Chương 2: Phương pháp phân tích thiết kế hệ thống

Bộ môn Hệ thống thông tin

Nội dung chương 2

1. Giới thiệu
2. Quy trình phát triển phần mềm
3. PP hướng chức năng và hướng đối tượng
4. Ưu điểm của phương pháp hướng đối tượng
5. Các hệ quản trị cơ sở dữ liệu phân tán

Giới thiệu

- Việc xây dựng phần mềm cần quan tâm đến tổ chức của hệ thống, cấu trúc và mối quan hệ các thành phần tạo nên các hoạt động trong hệ thống.
- Một sản phẩm phần mềm cần đảm bảo được các tiêu chí đánh giá:
  - Tính tiện dụng
  - Khả năng bảo hành và duy trì hoạt động
  - Tính tin cậy
  - Tính hiệu quả

Giới thiệu(tt)

- Nghiên cứu hệ thống cần thiết để:
  - Hiểu rõ hơn, nhất là những hệ thống phức tạp
  - Hoàn thiện hay phát triển hệ thống tốt hơn hiệu quả hơn nhằm đáp ứng yêu cầu đặt ra với hệ thống
- Các khía cạnh cần đề cập:
  - Các bước thực hiện trong thời kỳ phát triển hệ thống hay còn gọi là quy trình phát triển hệ thống
  - Các phương pháp nhận thức và diễn tả hệ thống, mô hình hóa hệ thống

Quy trình phát triển phần mềm

- Phát triển phần mềm là bài toán phức tạp, lý do:
  - Người phát triển phần mềm khó hiểu cho đúng những gì người dùng yêu cầu
  - Yêu cầu người dùng thường thay đổi trong thời gian phát triển, đôi khi được mô tả bằng văn bản dài dòng khó hiểu, thậm chí nhiều khi mâu thuẩn
  - Khả năng nắm bắt các dữ liệu phức tạp của con người là có hạn
  - Khó định lượng được hiệu suất của thành phẩm và thỏa mãn mong chờ của người dùng

=>Cần xây dựng hệ thống phần mềm theo quy trình.

Các mô hình chu trình phát triển phần mềm

Bất kể dùng theo phương pháp gì thì chu kỳ phát triển hệ thống nói chung gồm 4 công đoạn cơ bản sau:

1. Lập kế hoạch
2. Phân tích
3. Thiết kế
4. Cài đặt

Lập kế hoạch

- Khảo sát tổng thể hệ thống.
- Xác định phạm vi, nguồn lực, các nguyên tắc làm việc.
- Đánh giá khả thi.
- Xây dựng tài liệu mô tả hệ thống.

Phân tích hệ thống

- Xác định yêu cầu hệ thống
- Cấu trúc hóa yêu cầu: mô hình hoá và phân tích yêu cầu có thể dùng
  - Phương pháp phân tích hướng chức năng
  - Phương pháp hướng đối tượng
- Phát sinh các phương án hệ thống và chọn lựa phương án khả thi nhất

Thiết kế hệ thống

- Thiết kế luận lý
  - Thiết kế dữ liệu
  - Thiết kế kiến trúc
  - Thiết kế giao diện
- Thiết kế vật lý
  - Chuyển đổi thiết kế luận lý sang các đặc tả phần cứng, phần mềm, kỹ thuật được chọn để cài đặt hệ thống

Cài đặt, kiểm thử chương trình

- Lập trình cài đặt
  - Lập trình hệ thống
  - Thử nghiệm
  - Xây dựng tài liệu hệ thống: tài liệu đặc tả hệ thống, tài liệu sử dụng, tài liệu kỹ thuật cài đặt

