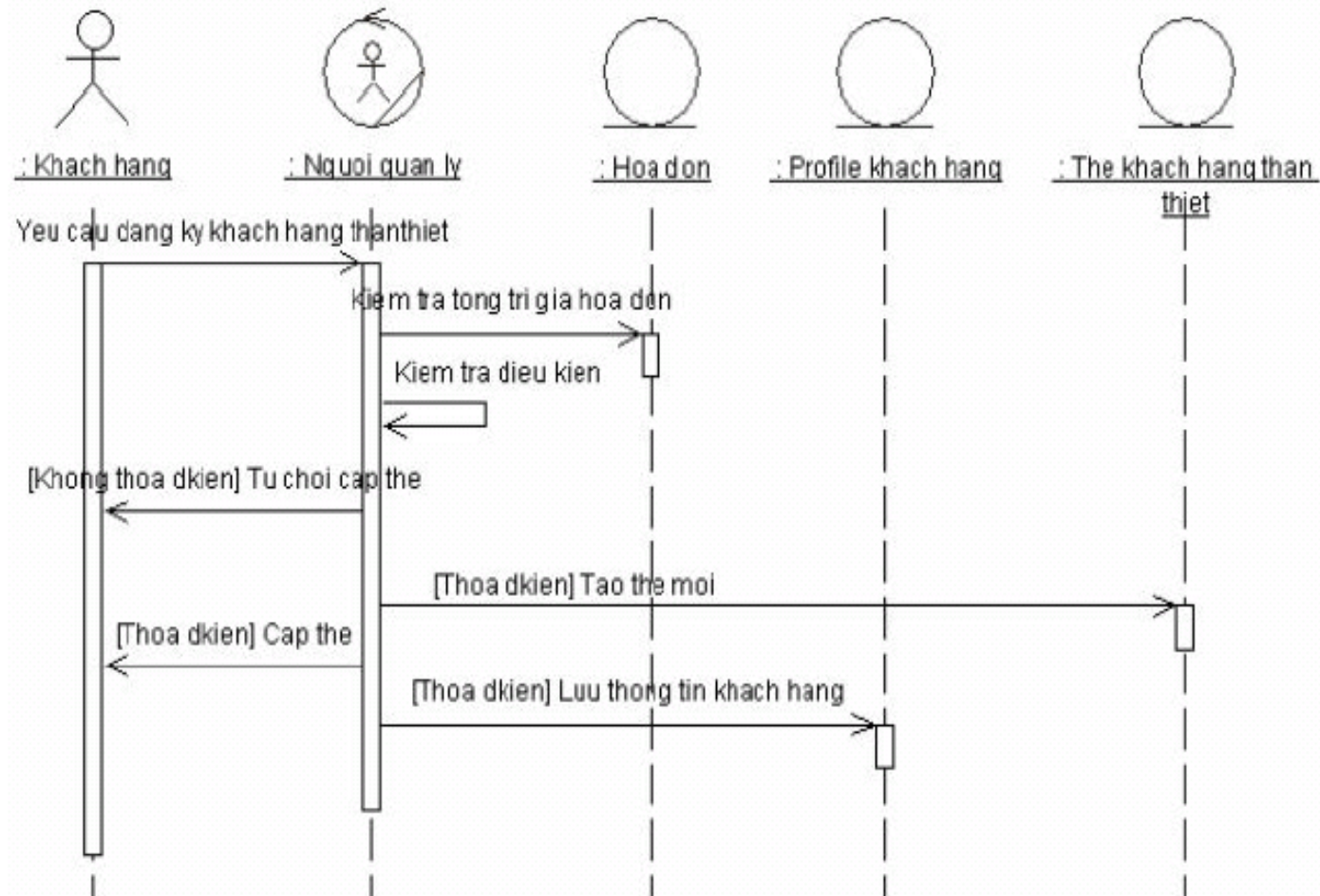
# VÍ DỤ ACTIVITY DIAGRAM

- sơ hoạt động hiện thực hoá use case Quản lý khách hàng thân thiết



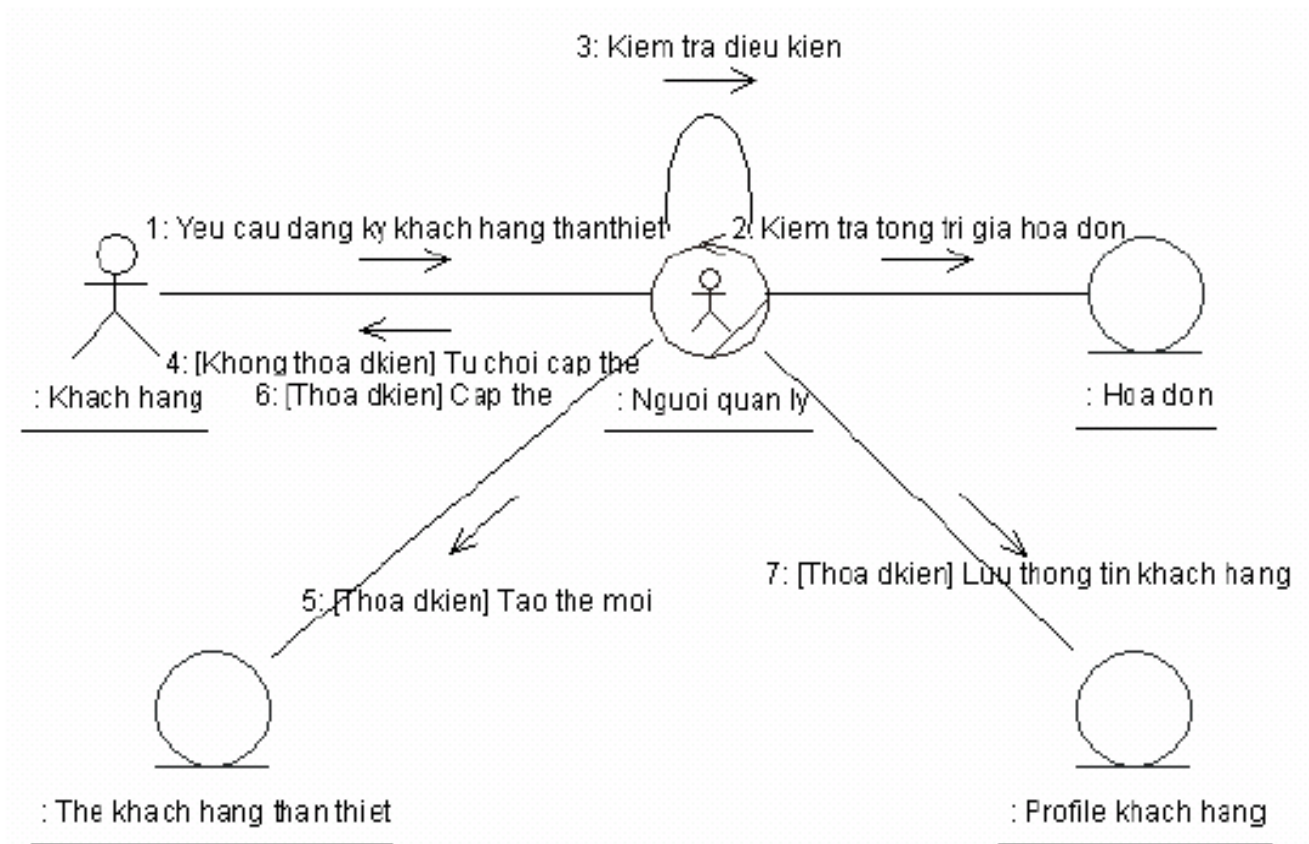
# SƠ ĐỒ TRÌNH TỰ - SEQUENCY DIAGRAM

Vẽ sơ trình tự cho hoạt động cấp thẻ khách hàng thành viên



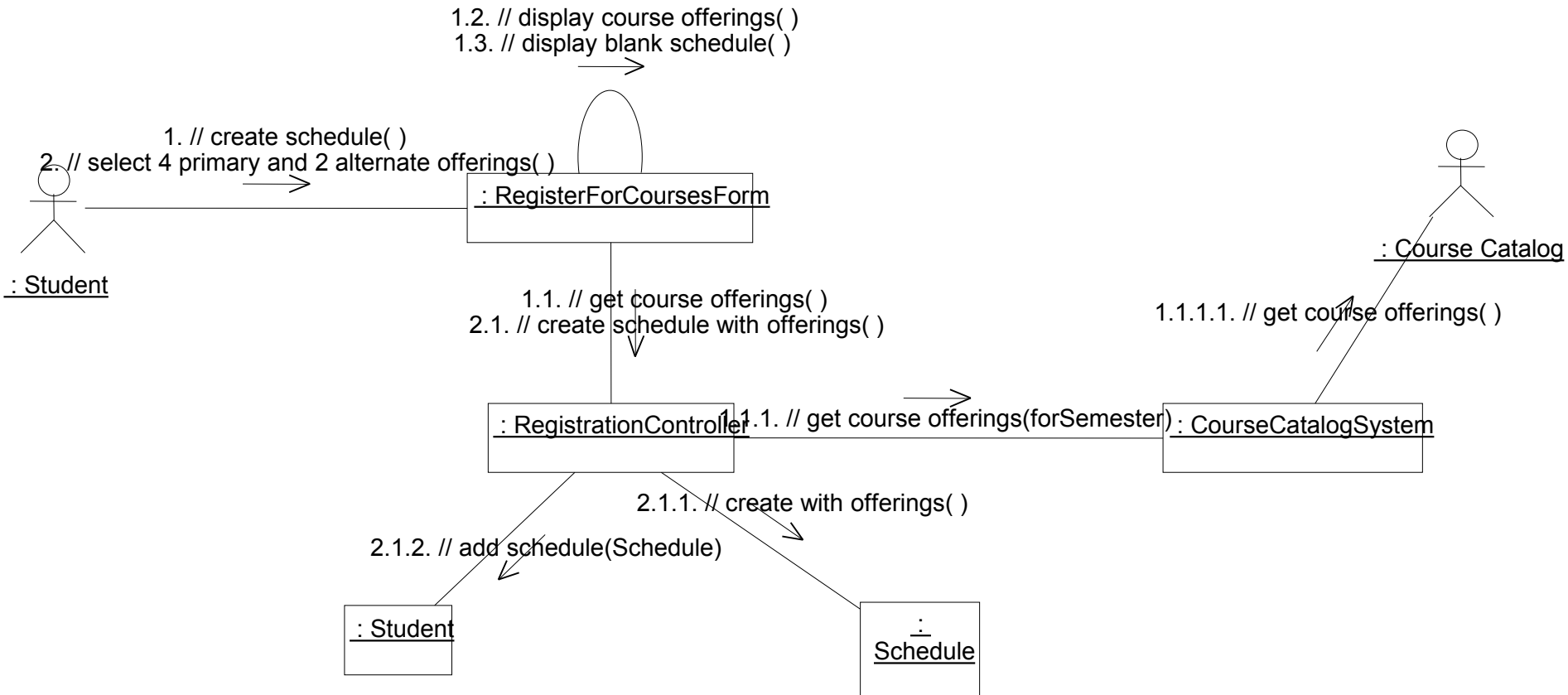
# SƠ ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

