YÊU CẦU PHẦN MỀM

ThS. Dương Hữu Thành
Khoa CNTT, Đại học Mở Tp.HCM.
Email: thanh.dh@ou.edu.vn

Nội dung

- Yêu cầu phần mềm
- Khảo sát hiện trạng
- Phân loại yêu cầu
- Kỹ thuật xác định yêu cầu
- Mô hình hóa yêu cầu
- Phân tích yêu cầu
- Đặc tả yêu cầu

Yêu cầu phần mềm

- Yêu cầu phần mềm là các ***công việc, nghiệp vụ*** được hổ trợ thực hiện trên máy tính với phần mềm.
- Mục đích bước xác định yêu cầu phần mềm
  - Đi đến ***thỏa thuận*** khách hàng và người dùng về các chức năng hệ thống
  - Cho phép nhà phát triển hiểu hơn yêu cầu hệ thống
  - Xác định ***phạm vị hệ thống***
  - Lập kế hoạch phát triển hệ thống
  - Xác định giao diện người dùng

Khảo sát hiện trạng

- Mục đích của khảo sát hiện trạng là ***tìm hiểu thế giới thực*** liên quan đến phần mềm.
- Các hiện trạng cần khảo sát
  - ***Hiện trạng tổ chức***: tìm hiểu sơ đồ tổ chức các phòng ban bên trong, các phòng ban có giao tiếp bên ngoài và mối quan hệ giữa các bộ phận.
  - ***Hiện trạng tin học***: khảo sát hiện trạng phần cứng, phần mềm, con người.
  - ***Hiện trạng nghiệp vụ***: khảo sát danh sách nghiệp vụ của từng bộ phận, bao gồm tên nghiệp vụ, biểu mẫu liên quan, người thực hiện, thời điểm, tần suất, và cách thức thực hiện nghiệp vụ.

Phân loại yêu cầu

- ***Yêu cầu chức năng***: mô tả phần mềm có thể làm được cái gì?
  - Đăng ký
  - Đăng nhập
  - Tra cứu
  - Tính toán
  - Xuất báo cáo
- ***Yêu cầu phi chức năng***
  - Tính dễ sử dụng
  - Độ tin cậy
  - Độ ổn định
  - Tốc độ xử lý
  - Tính tiền hóa
  - Tính hiệu quả

Kỹ thuật thu thập yêu cầu (1/2)

- Phỏng vấn
  - Phỏng vấn cá nhân/ phỏng vấn nhóm
  - Phỏng vấn tự do/ phỏng vấn có định hướng
- Khi phỏng vấn cần ghi nhận thông tin
  - Nội dung
  - Khi nào có nội dung
  - Bằng cách nào có nội dung thông tin
  - Đánh giá của người được phỏng vấn về tình hình hiện tại của nghiệp vụ
- Khi phỏng vấn không nên
  - Đưa nhận xét cá nhân người phỏng vấn
  - Dùng thuật ngữ/ngôn ngữ tin học

Kỹ thuật thu thập yêu cầu (2/2)

- Bảng câu hỏi
- Nghiên cứu tài liệu
- Quan sát thực tế
- ...

Làm việc nhóm (1/3)

- Chia làm nhiều nhóm, mỗi nhóm 3 thành viên
- Bắt cặp hai nhóm với nhau
  - Một nhóm đóng vai trò khách hàng, đưa ra phác thảo dự án
  - Một nhóm đóng vai trò nhà phát triển thực hiện phỏng vấn lấy yêu cầu
- Viết lại các đặc tả yêu cầu sau buổi phỏng vấn

Làm việc nhóm (2/3)

- Sau buổi phỏng vấn, trả lời các câu hỏi
  - Khách hàng đã thật sử hiểu rõ điều mình muốn chưa?
  - Nhóm phát triển có hiểu rõ yêu cầu khách hàng chưa, có cần buổi phỏng vấn khác hay thông tin gì thêm để hiểu rõ hơn yêu cầu?
  - Dự án có khả thi hay không? Những yêu cầu nào không khả thi và cần thảo luận thêm với khách hàng?
  - Những khó khăn và thuận lợi trong quá trình lấy yêu cầu khách hàng?
  - Bạn có nhận dự án hay không? Tại sao?

Làm việc nhóm (3/3)

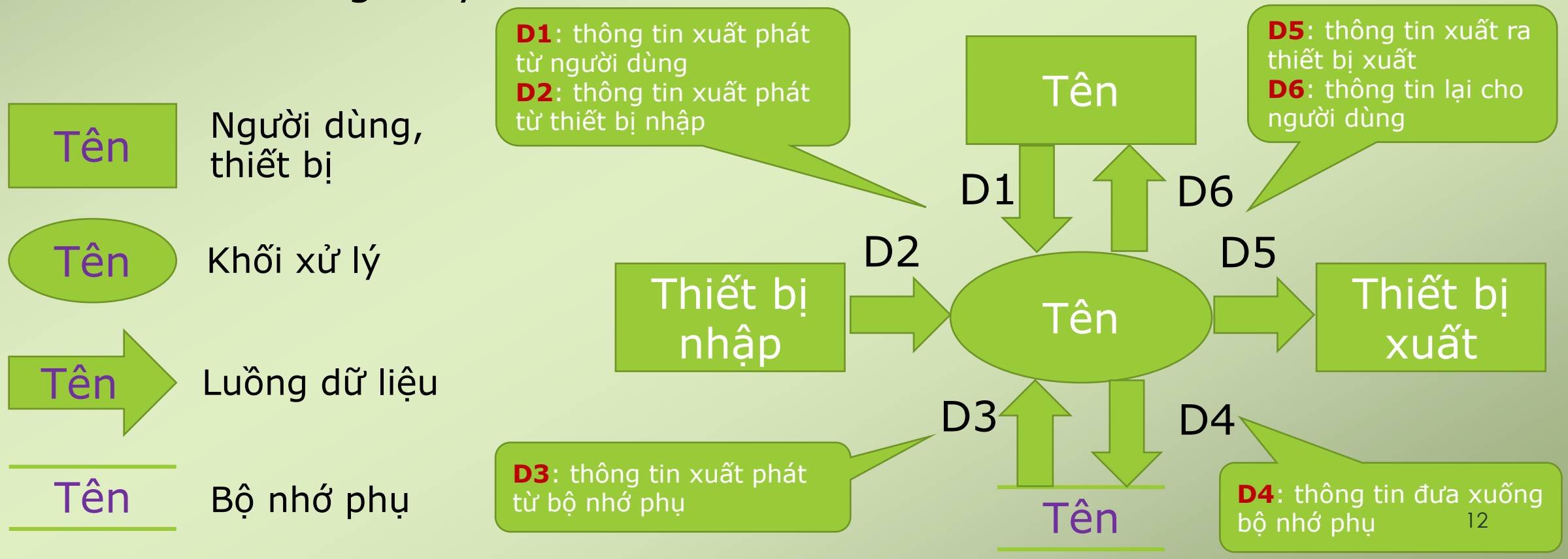
- Nộp 1 bản đặc tả yêu cầu từ khách hàng
- Nộp danh sách các câu hỏi phỏng vấn và các câu trả lời tự khách hàng.
- Nộp danh sách câu hỏi và câu trả lời phát sinh trong lúc thảo luận.