Bảo trì hệ thống

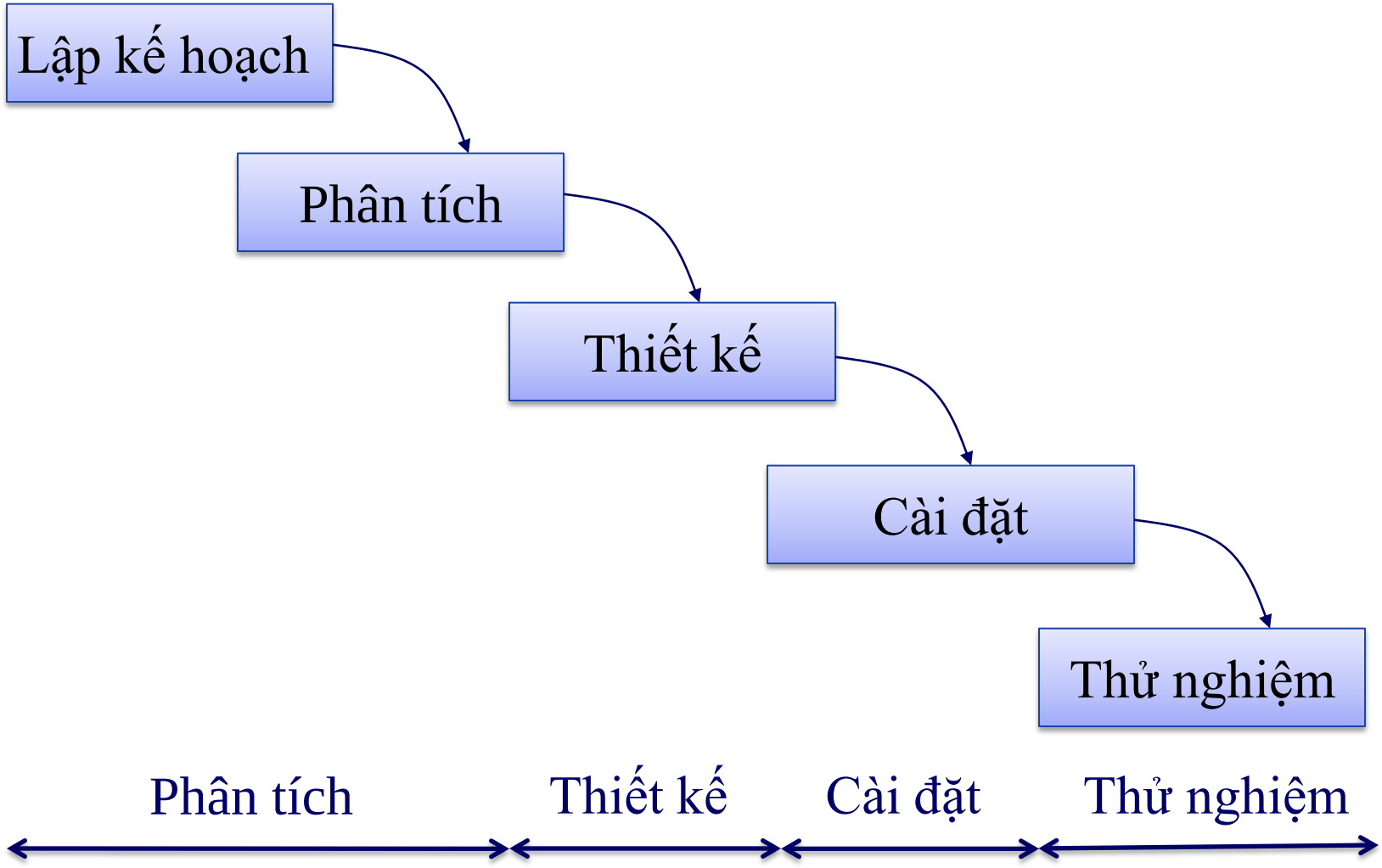
- Bảo trì
  - Fix các lỗi phát sinh trong quá trình sử dụng
  - Điều chỉnh những thay đổi sao cho phù hợp với các thay đổi hệ thống
  - Nâng cấp hệ thống mới

Quy trình phát triển phần mềm

- Quy trình thác nước
- Quy trình phát triển song song
- Quy trình phát triển theo giai đoạn
- Phương pháp sử dụng bản mẫu