- Ví dụ: sơ đồ cộng tác tạo thẻ thành viên

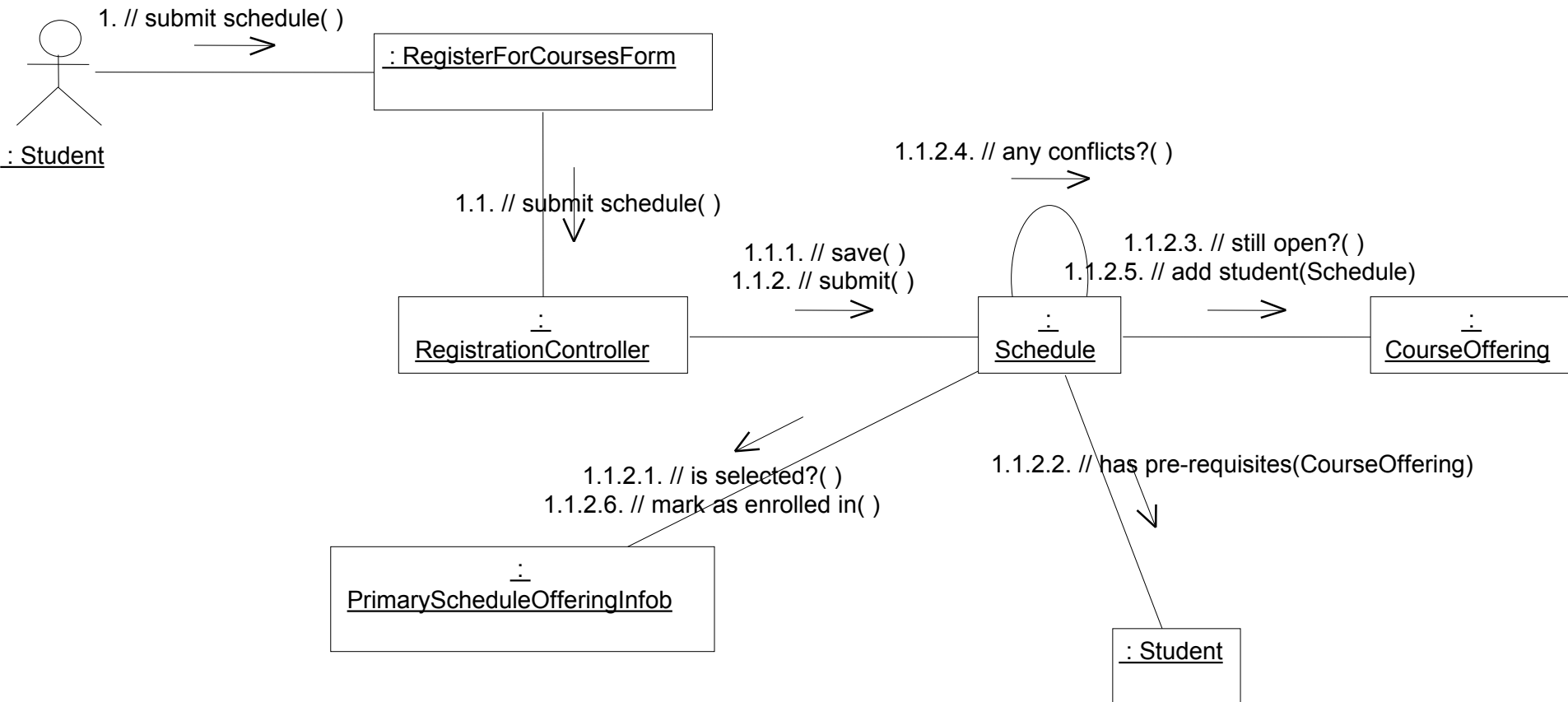




# Ví dụ: sơ cộng tác Đăng ký học phần



# Ví dụ: sơ cộng tác đăng ký học phần(tt)



# SO SÁNH 2 LƯỢC ĐỒ

Loại	Ưu điểm	Nhược điểm
Sequence	<ul style="list-style-type: none"><li>- Chỉ ra thứ tự thời gian của các message</li><li>- Ký hiệu đơn giản</li></ul>	Chỉ mở rộng lược đồ về bên phải khi bổ sung thêm đối tượng mới, tốn nhiều không gian theo chiều rộng
Collaboration	<ul style="list-style-type: none"><li>-Tiết kiệm không gian, dễ dàng thêm đối tượng mới vào theo cả 2 chiều</li><li>-Biểu diễn các hành vi phân nhánh, lặp lại, đồng thời rõ ràng hơn</li></ul>	<ul style="list-style-type: none"><li>-Khó thấy được thứ tự các thông báo</li><li>-Ký hiệu phức tạp hơn</li></ul>

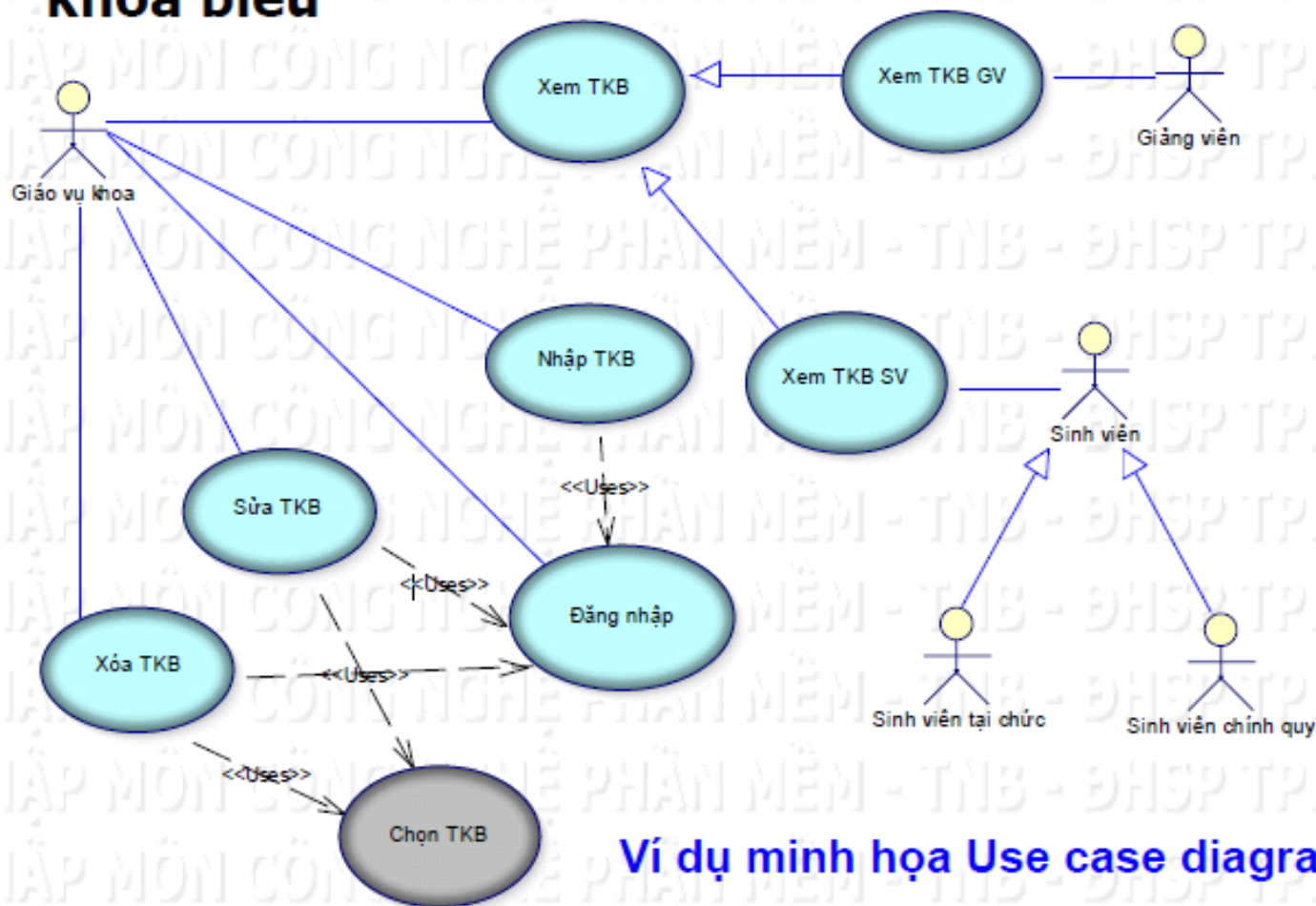
**Lược đồ tuần tự thông dụng hơn**

# **Bài tập: BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)**

1. Xây dựng sơ cộng tác cho Đặt vé tàu, sinh viên
2. Xây dựng sơ activity, sequence và Collaboration diagram của ứng dụng quản lý TKB:
  - SV chọn chức năng xem TKB lớp
  - Hệ thống hiển thị màn hình xem TKB lớp
  - SV chọn lớp, niên khóa, học kỳ
  - SV chọn chức năng xem TKB
  - Hệ thống truy cập CSDL lấy thông tin TKB tương ứng với lớp SV chọn
  - Hệ thống hiển thị thông tin TKB