Mô hình hóa yêu cầu (1/3)

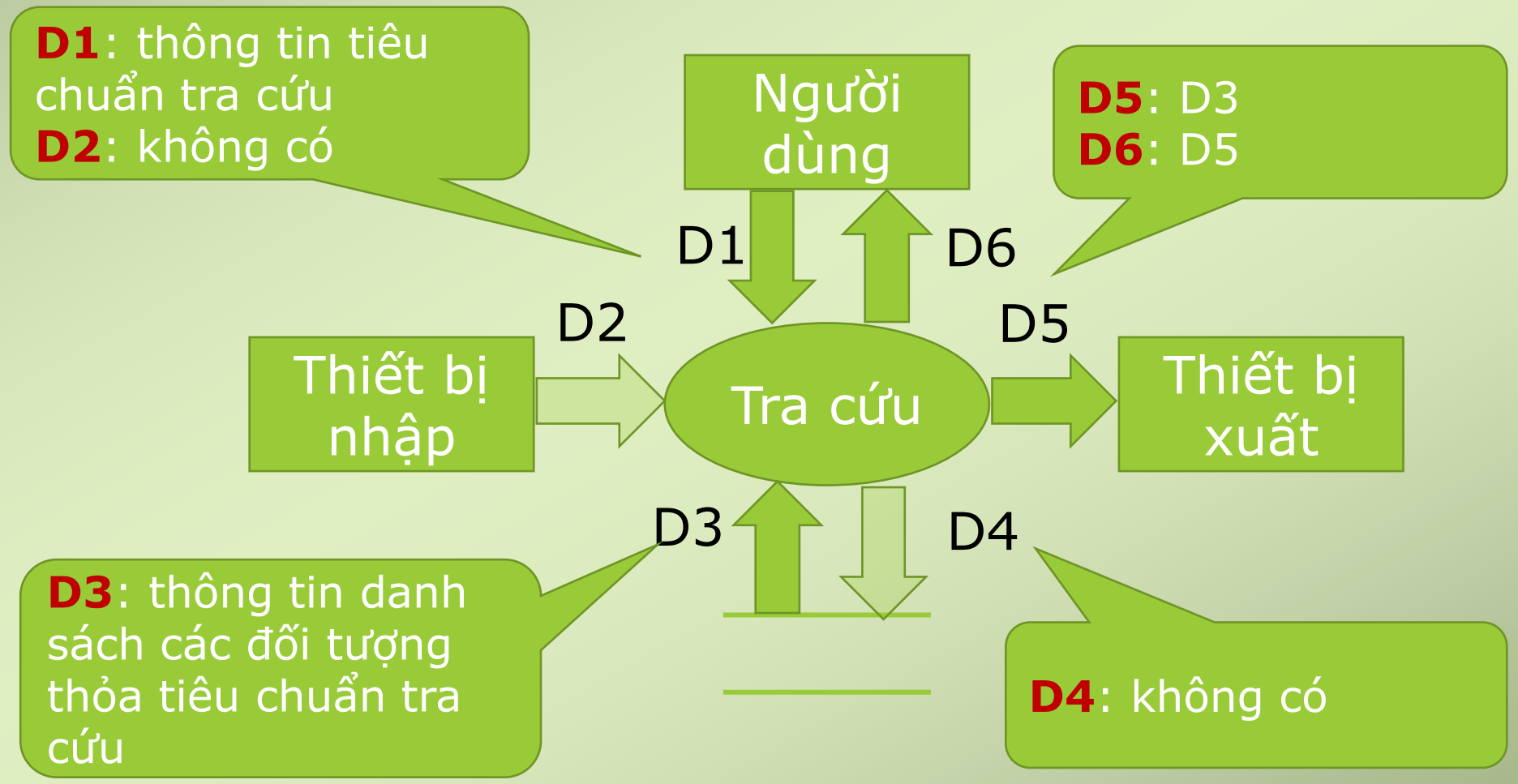
- Mục tiêu: ***mô hình hóa thế giới thực*** với các yêu cầu đã xác định.
- Kết quả
  - ***Sơ đồ luồng dữ liệu*** từng công việc
  - Sơ đồ phối hợp giữa các công việc

Mô hình hóa yêu cầu (2/3)

- Sơ đồ luồng dữ liệu (DFD) là sơ đồ biểu thị các thông tin liên quan đến việc ***thực hiện các nghiệp vụ*** trong thế giới thực bên trong máy tính.



Mô hình hóa yêu cầu (3/3)



Phân tích yêu cầu

- Phân tích yêu cầu nhằm ***xác định những đặc trưng*** nào phù hợp cho sản phẩm dựa trên nhu cầu của stakeholder.
- Stakeholder và developer thường có cách nhìn và cách mô tả khác nhau. Để hiểu tốt hơn yêu cầu phần mềm, kỹ sư phần mềm phải sử dụng những mô tả trừu tượng để dễ dàng hiểu giữa các bên.

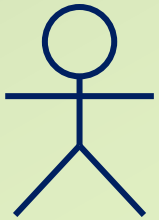
Khái niệm Actor (1/6)

- Actor là tác nhân ***bên ngoài hệ thống*** và ***có tương tác*** với hệ thống.
- Một actor xác định một ***tập các vai trò*** khi người sử dụng tương tác với hệ thống
- Actor có thể là ***con người***, ***phần cứng*** hoặc ***một phần mềm khác***.
- Tên của actor thường là ***danh từ***

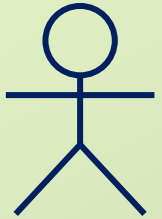
Tên Actor

Khái niệm Actor (2/6)

- Hệ thống đăng ký học phần, cho phép sinh viên vào đăng ký học phần và giảng viên có thể vào xem lịch giảng dạy của mình. Ta xác định được hai actor là sinh viên và giảng viên