Qui trình thác nước

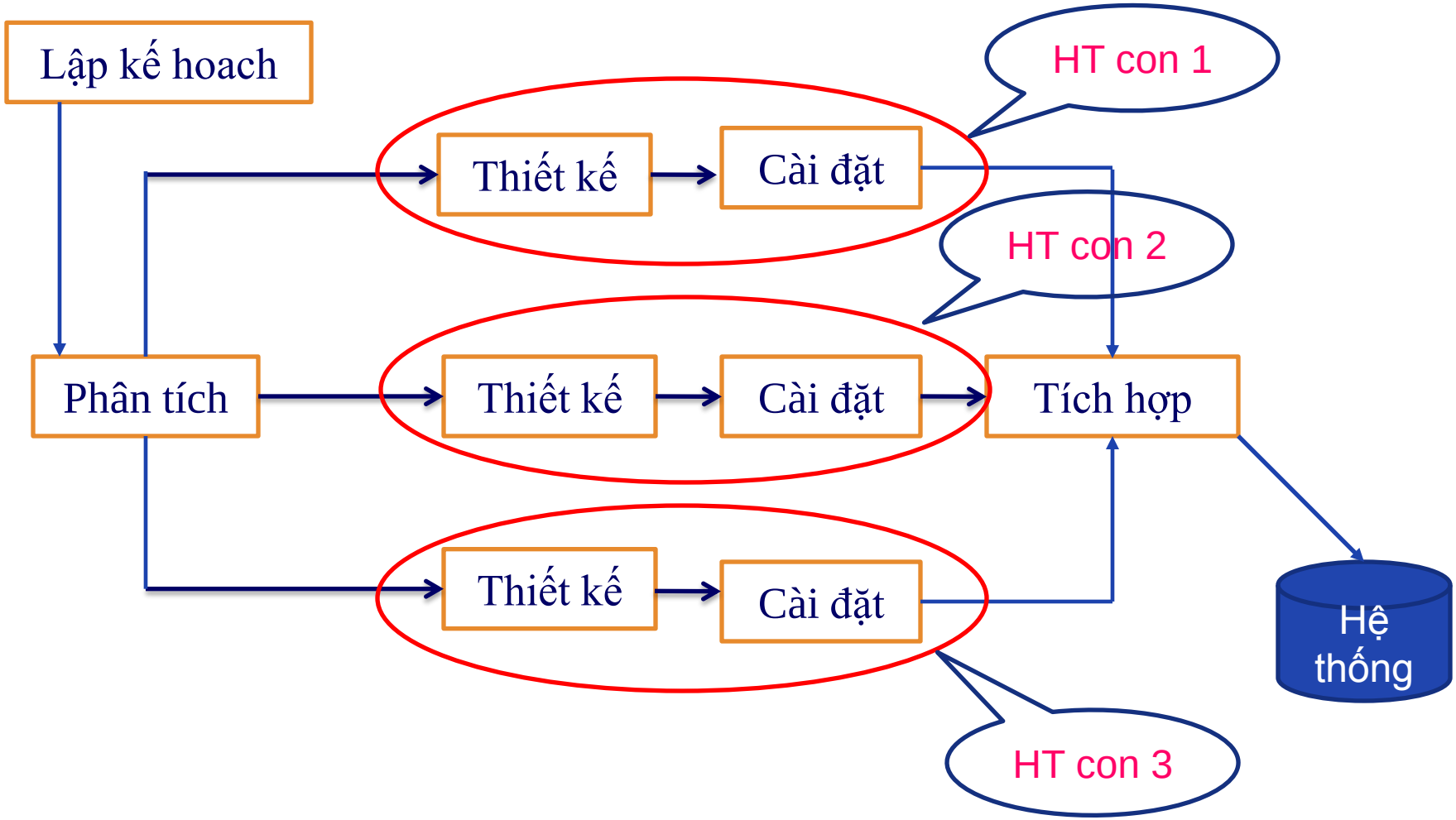
- **Được đưa ra bởi waterfall-Boehm 1970**