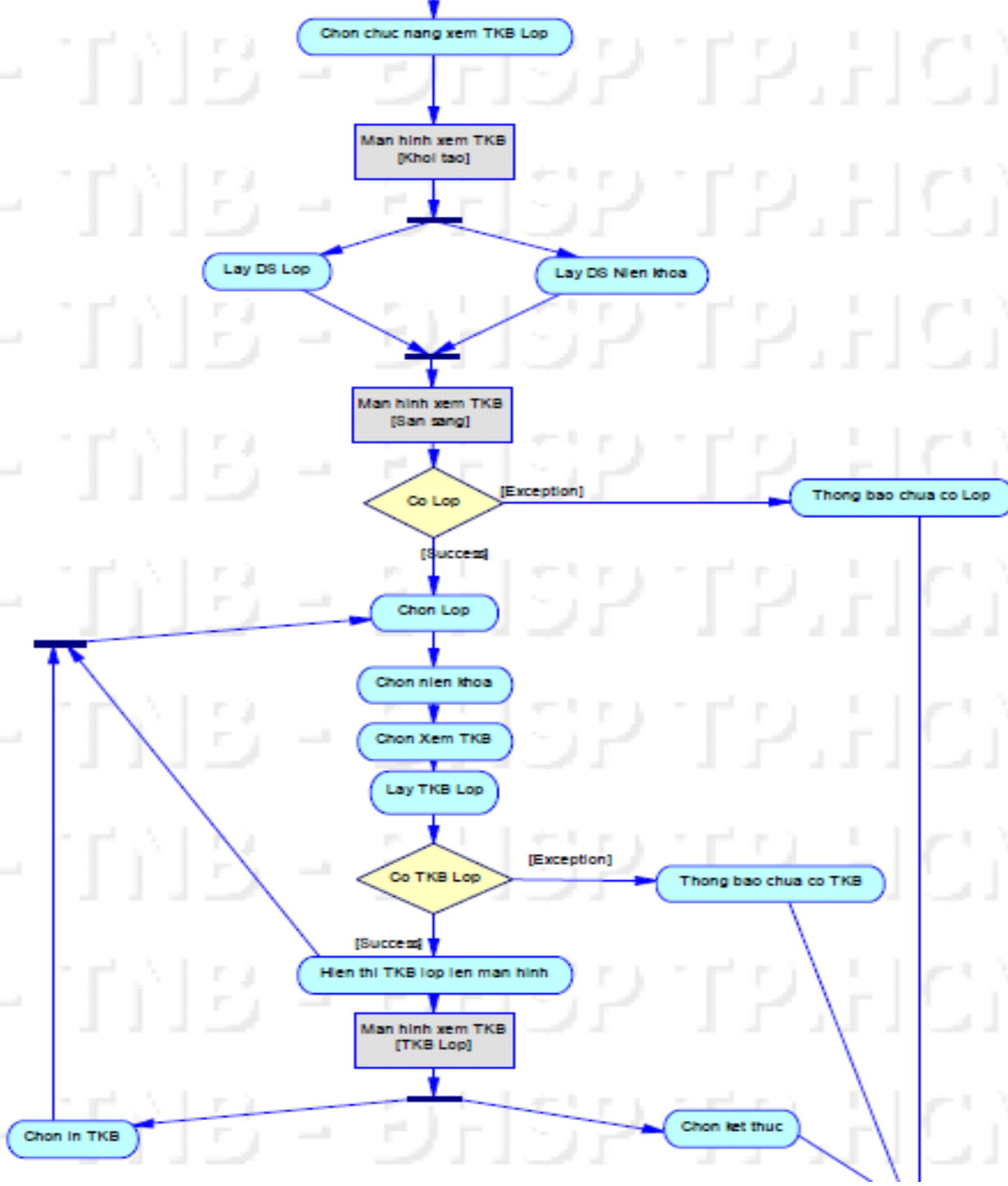
# VÍ DỤ ACTIVITY DIAGRAM

- **Use case diagrams** mô tả hệ thống quản lý thời khóa biểu



Ví dụ minh họa Use case diagram



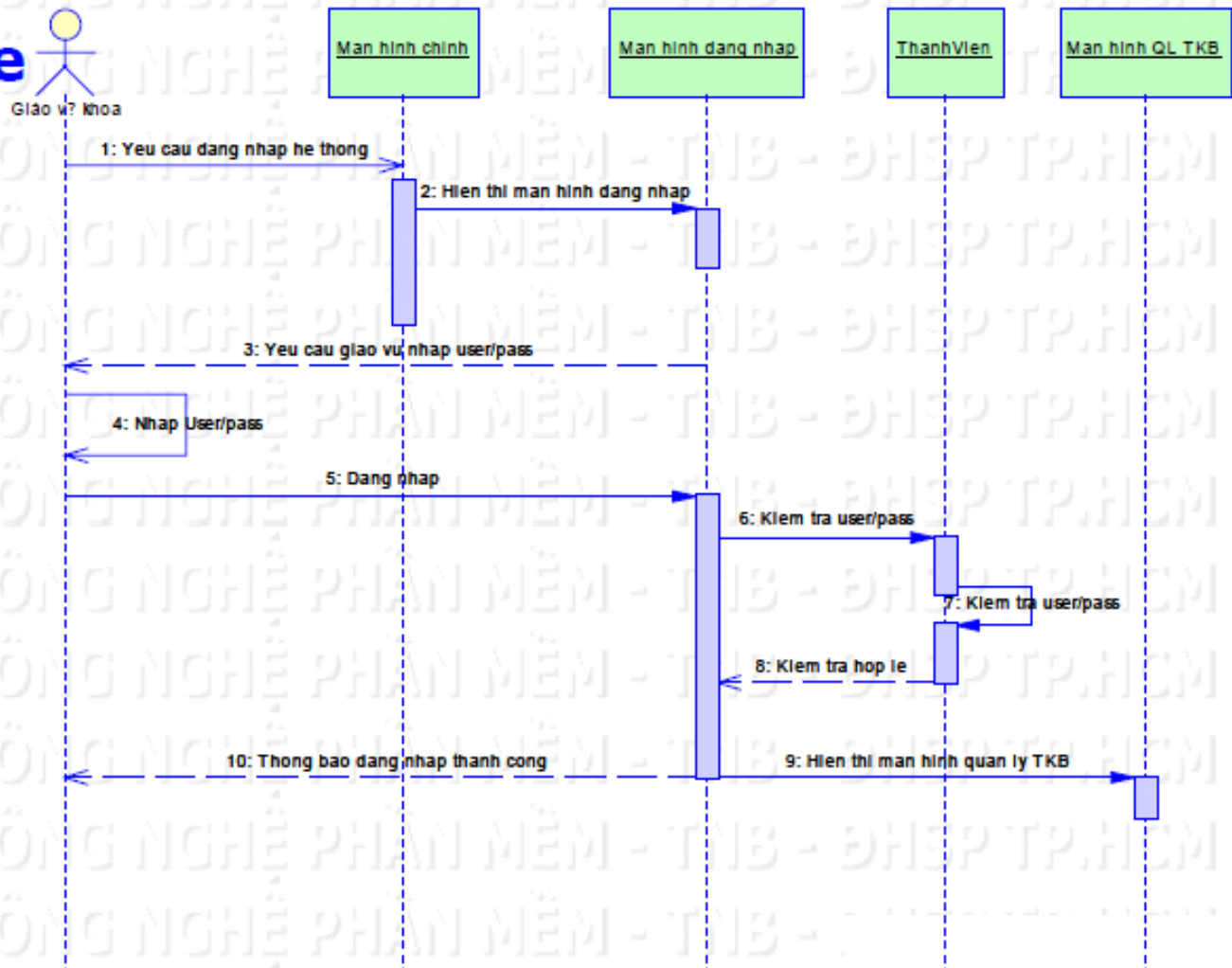


# BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

## Sequence diagram

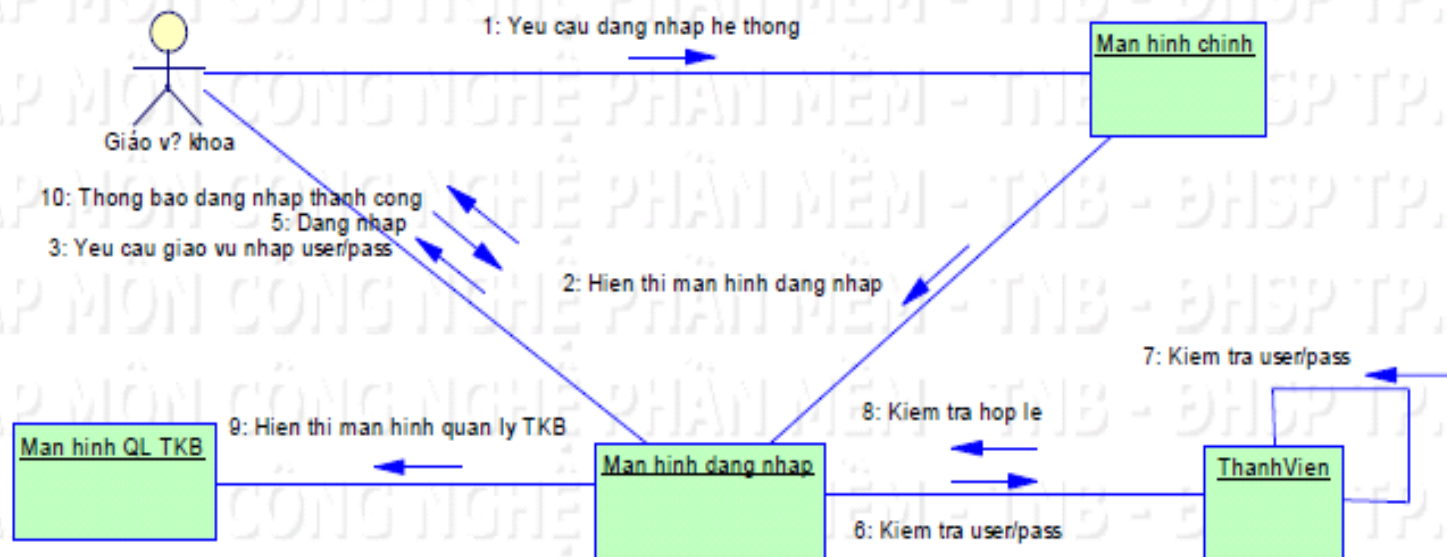
mô tả  
scenario

đăng  
nhập hệ  
thống  
thành  
công



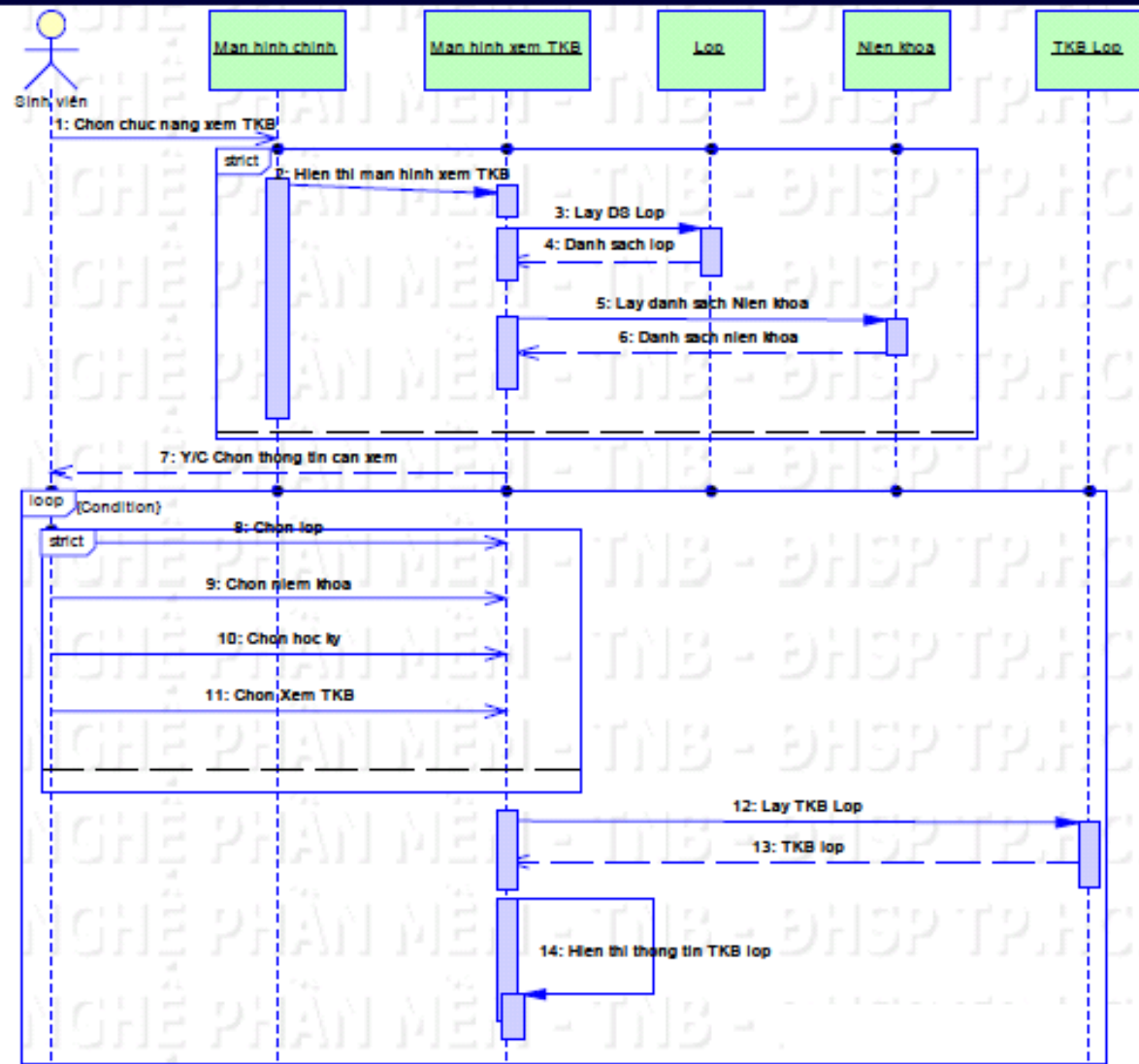
# BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