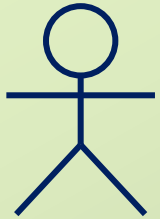


Sinh viên



Giảng viên

- Hệ thống hoạt động ATM của các ngân hàng ta có thể xác định các tác nhân

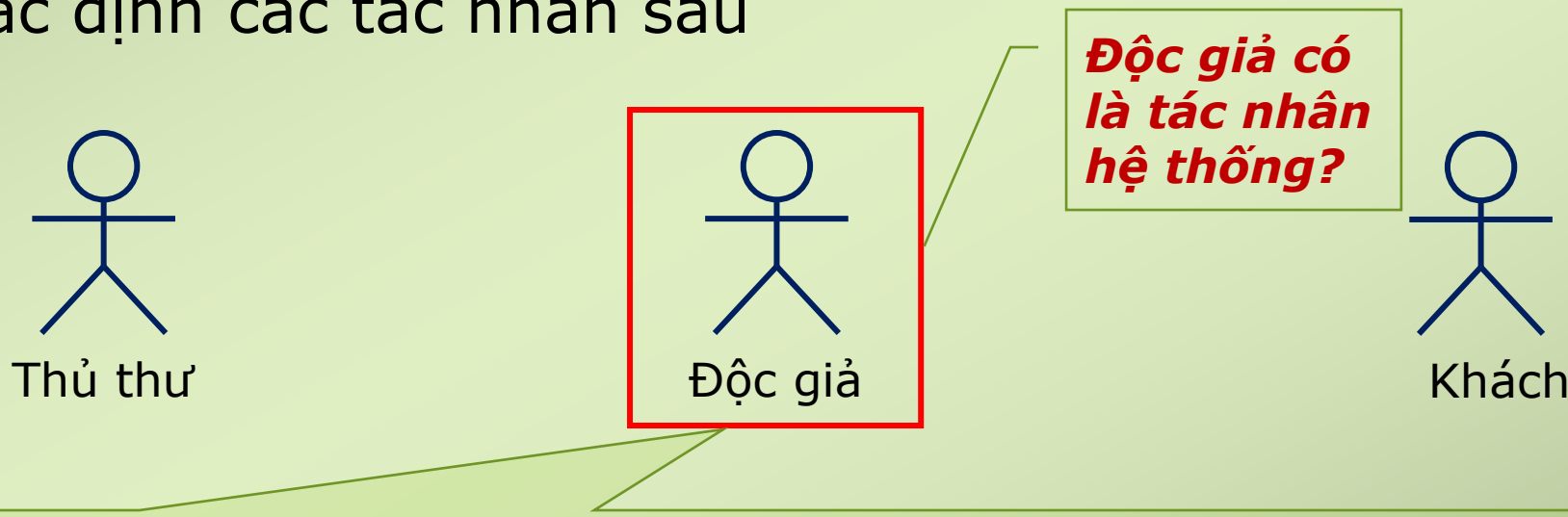


Khách hàng

Nhân viên ngân hàng

Khái niệm Actor (3/6)

- Hệ thống quản lý thư viện bao gồm chức năng nhập sách, mượn sách, trả sách, tra cứu, thống kê và lập báo cáo. Ta có thể xác định các tác nhân sau



Ta cần xét các trường hợp:
- Nếu độc giả chỉ mượn sách thông qua thủ thư thì độc giả không được xem là actor của hệ thống.
- Nếu độc giả có thể đăng ký mượn sách trực tuyến thì độc giả được xem là một tác nhân hệ thống

Khái niệm Actor (4/6)

- Actor có thể là một phần cứng
  - Phần mềm quản lý siêu thị đọc thông tin từ thiết bị độc mã vạch.
  - Phần mềm quản lý ra vào ở các văn phòng đọc tín hiệu từ thẻ từ, phát tín hiệu tự động mở cửa.
  - Phần mềm chống trộm nhận tín hiệu từ camera, phát tín hiệu cho các thiết bị loa, đèn, điện thoại,...

Khái niệm Actor (5/6)

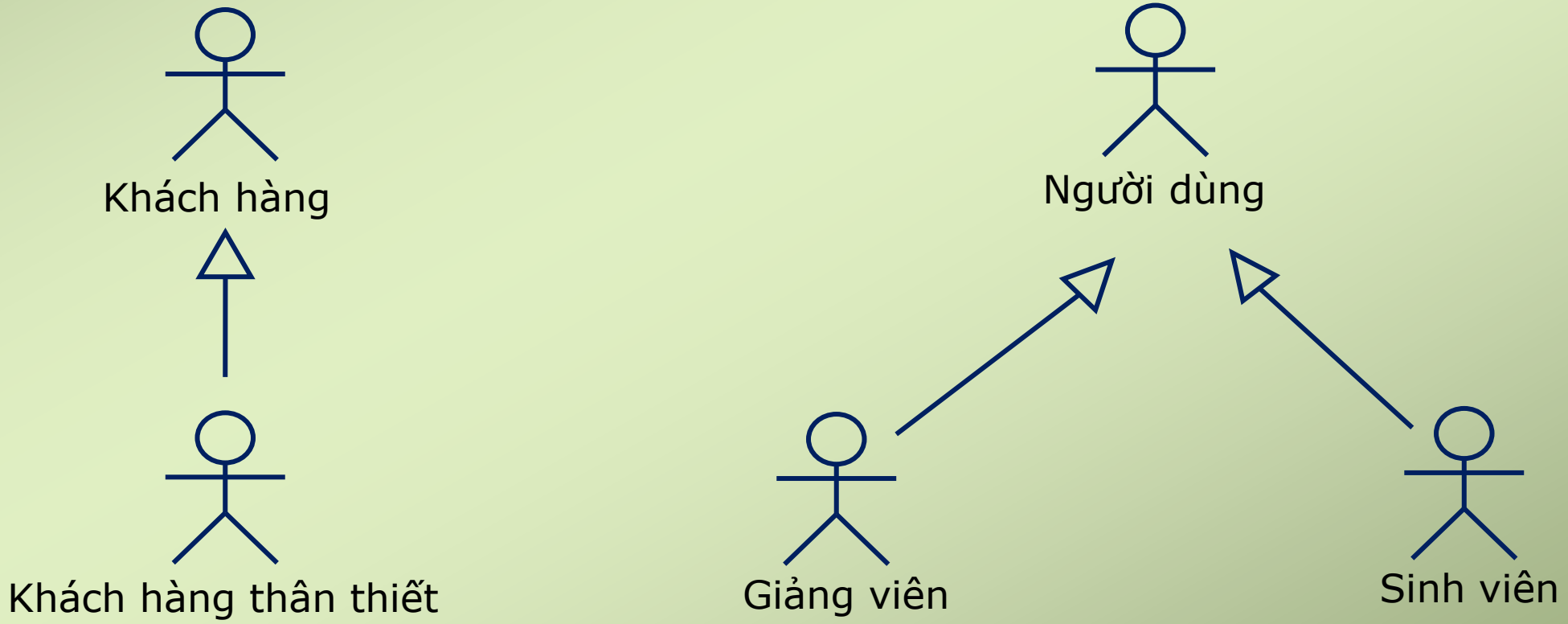
- Actor có thể là một phần mềm
  - Nhập/xuất dữ liệu từ file excel
  - Xuất dữ liệu báo cáo ra phần mềm gửi mail
  - Mở các ứng dụng có thể chia sẽ khi người dùng bấm share một tài nguyên (hình ảnh, video, ...) trên điện thoại.

Khái niệm Actor (6/6)

- Làm thế nào xác định actor?
  - Ai hoặc tổ chức ***sử dụng*** các chức năng của hệ thống?
  - Ai hoặc tổ chức là người ***cung cấp, sử dụng hoắc lấy thông tin*** từ hệ thống?
  - Ai là người ***duy trì, bảo dưỡng và quản*** lý hệ thống
  - ***Tài nguyên bên ngoài*** hệ thống là gì?
  - Có ***hệ thống nào khác tương tác*** với hệ thống này?