Quy trình thác nước (tt)

- Thuận lợi:
  - Do phải xác định xong yêu cầu trước khi bắt đầu lập trình →giảm thiểu các thay đổi về yêu cầu khi xúc tiến dự án.
- Khó khăn:
  - Thiết kế phải được hoàn tất trước khi lập trình và mất rất nhiều thời gian đến lúc chính thức bàn giao hệ thống cho người dùng.
  - Khi giai đoạn trước có sự thay đổi (do sai sót, do nhu cầu người dùng thay đổi, có sự tiến hóa hệ thống → gây khó khăn.

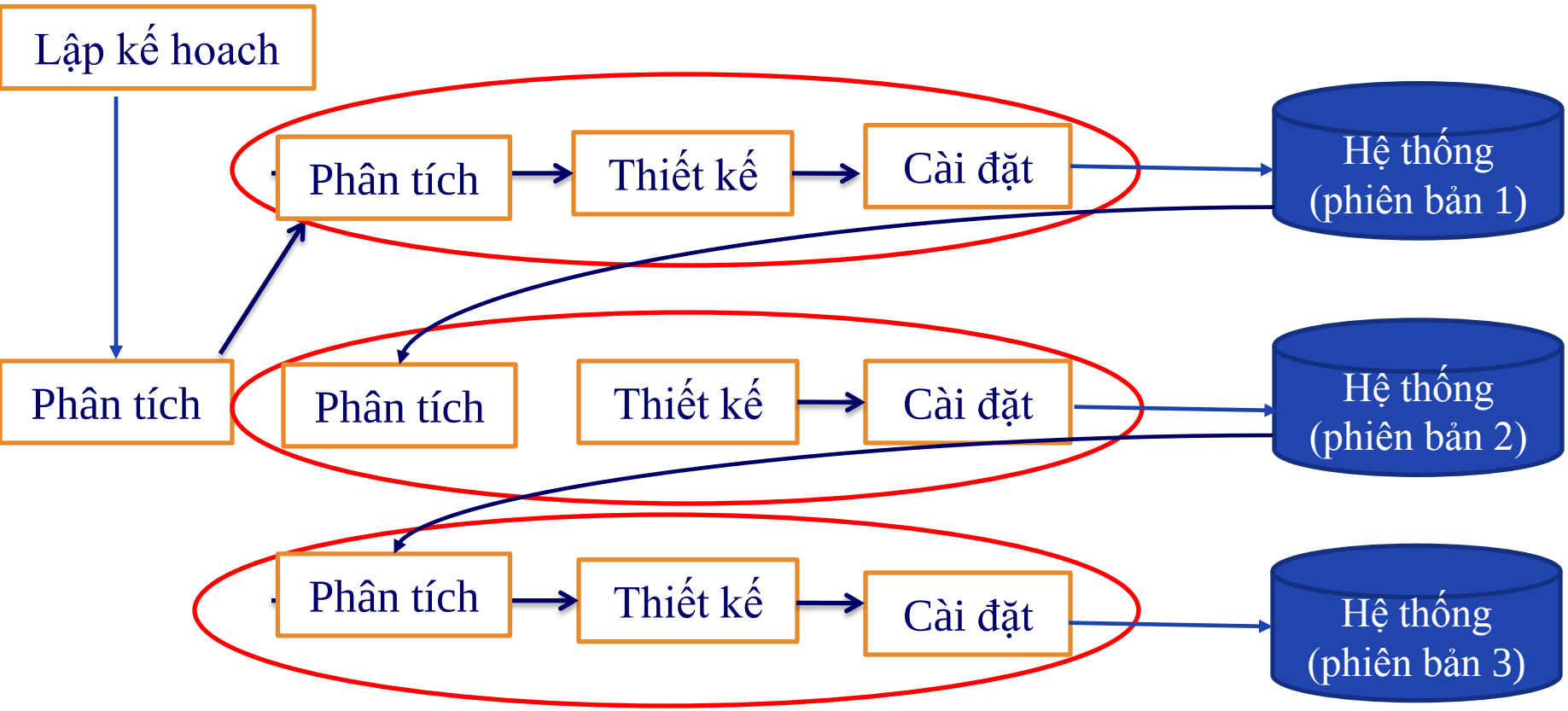
Quy trình phát triển song song



Quy trình phát triển song song

- Thuận lợi:
  - Giảm thời gian chuyển giao hệ thống
- Khó khăn:
  - Nhiều tài liệu giấy tờ đặc tả hệ thống
  - Các hệ thống con không hoàn toàn độc lập với nhau, hệ thống này ảnh hưởng hệ thống khác, việc tích hợp sẽ tốn kém.