- Collaboration diagram mô tả scenario đăng nhập hệ thống thành công



# BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

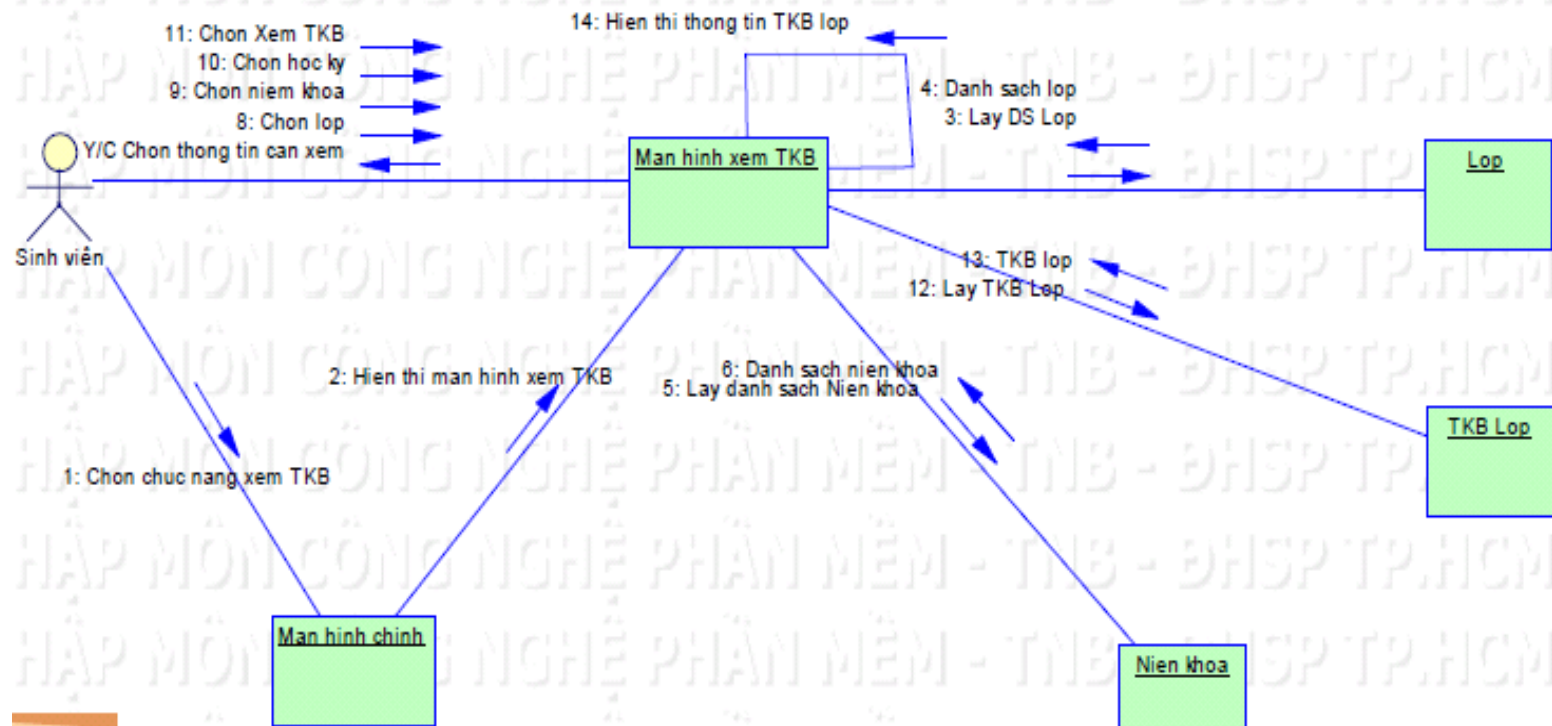
**Sequence  
diagram**  
mô tả  
use-case  
xem TKB  
lớp





# BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

- **Collaboration diagram** mô tả use-case xem TKB lớp hệ thống quản lý thời khóa biểu