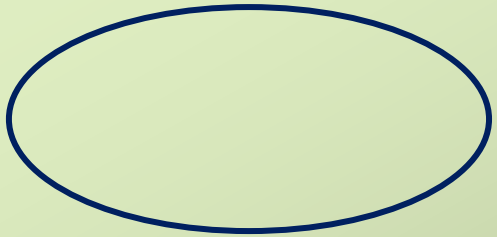
Quan hệ giữa các actor

- Quan hệ tổng quát hóa và chuyên biệt hóa



Khái niệm use-case (1/4)

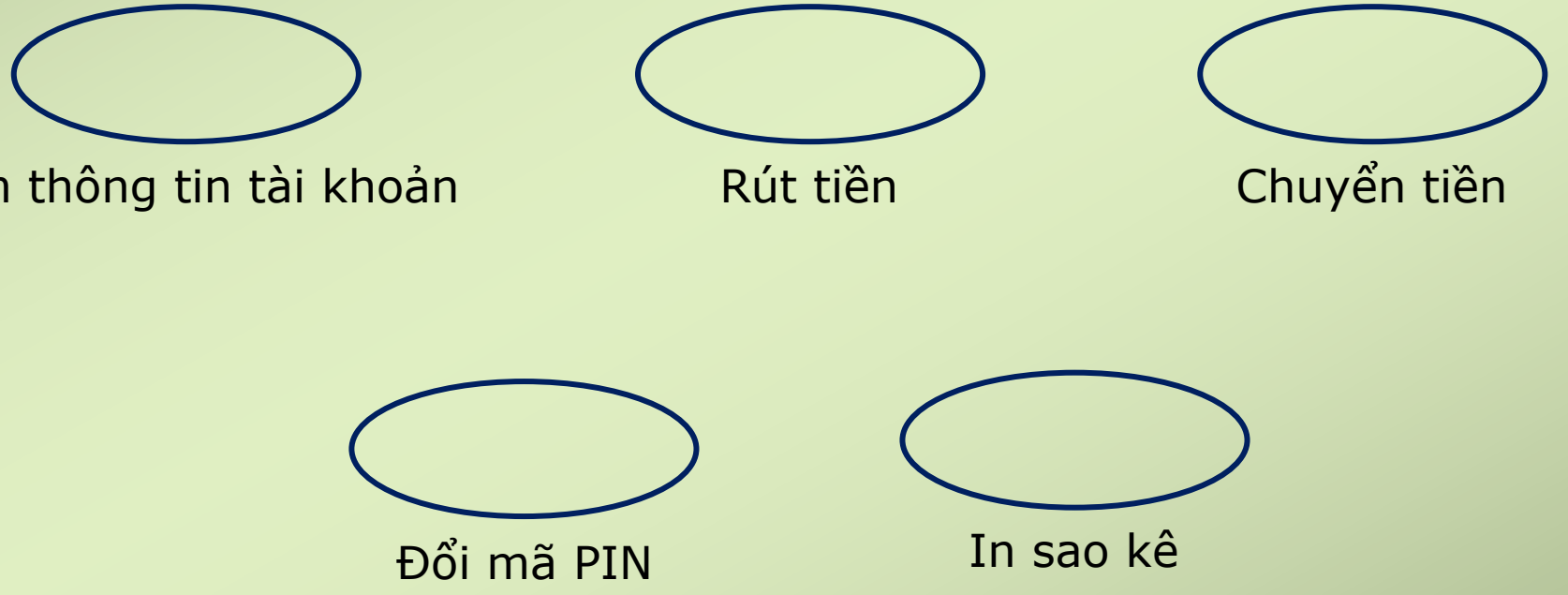
- Use-case là một ***chuỗi các hành động*** mà hệ thống có thể ***thực hiện, tương tác*** với actor của hệ thống.
- Use-case mô tả ***"ai" có thể "làm gì"*** từ quan sát của người dùng.
- Mỗi Use-case được xem là ***một chức năng*** của hệ thống, mang lại một kết quả nhất định đối với người dùng.



Tên Use-case

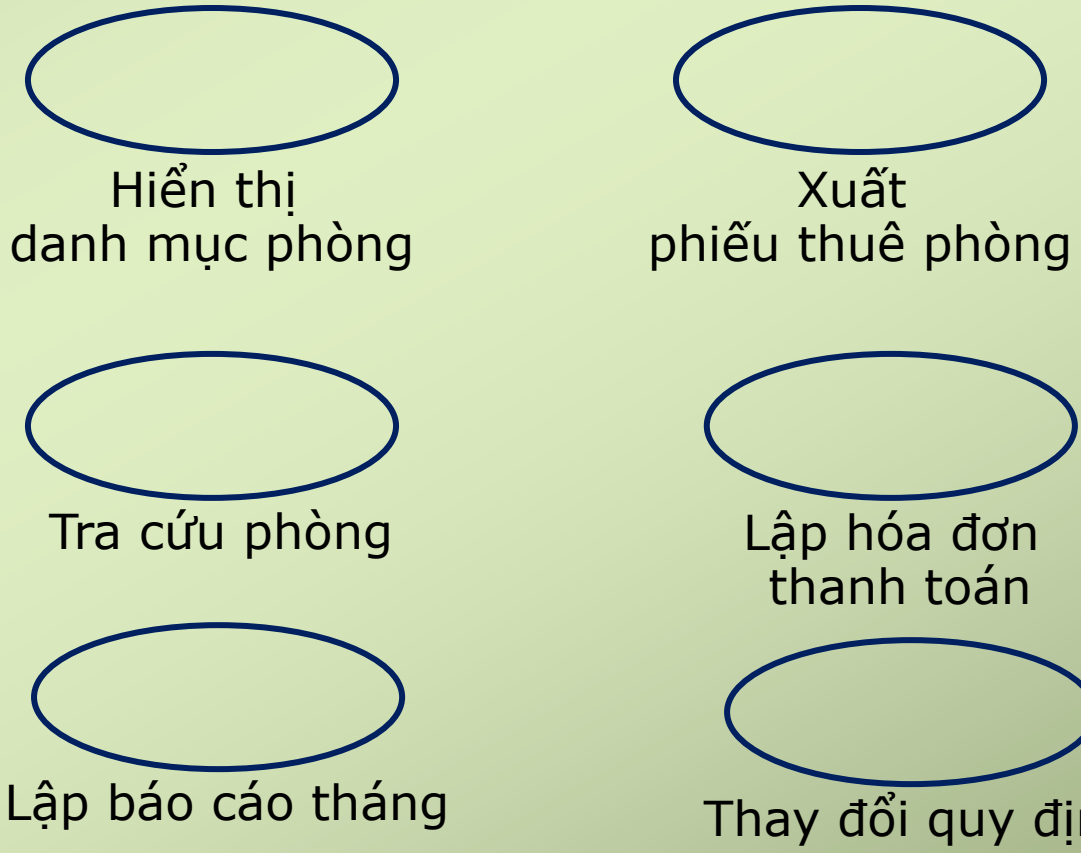
Khái niệm use-case (2/4)

- Trong hệ thống ATM của các ngân hàng, ta có thể xác định các use-case sau



Khái niệm use-case (3/4)

- Ví dụ hệ thống quản lý khách sạn gồm các chức năng
  - Lập danh mục phòng
  - Lập phiều thuê phòng
  - Tra cứu phòng
  - Lập hóa đơn thanh toán
  - Lập báo cáo tháng
  - Thay đổi quy định