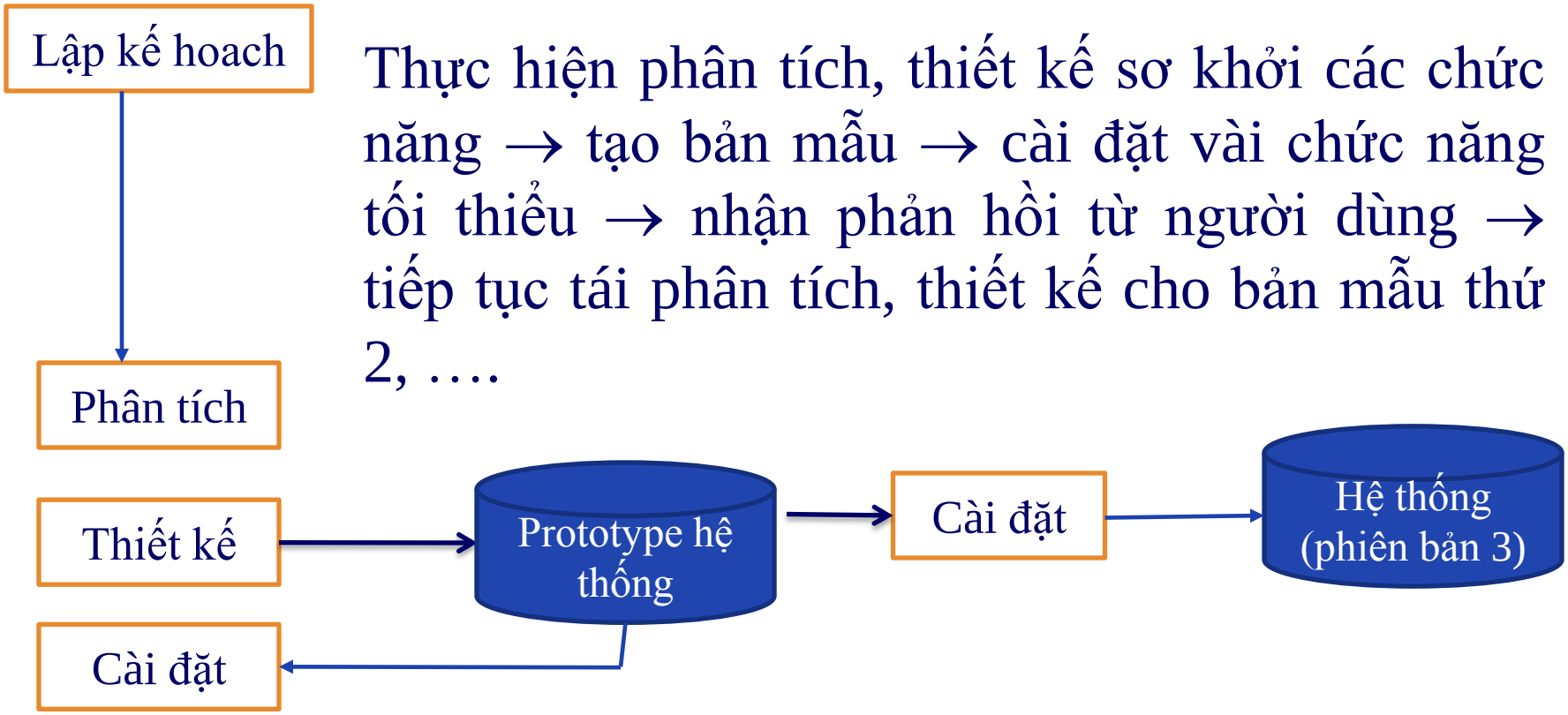
Quy trình phát triển theo giai đoạn



Quy trình phát triển theo giai đoạn

- Thuận lợi:
  - Nhanh chóng chuyển giao hệ thống vào tay người dùng (mặc dù chưa thực hiện đầy đủ hết các chức năng). Từ đó xác định các yêu cầu quan trọng bổ sung sớm hơn.
- Khó khăn:
  - Người dùng bắt đầu làm việc với hệ thống không đầy đủ
  - Xác định các yêu cầu quan trọng để đưa vào phiên bản đầu tiên đôi khi khó khắn.

Phương pháp sử dụng bản mẫu



Thực hiện phân tích, thiết kế sơ khởi các chức năng → tạo bản mẫu → cài đặt vài chức năng tối thiểu → nhận phản hồi từ người dùng → tiếp tục tái phân tích, thiết kế cho bản mẫu thứ 2, ….

Phương pháp sử dụng bản mẫu

- Thuận lợi:
  - Nhanh chóng cung cấp hệ thống cho người dùng ngay cả khi chưa sẵn sang áp dụng trên toàn đơn vị
  - Tạo sự tin tưởng về tiến độ cho người dùng
  - Cung cấp cho người dùng bản mẫu trực quan để hiểu những gì hệ thống làm được.
- Khó khăn:
  - Cần phải phân tích thiết kế cẩn thận.
  - Trong các hệ thống phức tạp, các vấn đề phức tạp thường được phát hiện thông qua cài đặt.

Mô hình hóa hệ thống

- **Mô hình** là một dạng biểu diễn trừu tượng của một hệ thống thực, được diễn tả:
  - Ở một mức độ trừu tượng hoá nào đó,