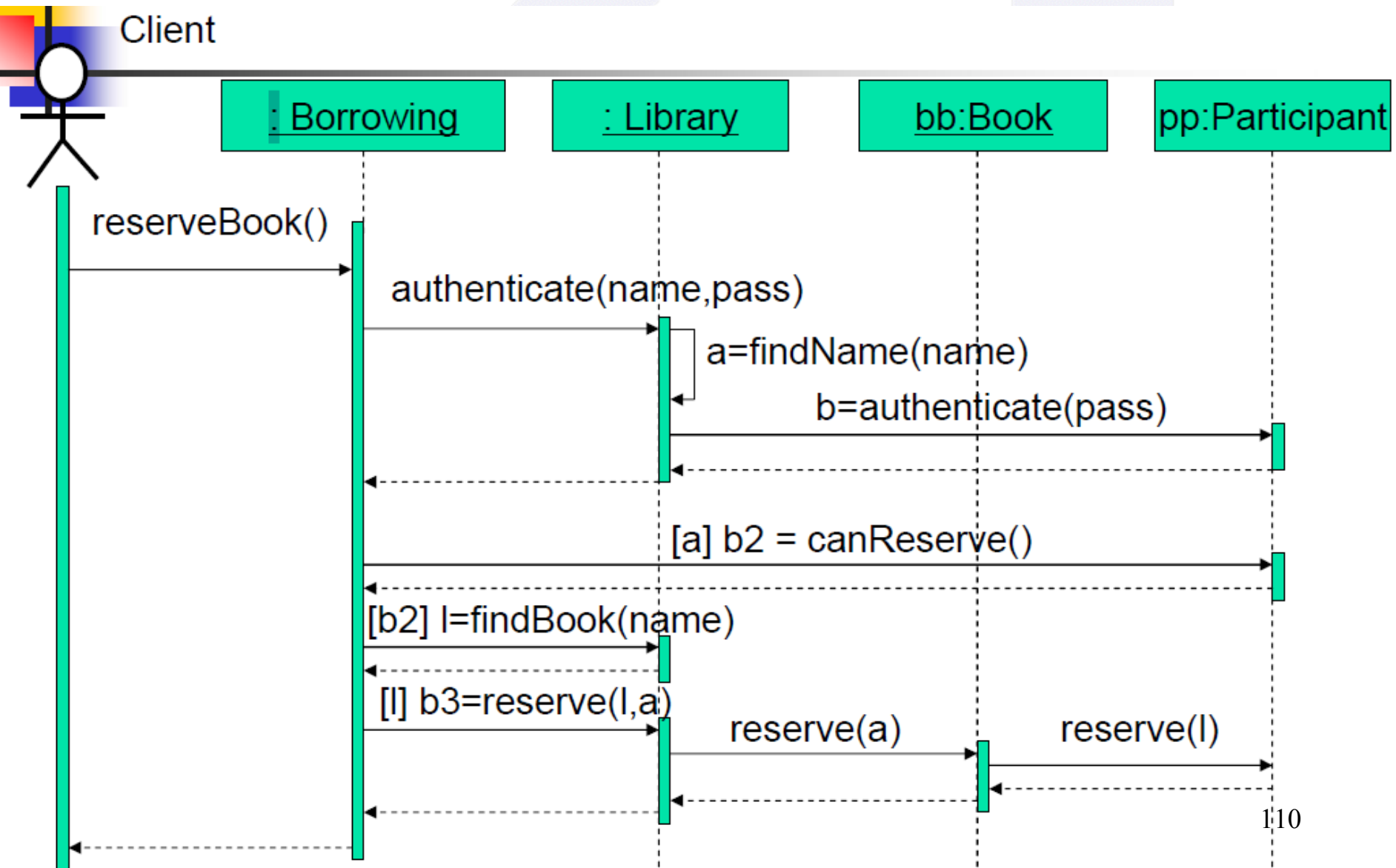


# BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

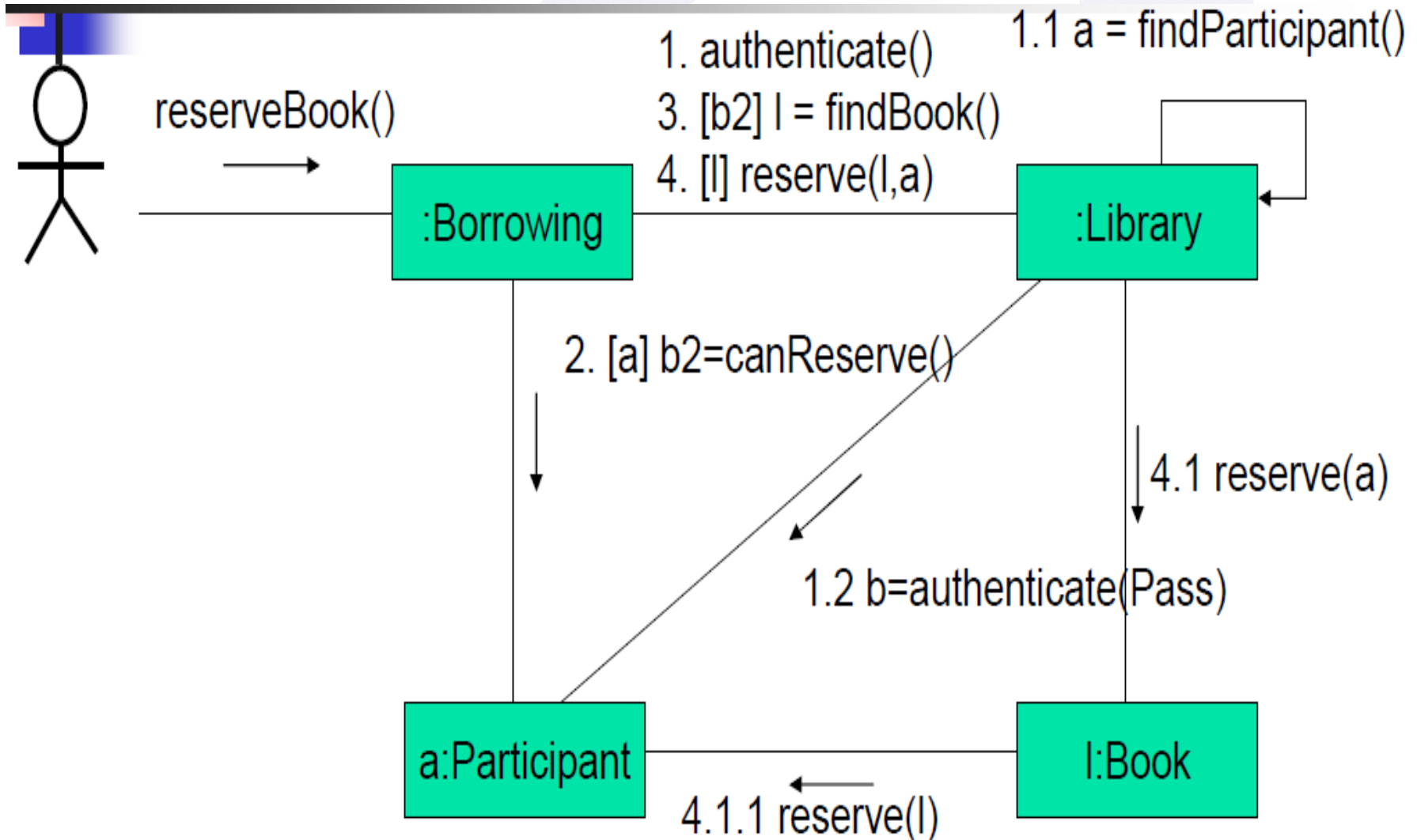
---

Bài tập: Vẽ biểu đồ trình tự và cộng tác cho qui trình đặt sách

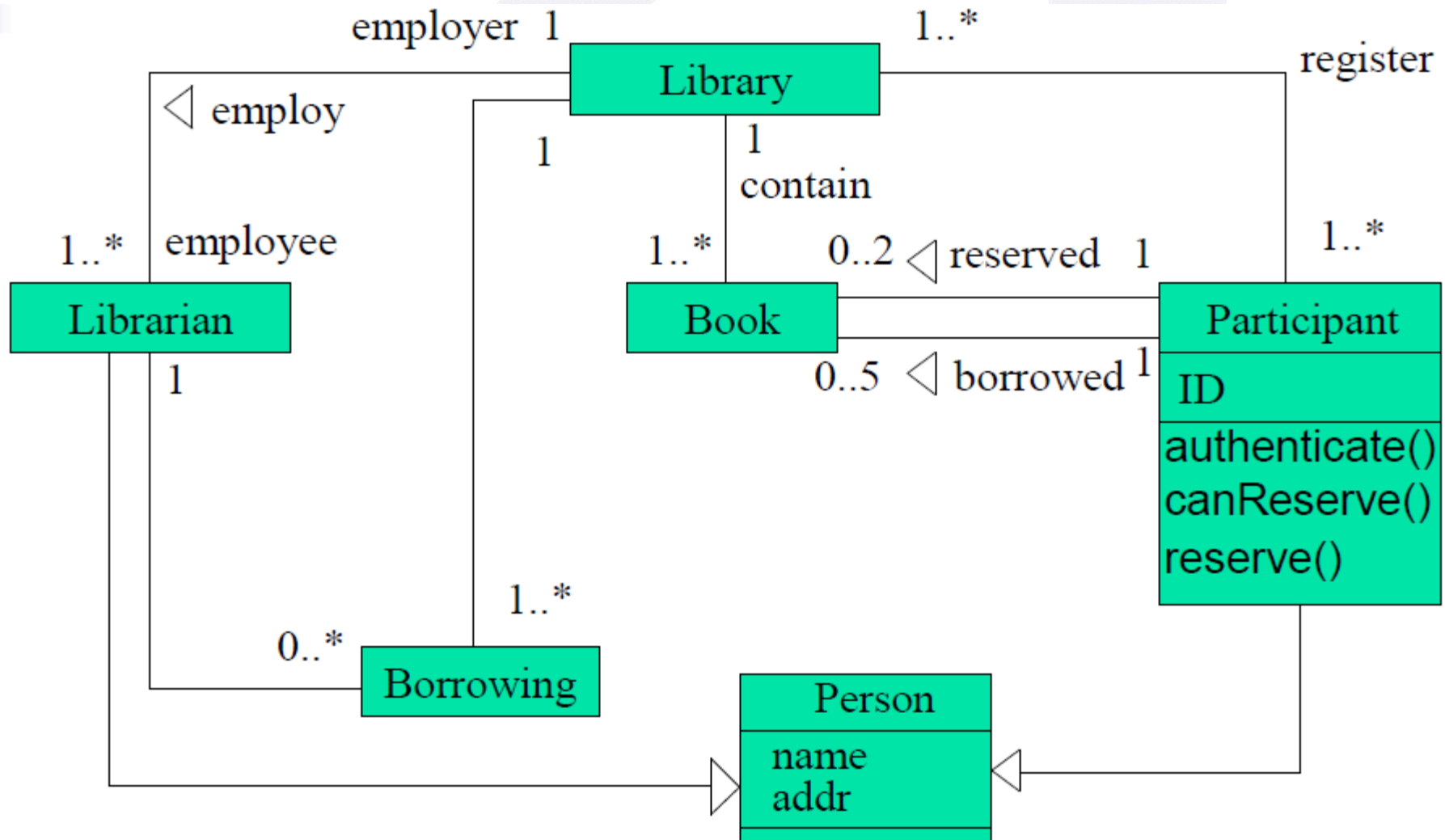
# Biểu đồ tuần tự: Đặt trước sách



# Biểu đồ cộng tác: Đặt sách



# Thêm các phương thức vào các lớp



# BIỂU ĐỒ CỘNG TÁC (COLLABORATION DIAGRAM)

---

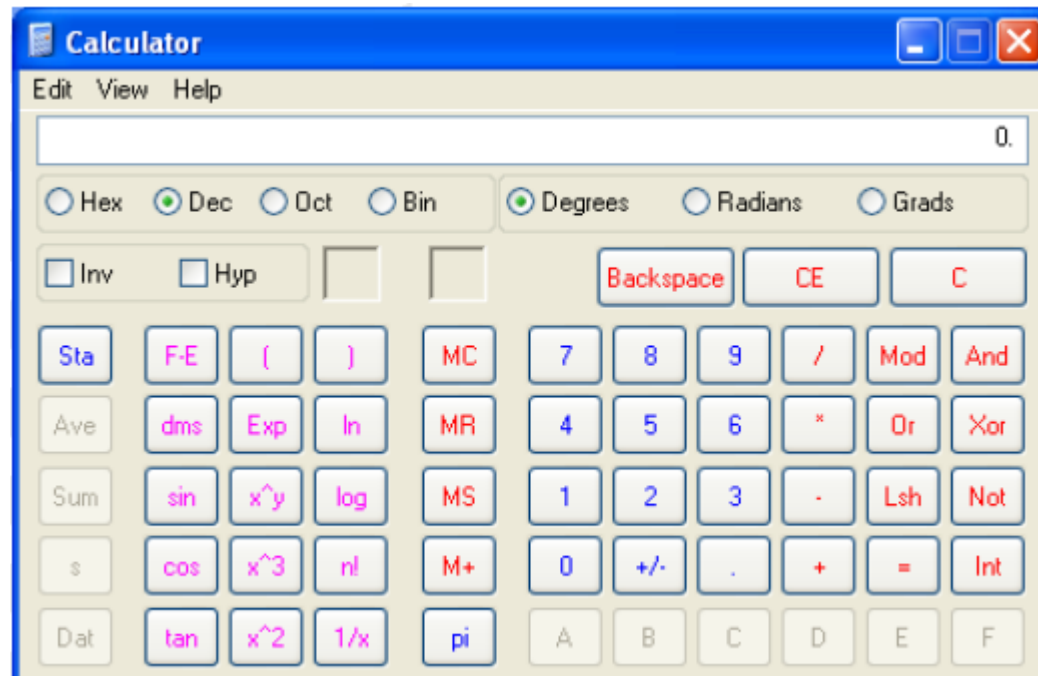
Bài tập:

1. Vẽ biểu đồ trình tự và cộng tác cho qui trình đặt sách
2. Vẽ biểu đồ tuần tự của kịch bản in một file ra máy in
3. Chuyển biểu đồ tuần tự trên sang biểu đồ tương tác



# SƠ ĐỒ TRẠNG THÁI

- sơ trạng thái (State Diagram) biểu diễn mối liên hệ giữa các trạng thái của đối tượng
- Thông thường sơ đồ trạng thái áp dụng cho đối tượng/ lớp → biểu diễn hành vi của lớp
- Thông thường mỗi đối tượng nằm ở một trạng thái trong một khoảng thời gian nhất định → nó sẽ dịch chuyển từ trạng thái này sang trạng thái khác

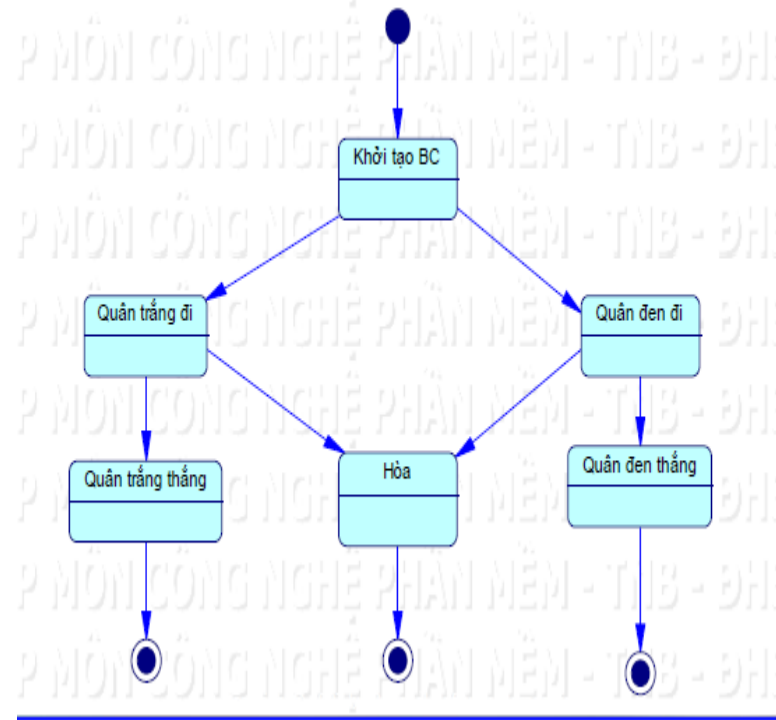


# SƠ ĐỒ TRẠNG THÁI

- Một lớp có thể có một thuộc tính đặc biệt xác định trạng thái, hoặc trạng thái cũng có thể được xác định qua giá trị của các thuộc tính “bình thường” trong đối tượng.

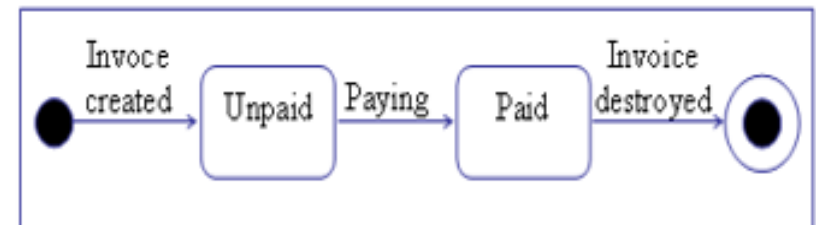
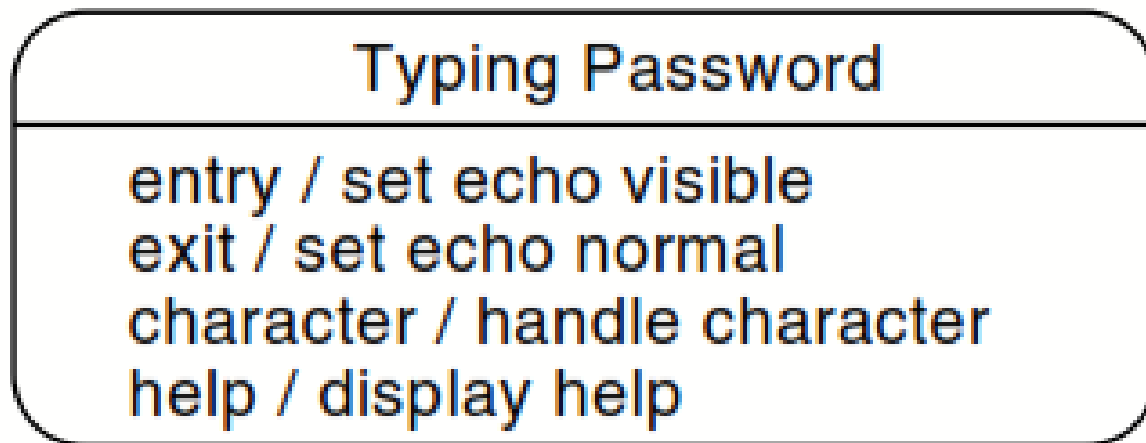
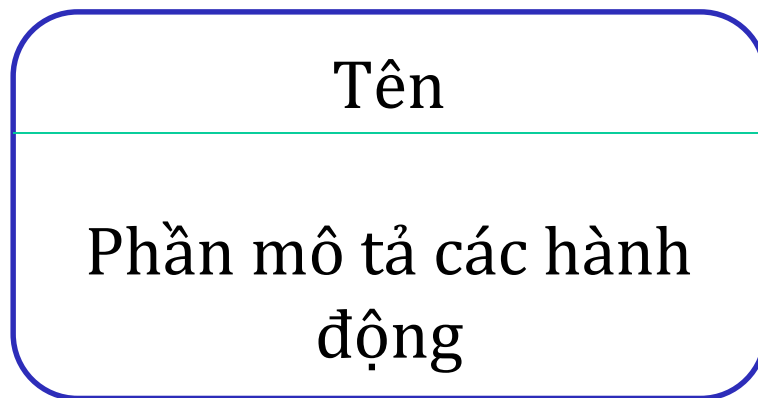
Ví dụ về các trạng thái của đối tượng:

- Hóa đơn (đối tượng) đã được trả tiền (trạng thái).
- Chiếc xe ô tô (đối tượng) đang đứng yên (trạng thái).
- Động cơ (đối tượng) đang chạy (trạng thái).
- Jen (đối tượng) đang đóng vai trò người bán hàng (trạng thái).
- Kate (đối tượng) đã lấy chồng (trạng thái).