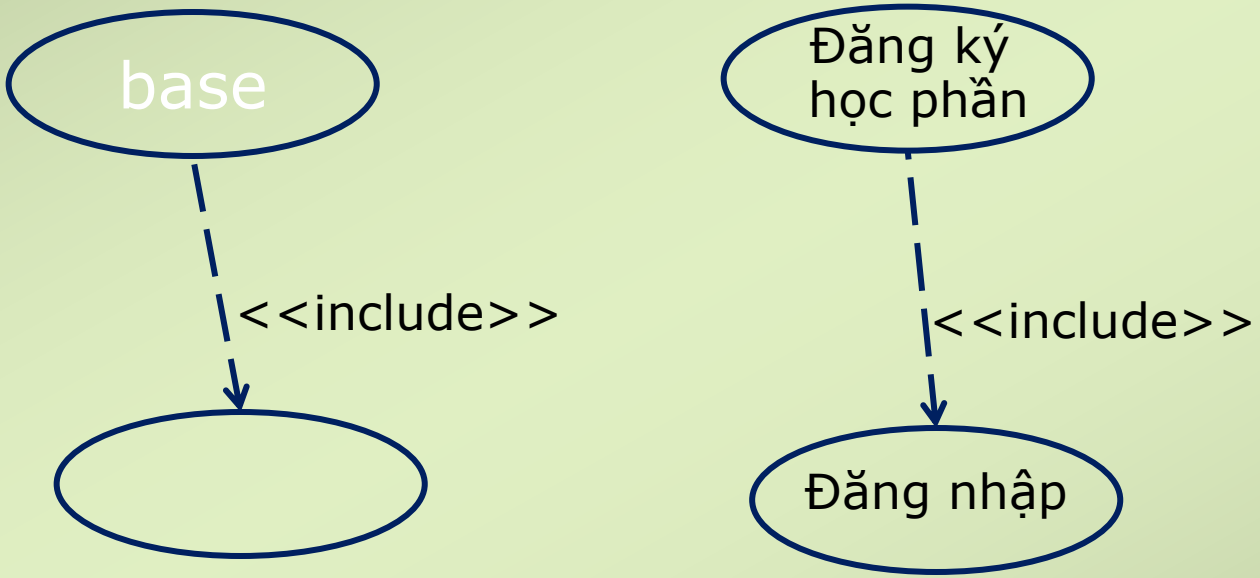


Khái niệm use-case (4/4)

- Làm thế nào xác định use-case?
  - Với mỗi actor, xác định những ***việc liên quan*** đến actor?
  - Hệ thống ***hỗ trợ những nghiệp vụ nào*** trong thế giới thực?
  - Những ***thông tin nào cần được quản lý*** trong hệ thống?
  - Những thông tin nào ***cần được kết xuất*** khỏi hệ thống?

Mối quan hệ giữa các use-case (1/2)

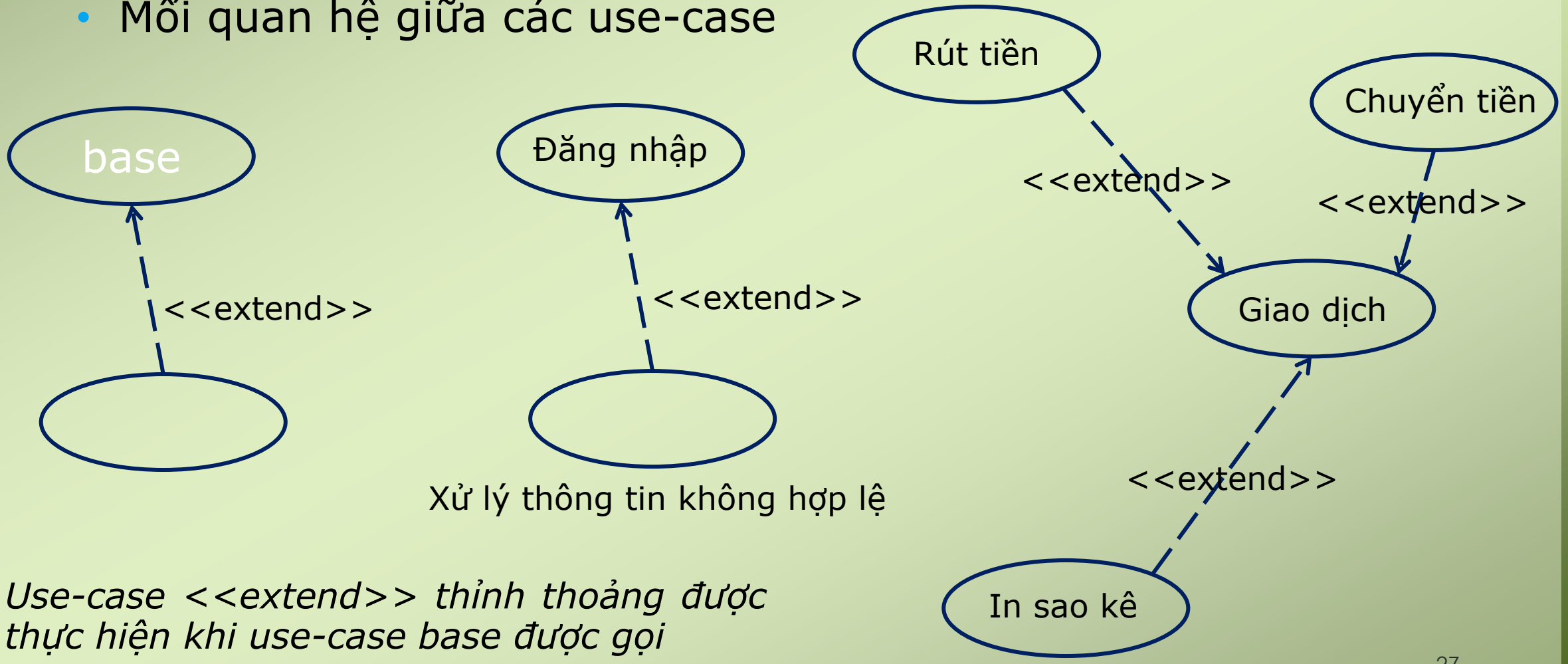
- Mối quan hệ giữa các use-case



Use-case <<include>> luôn luôn được thực hiện khi use-case base được gọi

Mối quan hệ giữa các use-case (2/2)

- Mối quan hệ giữa các use-case



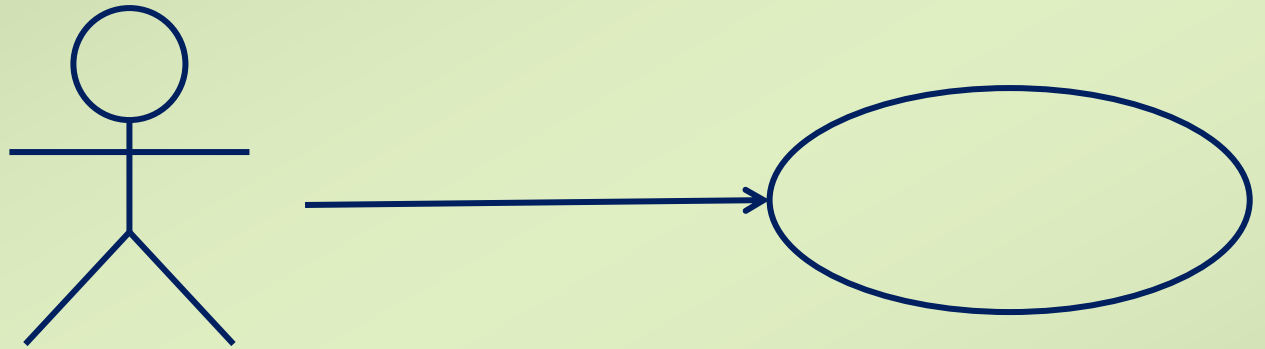
Use-case <<extend>> thỉnh thoảng được thực hiện khi use-case base được gọi