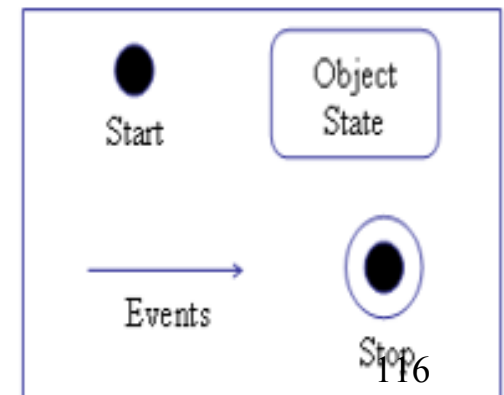


# SƠ ĐỒ TRẠNG THÁI

- Trạng thái tổng hợp là trạng thái có thể được phân rã về các trạng thái đơn giản
- Ký hiệu



Hình 6.7- Biểu đồ trạng thái thực hiện hoá đơn.





# CÁC THÀNH PHẦN CỦA SƠ ĐỒ TRẠNG THÁI

- **Trạng thái - State**
- **Sự kiện – Event**
- **Hành động – Action**
- **Mối liên hệ giữa các trạng thái**

# SƠ ĐỒ TRẠNG THÁI

- **Trạng thái – State**

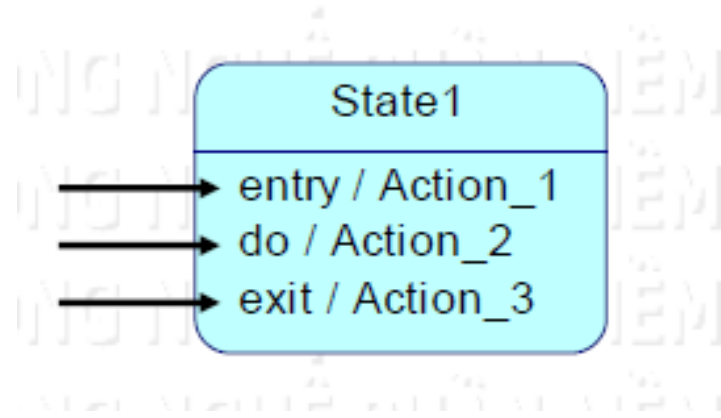


- Trạng thái bắt đầu: khi đối tượng được tạo ra hoặc trạng thái tổng hợp được xác định. Ký hiệu: 
- Trạng thái kết thúc: khi đối tượng bị hủy bỏ hoặc trạng thái tổng hợp trở nên không xác định. 
- **Trạng thái trung gian**



# SƠ ĐỒ TRẠNG THÁI

- **Sự kiện – Event**

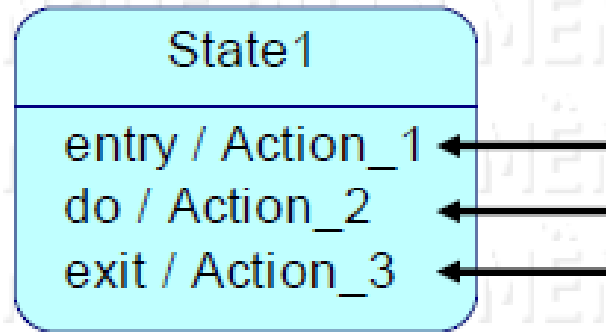


## Event

- **Entry:** sự kiện phát sinh khi đối tượng bắt đầu nhận trạng thái
- **Exit:** sự kiện phát sinh khi đối tượng kết thúc trạng thái
- **Do:** sự kiện phát sinh khi user thực hiện một hành động thông qua bàn phím/chuột.

# SƠ ĐỒ TRẠNG THÁI

- **Trạng thái - State**

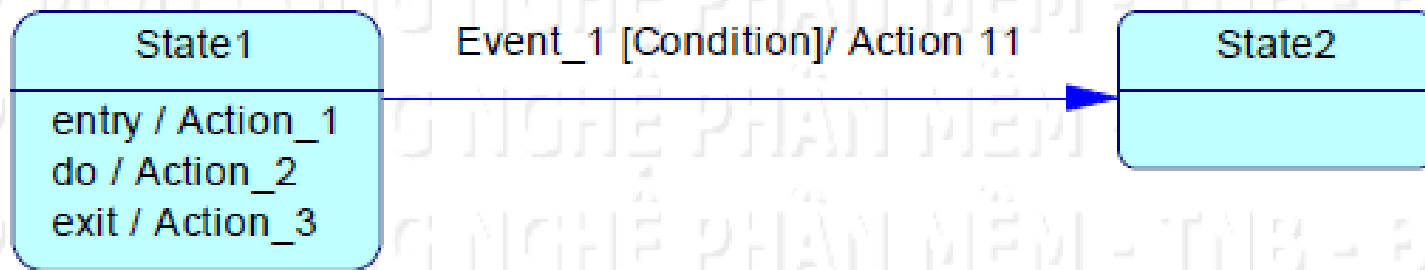


## Action

- **Entry:** hành động được thực hiện khi đối tượng bắt đầu trạng thái
- **Do:** tập các hành động có thể thực hiện với trạng thái
- **Exit:** hành động được thực hiện khi đối tượng kết thúc trạng thái

# SƠ ĐỒ TRẠNG THÁI

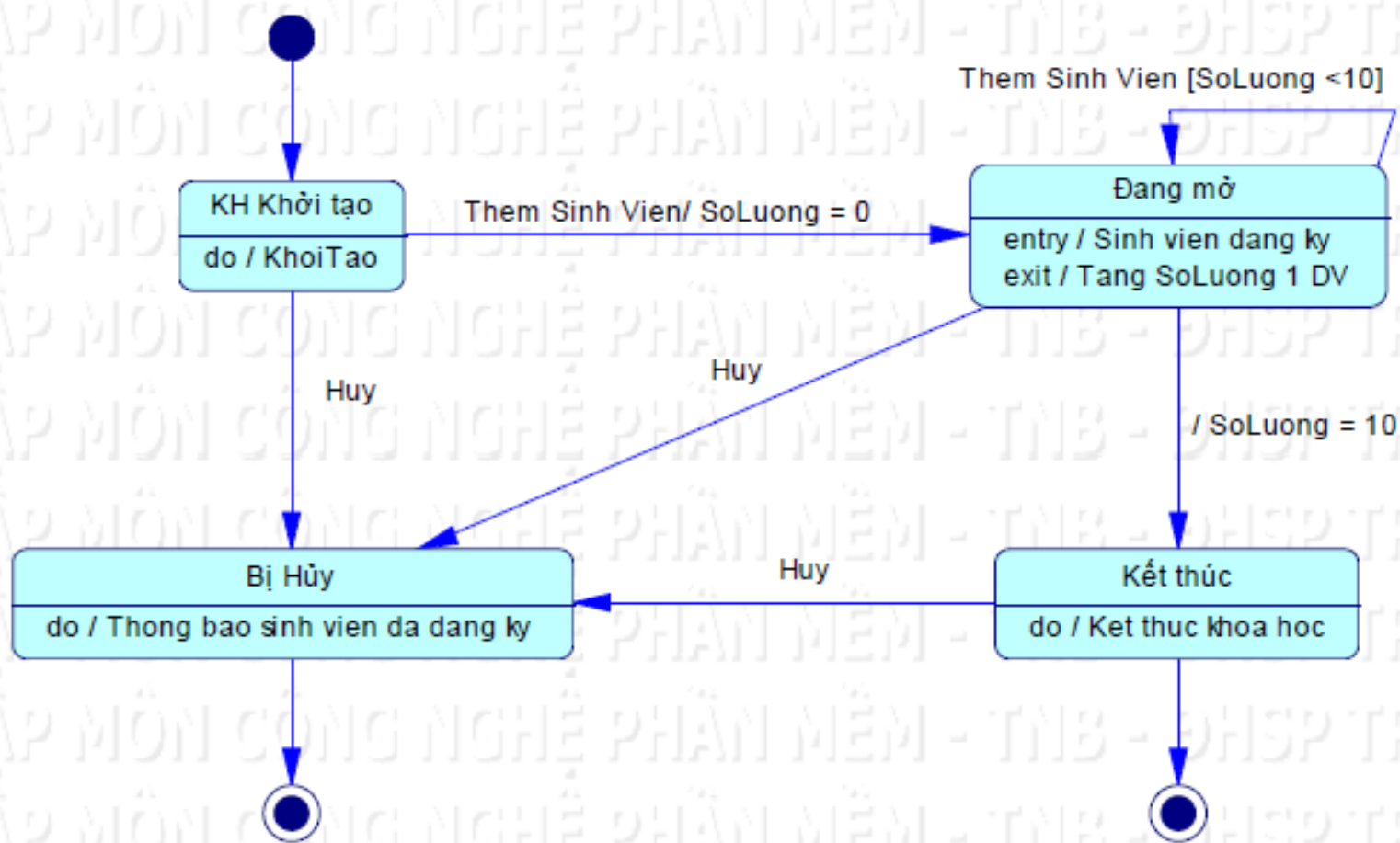
- **Mối liên hệ giữa các trạng thái -Transition**



- **Event**
- **Action**
- **Condition:** điều kiện cho phép chuyển từ trạng thái này sang trạng thái khác

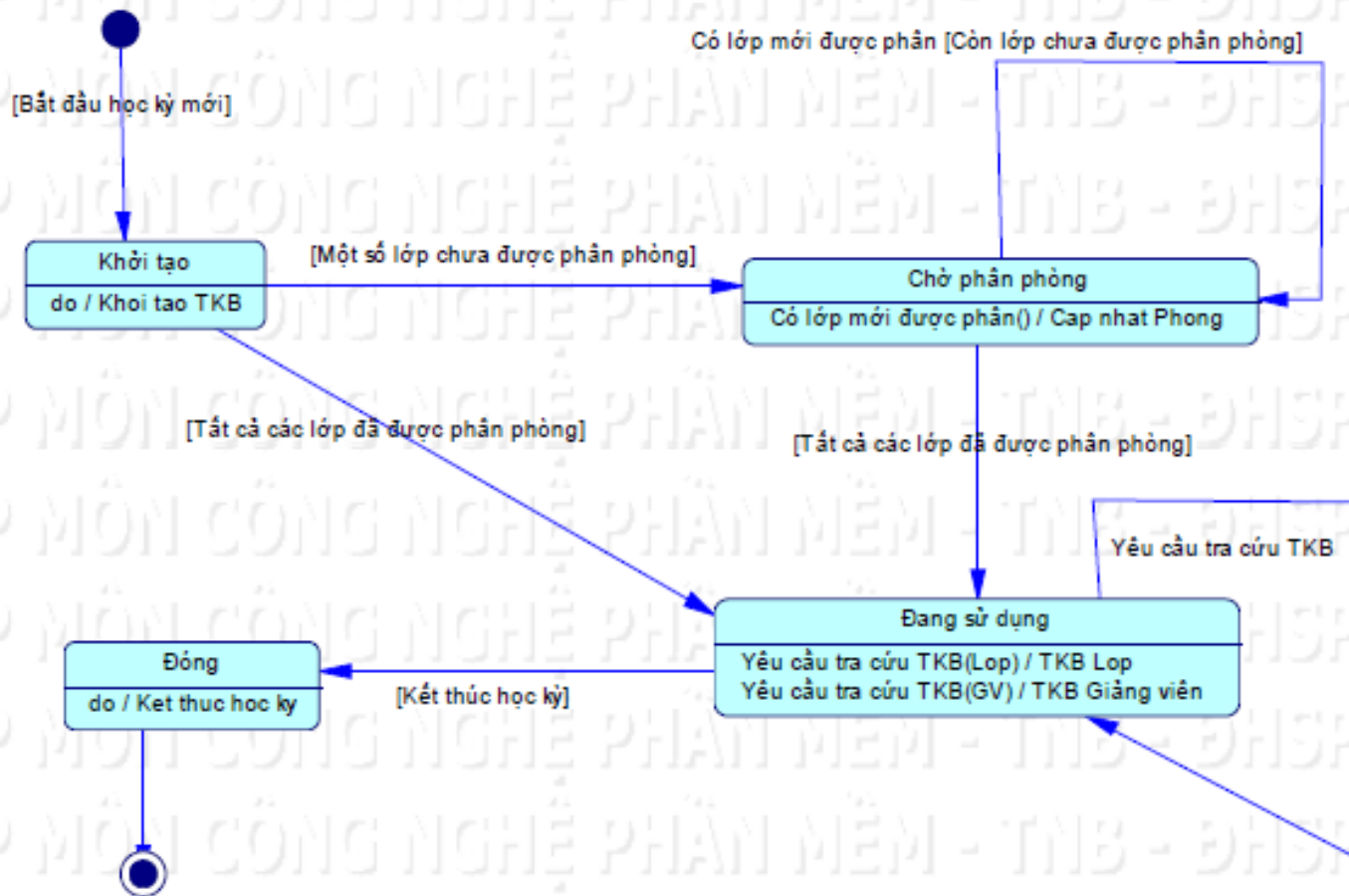
# SƠ ĐỒ TRẠNG THÁI

- State – Event – Action - Transition



# SƠ ĐỒ TRẠNG THÁI

- **State diagram** mô tả trạng thái thời khóa biểu



Ví dụ minh họa State diagram



# SƠ ĐỒ TRẠNG THÁI

- **State diagram** mô trạng của màn hình quản lý danh mục giáo viên

**Danh mục giáo viên**

Thông tin giáo viên:

Mã GV: NV005      Họ tên: Lý Thành

Địa chỉ: 123 Trương Định - Quận 3

Điện thoại: 9321 213      Email: lthanh@yahoo.com

Ghi chú:

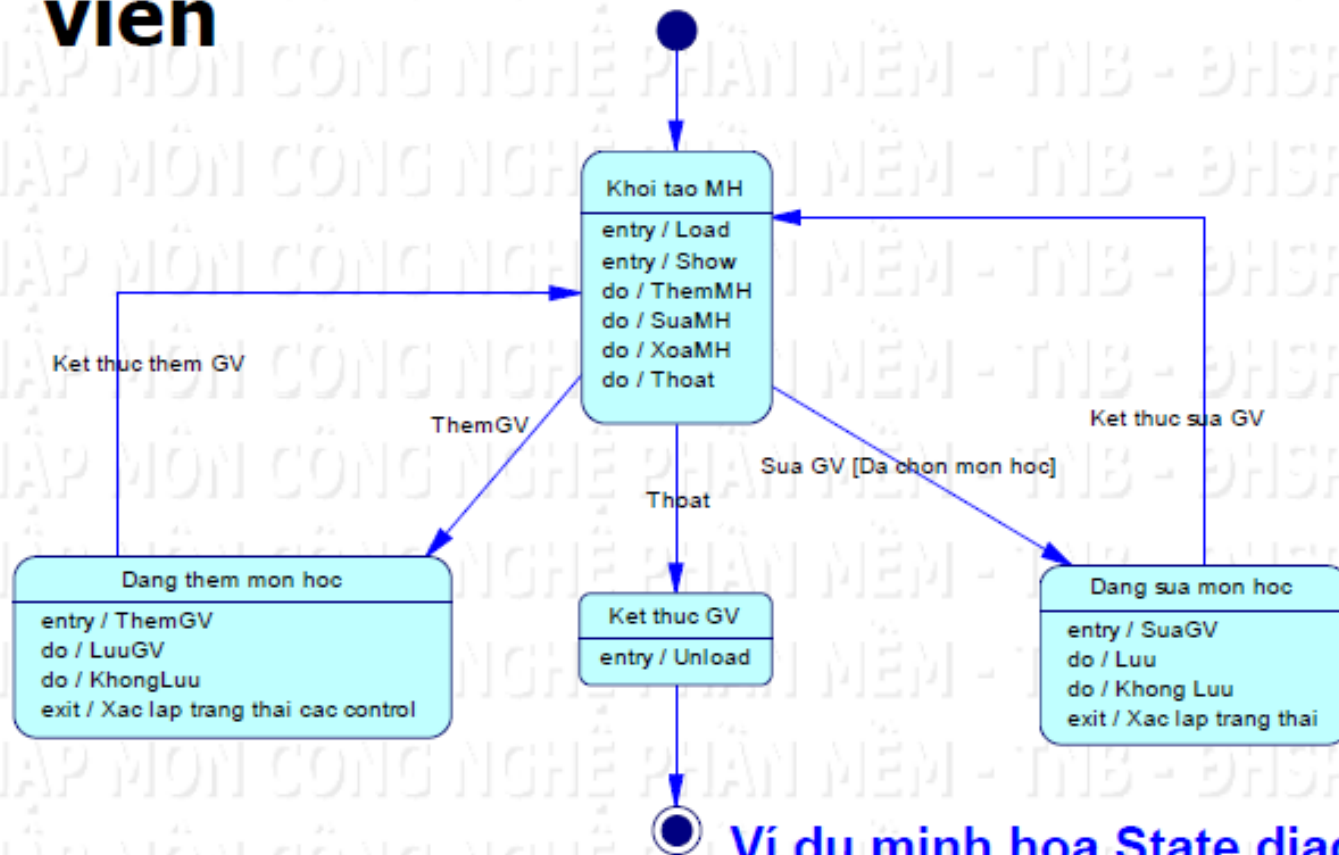
Danh sách giáo viên:

Mã GV	Họ Tên	Địa chỉ	Điện thoại	Email
NV004	Cao Thị Tố Trinh	24 Nguyễn Thị Minh Khai - Q.3	9345882	ctttrinh@yahoo.com
NV005	Lý Thành	123 Trương Định - Quận 3	9321213	lthanh@yahoo.com
NV006	Trần Thị Mỹ Châu	26 Nguyễn Bình Khiêm - Quận 1	0913670277	
NV007	Trần Thị Minh Nguyệt	32 Trần Bình Trọng - Quận 5		
NV008	Phan Thị Anh Khanh	67 Trần Bình Trọng - Quận 5		ptakhanh@yahoo.co
NV009	Hồ Anh Thư	12 Trần Bình Trọng - Quận 5		hathu@yahoo.com
NV010	Trần Ngọc Dung	123/4 Nguyễn Thị Minh Khai - Quận 1	9456123	trndung@yahoo.com

Thêm Xóa Sửa Ghi Không Thoát

# SƠ ĐỒ TRẠNG THÁI

- State diagram** mô tả trạng thái của màn hình quản lý danh mục giáo viên



Ví dụ minh họa State diagram

# SƠ ĐỒ TRẠNG THÁI

- Lưu ý:
  - Tên trạng thái là duy nhất trong sơ đồ
  - Các hành động bên trong: các hành động hoặc tác vụ được thực hiện khi nằm ở trạng thái đang xét , có cú pháp:  
action-label / action –expression
- Các nhãn hành động khác chỉ ra sự kiện kích hoạt hành động tương ứng trong biểu thức hành động(action-expression)
- Cú pháp của biểu thức hành động:

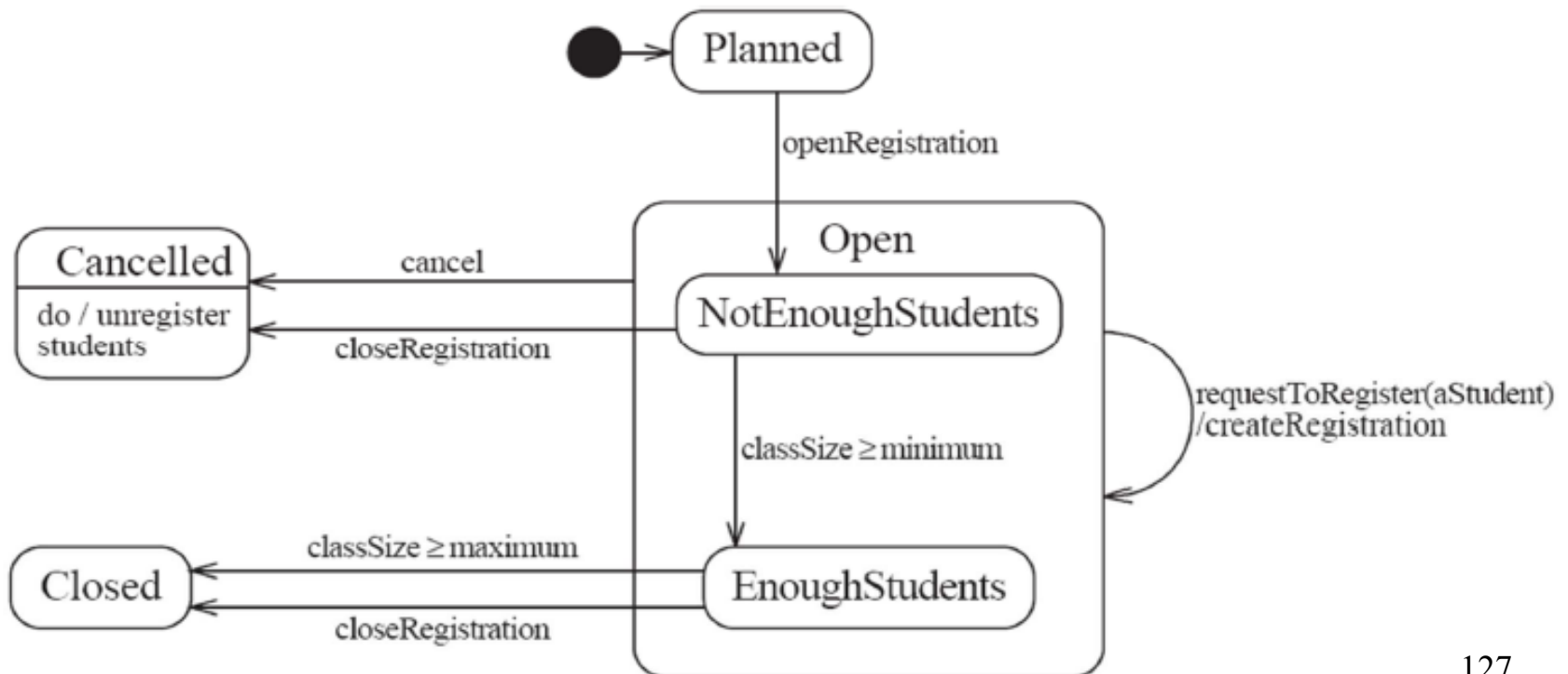
```
event-name ' ( ' parameter-list ' ) ' ' [ 'guard-condition' ] '  
' / ' action-expression
```

# SƠ ĐỒ TRẠNG THÁI

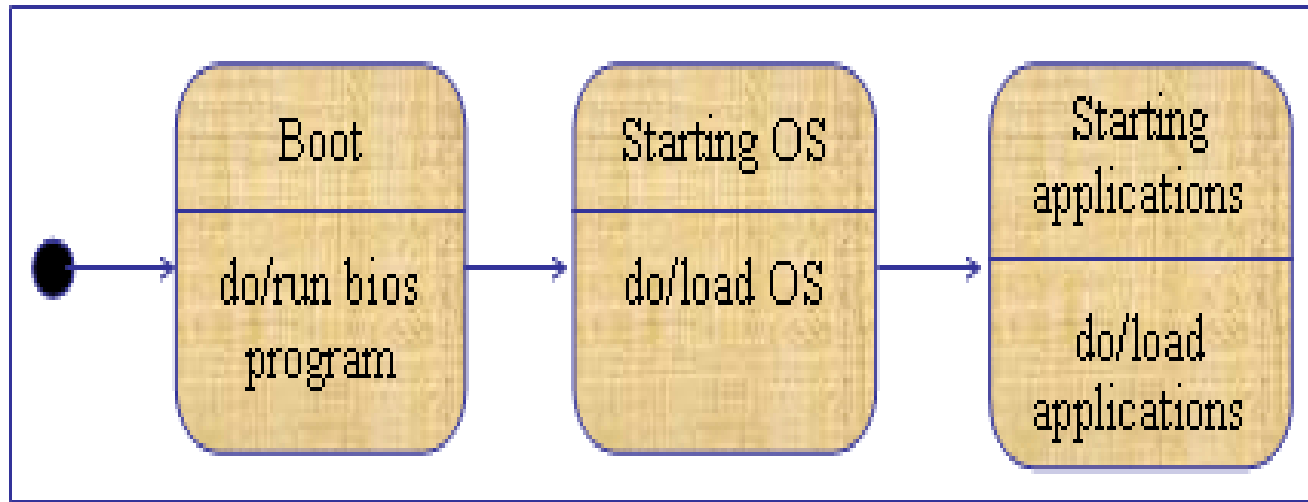
- Cú pháp nhãn:

**even t-signature[guard-condition] /action-expression**

**VD: sơ đồ trạng thái của lớp Message**



# SƠ ĐỒ TRẠNG THÁI



Biến đổi trạng thái không có sự kiện từ ngoài. Sự thay đổi trạng thái xảy ra khi các hoạt động trong mỗi trạng thái được thực hiện xong.