Lược đồ use-case (1/6)

- Mục đích chỉnh của lược đồ use-case (UCD – User-case Diagram) là ***mô tả chức năng*** gì của hệ thống được thực hiện bởi actor nào và vai trò các actor trong hệ thống.
- Một UCD có 4 thành phần chính
  - Các actor tương tác với hệ thống
  - Chính hệ thống
  - Các use-case hay các dịch vụ mà hệ thống thực hiện
  - Các đường thẳng thể hiện mối quan hệ giữa các thành phần

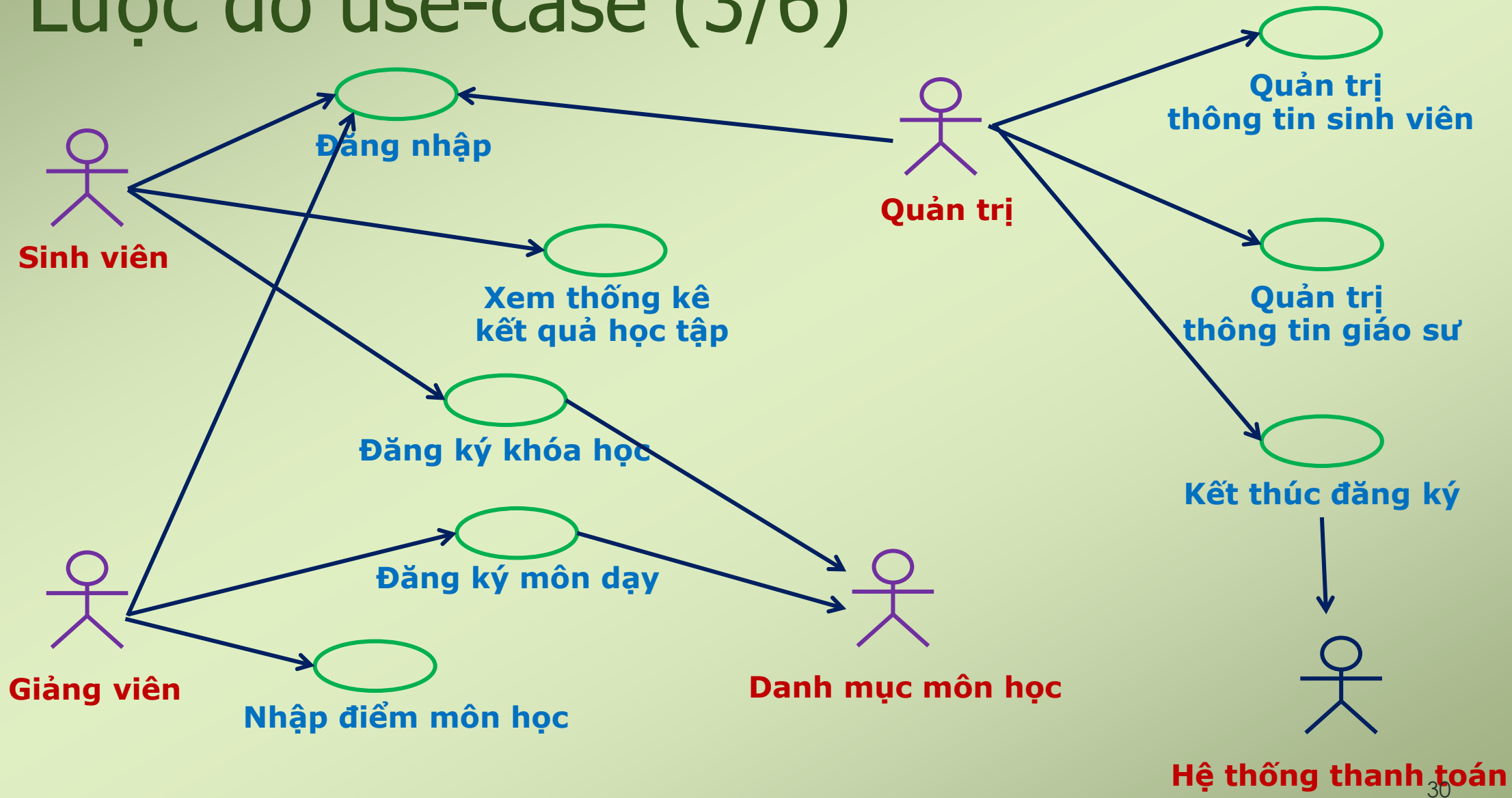
Lược đồ use-case (2/6)

- Lược đồ use-case thể hiện sự tương tác giữa actor và use-case

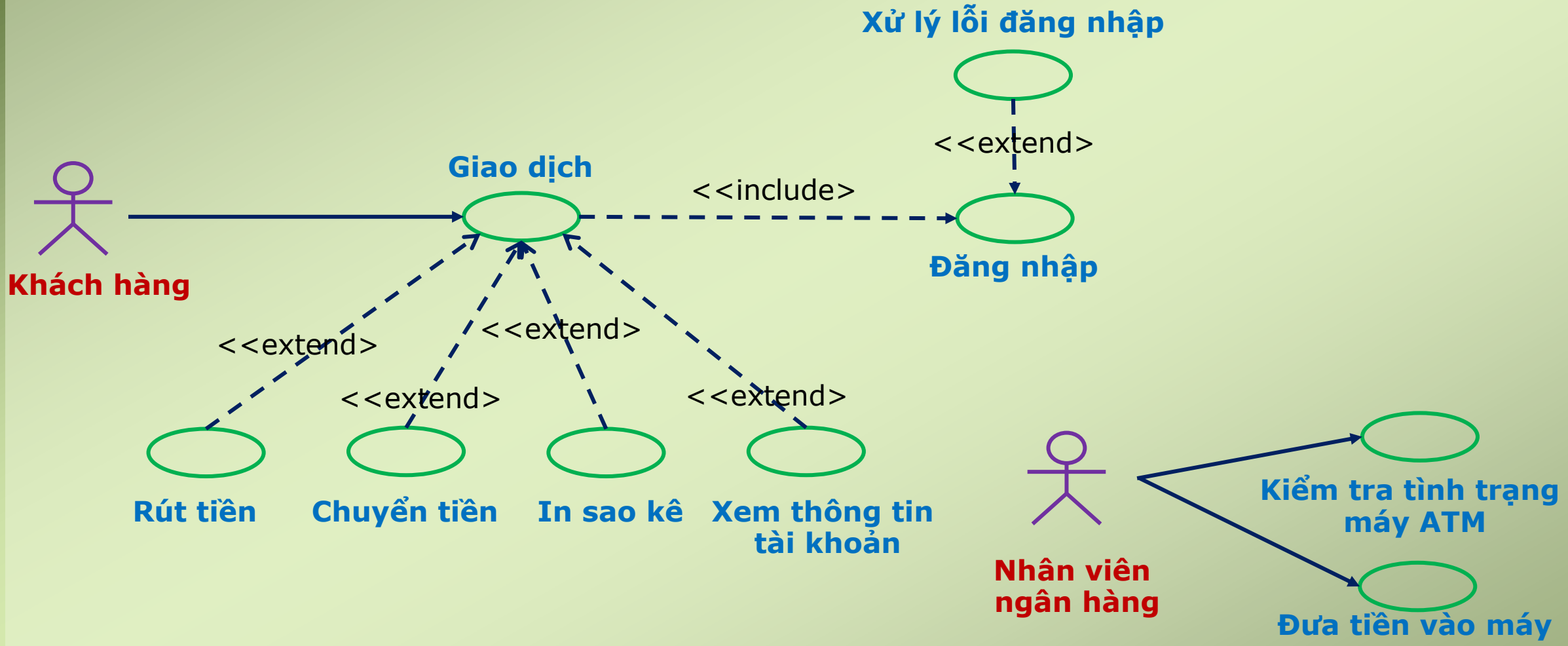


- Chiều của mũi tên thể hiện sử chủ động tương tác

Lược đồ use-case (3/6)

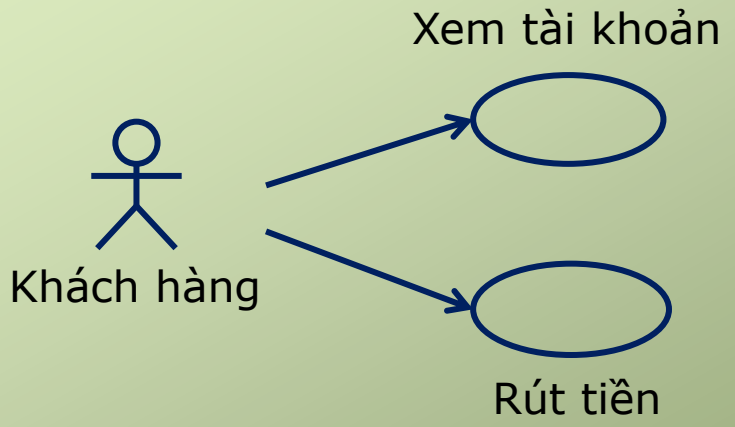


Lược đồ use-case (4/6)



Lược đồ use-case (5/6)

- Mô tả use-case rút tiền
  - Use-case bắt đầu khi người dùng đưa thẻ vào máy ATM, hệ thống đọc và thẩm tra thông tin thẻ.
  - Hệ thống yêu cầu nhập số PIN và kiểm tra sô PIN
  - Hệ thống hỏi tác vụ mà người dùng muốn thực hiện. Người dùng chọn "Rút tiền"
  - Hệ thống yêu cầu chọn hoặc nhập số tiền muốn rút
  - Hệ thống yêu cầu chọn kiểu tài khoản
  - Hệ thông liên lạc ATM network...



Lược đồ use-case (6/6)

- Tại sao sử dụng lược đồ Use-case?
  - Mô hình Use-case là ***kỹ thuật hiệu quả*** để lấy được những yêu cầu cần thiết từ stakeholder và giúp tập trung vào những yêu cầu thực sự.
  - Use-case là một tài liệu ***đơn giản, dễ hiểu, tổng quan chức năng*** hệ thống mà người phân tích hay khách hàng đều có thể dễ dàng đọc và hiểu được. Nó giúp họ có thể chia sẽ ý kiến, trao đổi những gì sản phẩm có thể sẽ làm.
  - Nhóm phát triển có thể dễ dàng hiểu được miền ứng dụng của user, phân chia được cái ***"Needs" từ danh sách "Wants" và "Wishes"*** của stakeholder

Đặc tả yêu cầu (1/4)

- Đặc tả yêu cầu là quá trình **sàn lọc, tinh chế, tổ chức** các yêu cầu thành tài liệu.
- *"A specification can be a written **document**, a **graphical model**, a **formal mathematical model**, a **collection of usage scenarios**, a **prototype**, or a **combination of these**."*

  Roger Pressman
- *"A requirements specification describes the **functions** and **performance of a computer** based system and the **constraints** that will govern its development."*

  Alan Davis

Đặc tả yêu cầu (2/4)

- Các loại tài liệu đặc tả cơ bản
  - Business Requirements Document (BRD)
  - User's Requirements Document (URD)
  - System Requirements Document (SRD)
  - Functional Requirements Document (FRD)
  - Architecture Design Document (ADD)
  - Hardware Requirements Documentt (HRD)
  - **Software Requirements Specification (SRS)**
  - Interfaces Definition Document (IDD)

Đặc tả yêu cầu (3/4)

- Đặc tả yêu cầu phần mềm (Software Requirement Specification: SRS) là các yêu cầu mà nhóm phát triển phần mềm có thể ***thiết kế, lập trình, kiểm chứng các giải pháp phần mềm***.
- SRS chứa tập các ***yêu cầu chức năng ưu tiên và các yêu cầu phi chức năng*** mà sản phẩm phần mềm cần đáp ứng.
- SRS xem như ***hợp đồng giữa khách hàng và tổ chức phát triển phần mềm***.

Đặc tả yêu cầu (4/4)

- Tài liệu SRS phải bao gồm
  - Chuyển từ các mô hình phân tích và các thông tin yêu cầu từ người dùng ***thành các thành phần yêu cầu phần mềm***.
  - ***Kết hợp các mô hình*** phân tích trực tiếp vào đặc tả yêu cầu phần mềm
  - Cung cấp những ***chi tiết cần thiết để thiết kế và viết mã nguồn*** cho sản phẩm.
  - Cung cấp thông tin căn bản ***viết test plan và các thủ tục***.
  - Một cơ sở để tạo ra các sổ tay và tài liệu ***hướng dẫn người dùng***.
  - Chỉ định ***độ ưu tiên*** của các yêu cầu.

Thiết kế sơ đồ lớp (1/9)