# NHẬN BIẾT TRẠNG THÁI VÀ SỰ KIỆN

- Quá trình phát hiện sự kiện và trạng thái về mặt bản chất bao gồm việc hỏi một số các câu hỏi thích hợp:
  - Một đối tượng có thể có những trạng thái nào?: Hãy liệt kê ra tất cả những trạng thái mà một đối tượng có thể có trong vòng đời của nó.
  - Những sự kiện nào có thể xảy ra?: Bởi sự kiện gây ra việc thay đổi trạng thái nên nhận ra các sự kiện là một bước quan trọng để nhận diện trạng thái.
  - Trạng thái mới sẽ là gì?: Sau khi nhận diện sự kiện, hãy xác định trạng thái khi sự kiện này xảy ra và trạng thái sau khi sự kiện này xảy ra.
  - Có những thủ tục nào sẽ được thực thi?: Hãy để ý đến các thủ tục ảnh hưởng đến trạng thái của một đối tượng.
  - Chuỗi tương tác giữa các đối tượng là gì?: Tương tác giữa các đối tượng cũng có thể ảnh hưởng đến trạng thái của đối tượng.
  - Qui định nào sẽ được áp dụng cho các phản ứng của các đối tượng với nhau?: Các qui định kiểm tỏa phản ứng đối với một sự kiện sẽ xác định rõ hơn các trạng thái.

# NHẬN BIẾT TRẠNG THÁI VÀ SỰ KIỆN

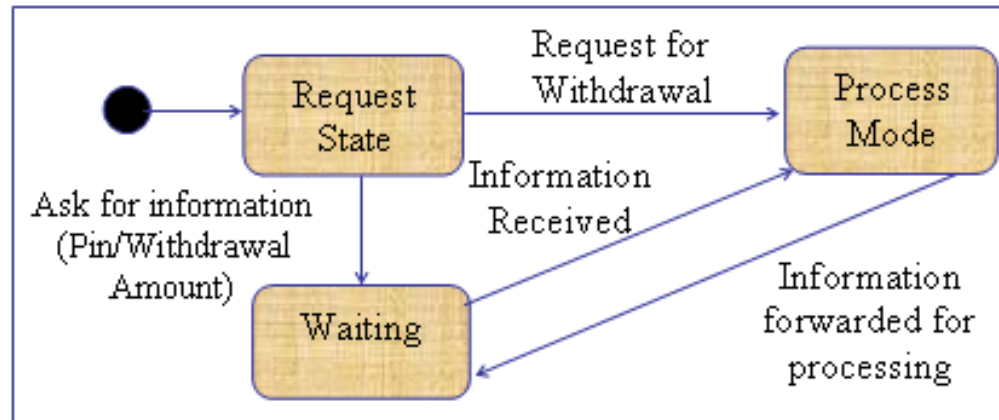
- Những sự kiện và sự chuyển tải nào là không thể xảy ra?: Nhiều khi có một số sự kiện hoặc sự thay đổi trạng thái không thể xảy ra. Ví dụ như bán một chiếc ô tô đã được bán rồi.
- Cái gì khiến cho một đối tượng được tạo ra?: Đối tượng được tạo ra để trả lời cho một sự kiện. Ví dụ như một sinh viên ghi danh cho một khóa học.
- Cái gì khiến cho một đối tượng bị hủy?: Đối tượng sẽ bị hủy đi khi chúng không được cần tới nữa. Ví dụ khi một sinh viên kết thúc một khóa học.
- Cái gì khiến cho đối tượng cần phải được tái phân loại (reclassified)? : Những loại sự kiện như một nhân viên được tăng chức thành nhà quản trị sẽ khiến cho động tác tái phân loại của nhân viên đó được thực hiện.

# NHẬN BIẾT TRẠNG THÁI VÀ SỰ KIỆN

- Một số lời mách bảo cho việc tạo dựng biểu đồ trạng thái
  - Chuyển biểu đồ trình tự thành biểu đồ trạng thái
  - Xác định các vòng lặp (loop)
  - Bổ sung thêm các điều kiện biên và các điều kiện đặc biệt
  - Trộn lẫn các kịch bản khác vào trong biểu đồ trạng thái
- Một khi mô hình đã được tạo nên, hãy nêu ra các câu hỏi và kiểm tra xem mô hình có khả năng cung cấp tất cả các câu trả lời. Qui trình sau đây cần phải được nhắc lại cho mỗi đối tượng.

# Chuyển biểu đồ tuần tự thành biểu đồ trạng thái

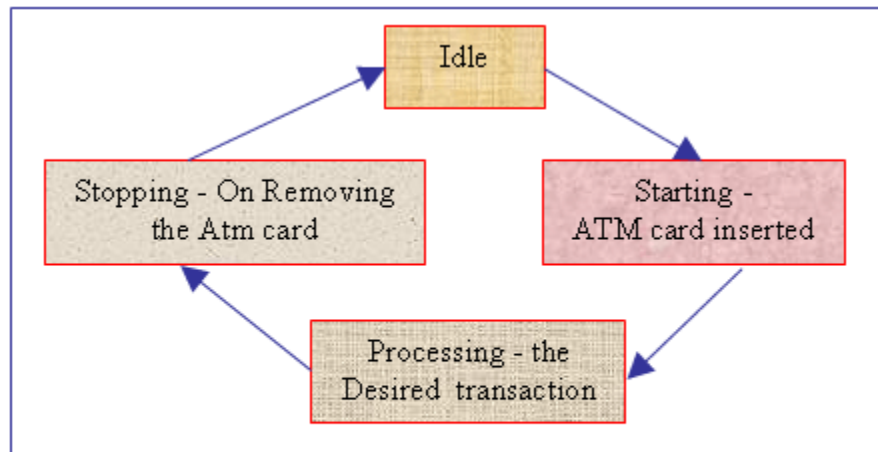
- Sắp xếp các sự kiện thành một đường dẫn, dán nhãn input (hoặc entry) và output (exit) cho các sự kiện. Khoảng cách giữa hai sự kiện này sẽ là một trạng thái.
- Nếu cảnh kịch có thể được nhắc đi nhắc lại rất nhiều lần (vô giới hạn), hãy nối đường dẫn từ trạng thái cuối cùng đến trạng thái đầu tiên



**Hình 6.10-** Chuyển một biểu đồ tuần tự sang biểu đồ trạng thái

# Nhận ra các vòng lặp (loop)

- Một chuỗi sự kiện có thể được nhắc đi nhắc lại vô số lần được gọi là vòng lặp (loop).
- Chú ý:
  - Trong một vòng lặp, chuỗi các sự kiện được nhắc đi nhắc lại cần phải đồng nhất với nhau. Nếu có một chuỗi các sự kiện khác chuỗi khác thì trường hợp đó không có vòng lặp.
  - Lý tưởng nhất là một trạng thái trong vòng lặp sẽ có sự kiện kết thúc. Đây là yếu tố quan trọng, nếu không thì vòng lặp sẽ không bao giờ kết thúc.



**Hình 6.11- Biểu đồ lặp**



# **Bổ sung thêm các điều kiện biên và các điều kiện đặc biệt**

- Kiểm tra, đối chứng chúng với điều kiện biên và các điều kiện đặc biệt khác, những điều kiện rất có thể đã chưa được quan tâm đủ độ trong thời gian tạo dựng biểu đồ trạng thái.
- Điều kiện biên là những điều kiện thao tác trên giá trị, đây là những giá trị nằm bên ranh giới của một điều kiện để quyết định về trạng thái của đối tượng.
  - Ví dụ như quy định về kỳ hạn của một tài khoản là 30 ngày thì ngày thứ 31 đối với tài khoản này sẽ là một điều kiện biên.
- Các điều kiện đặc biệt là những điều kiện ngoại lệ
  - Ví dụ ngày thứ 30 của tháng 2 năm 2000 (nếu có một điều kiện thật sự như vậy tồn tại ngoài đời thực).

# Trộn lẫn các cảnh kịch khác vào trong biểu đồ trạng thái

- Ấn định một điểm bắt đầu chung cho tất cả các chuỗi sự kiện bổ sung.
- Xác định điểm nơi các ứng xử bắt đầu khác biệt với những ứng xử đã được mô hình hóa trong biểu đồ trạng thái.

## Chú ý:

- Biểu đồ trạng thái chỉ cần được tạo dựng nên cho các lớp đối tượng có ứng xử động quan trọng.
- Hãy thẩm tra biểu đồ trạng thái theo khía cạnh tính nhất quán đối với những sự kiện dùng chung để cho toàn bộ mô hình động được đúng đắn.
- Dùng các trường hợp sử dụng để hỗ trợ cho quá trình tạo dựng biểu đồ trạng thái.
- Khi định nghĩa một trạng thái, hãy chỉ đề ý đến những thuộc tính liên quan.

# CÂU HỎI VÀ BÀI TẬP

---

## 1. Hỏi: Thế nào là một vòng lặp?

**Đáp:** Một chuỗi sự kiện có thể được nhắc đi, nhắc lại vô số lần được gọi là vòng lặp (loop).

## 2. Hỏi: Mô hình động chính là mô hình đối tượng cộng thêm phần ứng xử động của hệ thống

**Đáp:** Đúng

## 3. Hỏi: Các sự kiện độc lập cũng có thể là các sự kiện song song

**Đáp:** Đúng

## 4. Hỏi: Một đối tượng không nhất thiết phải có trạng thái.

**Đáp:** Sai, mọi đối tượng đều có trạng thái

# CÂU HỎI VÀ BÀI TẬP

---

1. Một lớp có thể có trạng thái ban đầu và trạng thái kết thúc.

**Đáp:** Sai, một đối tượng có thể có trạng thái ban đầu và trạng thái kết thúc.

2. Hỏi: Một vòng đời (chu trình) vòng lặp của đối tượng không có trạng thái khởi tạo cũng không có trạng thái kết thúc

**Đáp:** Đúng, đối tượng được coi là đã luôn luôn tồn tại ở đây và sẽ còn mãi mãi tiếp tục tồn tại.

# CÂU HỎI VÀ BÀI TẬP

---

- Bài tập: trong hệ thống ATM chúng ta xem hoạt động của use case “Rút tiền”. Các hoạt động tuần tự mà khách hàng thực hiện:
  - Đưa vào thẻ ATM
  - Nhập mã PIN
  - Rút t ATMVẽ các sơ trình tự, cộng tác và hoạt động để hiện thực hóa use case trên