+ Thuộc tính/phương thức public

Thuộc tính/phương thức protected

– Thuộc tính/phương thức private

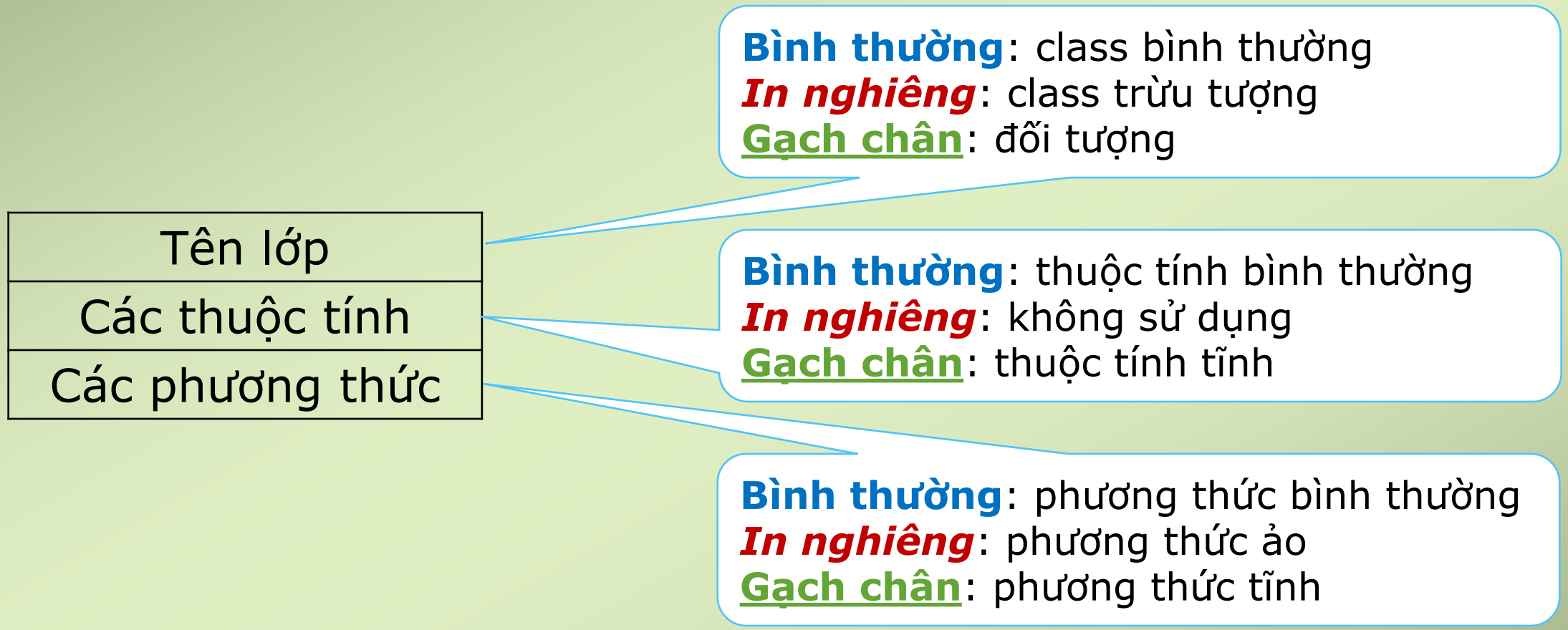
Tên lớp
Các thuộc tính
Các phương thức

Tên lớp
-<thuộc tính private> #<thuộc tính protected>
+<phương thức public>() #<phương thức protected>() -<phương thức private>()

Thiết kế sơ đồ lớp (2/9)

Sinh viên
- Họ tên - Giới tính - Năm sinh - Quê quán
+ Tra cứu điểm + Đăng ký học phần + Xem lịch thi + Xem lịch học

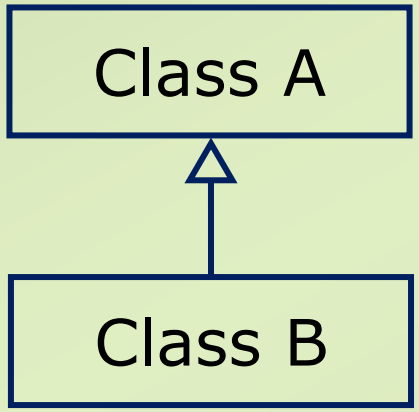
Thiết kế sơ đồ lớp (3/9)



Thiết kế sơ đồ lớp (4/9)

- Quan hệ giữa các lớp đối tượng
  - Quan hệ kế thừa (inheritance)
  - Quan hệ association
  - Quan hệ aggregation
  - Quan hệ composition
  - Quan hệ dependency

Thiết kế sơ đồ lớp (5/9)



Quan hệ kế thừa

- Class A kế thừa từ Class B
- Class B là trường hợp đặc biệt của Class A
- Class A là trường hợp tổng quát của Class B

Thiết kế sơ đồ lớp (6/9)

Quan hệ association

- Hoặc Class A có thuộc tính kiểu Class B
- Hoặc Class B có thuộc tính kiểu Class A

Thiết kế sơ đồ lớp (7/9)

Quan hệ aggregation

- Trong đối tượng của Class A chứa đối tượng của Class B, nghĩa là trong danh sách thuộc tính của Class A có thuộc tính kiểu class B
- Nếu đối tượng a của Class A bị hủy thì đối tượng b của Class B (bên trong a) ***vẫn có thể tồn tại***.

Thiết kế sơ đồ lớp (8/9)

Quan hệ composition

- Trong đối tượng của Class A chứa đối tượng của Class B, nghĩa là trong danh sách thuộc tính của Class A có thuộc tính kiểu class B
- Nếu đối tượng a của Class A bị hủy thì đối tượng b của Class B (bên trong a) ***không thể còn tòn tại***.

Thiết kế sơ đồ lớp (9/9)

Quan hệ dependency

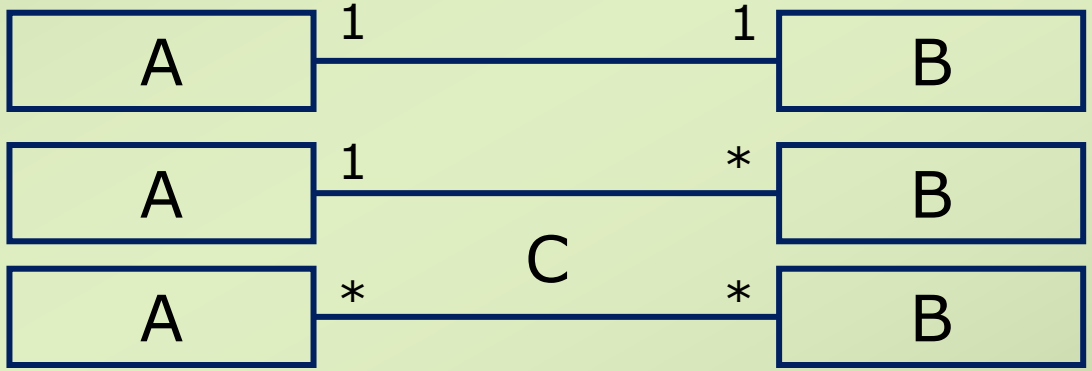
- Class A và Class B không có quan hệ association
- Đối tượng kiểu class B có thể là
  - Đối số trong một phương thức của Class A
  - Kết quả trả về trong một phương thức của Class A
  - Biến cục bộ trong một phương thức của Class A

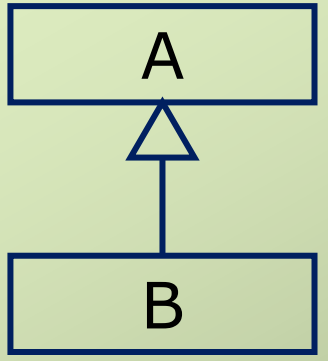
Thiết kế dữ liệu (1/3)

- Mục tiêu của thiết kê dữ liệu là nhằm mô tả cách thức lưu trữ dữ liệu của phần mềm bên trong máy tính.
- Kết quả của quá trình thiết kế dữ liệu
  - Danh sách các bảng (table)
  - Mối quan hệ giữa các bảng (relation)
  - Thông tin các thuộc tính, khóa của các bảng

Thiết kế cơ sở dữ liệu (2/3)

- Mỗi lớp đối tượng được ánh xạ thành một bảng
- Quan hệ 1 – 1
- Quan hệ 1 – n
- Quan hệ n – n
- Quan hệ kế thừa





Thiết kế cơ sở dữ liệu (3/3)

- Nếu lớp đối tượng có thuộc tính có cấu trúc phức tạp thì tách thành một bảng lưu cấu trúc cho thuộc tính đó.
- Nếu lớp đối tượng có thuộc tính kiểu mảng thì tách thành bảng chỉ tiết.
- Thuộc tính có giá trị rời rạc thì tách thành bảng danh mục

Q&A