  - Theo một góc nhìn nào đó,
  - Bởi một hình thức diễn tả hiểu được (chẳng hạn văn bản, đồ thị)
- Diễn tả hệ thống bằng mô hình (bao gồm cả khi phân tích và khi thiết kế) được gọi là **mô hình hoá.**

Tại sao cần tạo mô hình?

- Mô hình giúp chúng ta hình dung được hệ thống như thế nào.
- Mô hình cho phép xác định được cấu trúc và hành vi của hệ thống.
- Mô hình giúp chúng ta xây dựng hệ thống theo các mẫu.
- Mô hình lưu trữ lại các quyết định trong lúc xây dựng hệ thống.

Các mức mô hình hóa

- Quá trình mô hình hóa thực hiện theo 2 cấp:
  - Mô hình logic: mô tả các thành phần và mối quan hệ của chúng để tổ chức thực hiện. Trả lời câu hỏi: "là gì ?" và bỏ qua câu hỏi "như thế nào?".
  - Mô hình vật lý: xác định kiến trúc thành phần và tổng thể của hệ thống. Trả lời câu hỏi: "như thế nào?", quan tâm tới biện pháp, công cụ, kế hoạch thực hiện

Phương pháp phát triển hệ thống

- Có hai cách tiếp cận cơ bản để phát triển hệ thống:
  - Cách tiếp cận hướng chức năng.
  - Cách tiếp cận hướng đối tượng.

Phương pháp hướng chức năng

- Phương pháp truyền thống của ngành CNPM.
- Phần mềm như là tập hợp các chương trình (chức năng) và dữ liệu giả lập.

  Chương trình = thuật giải + cấu trúc

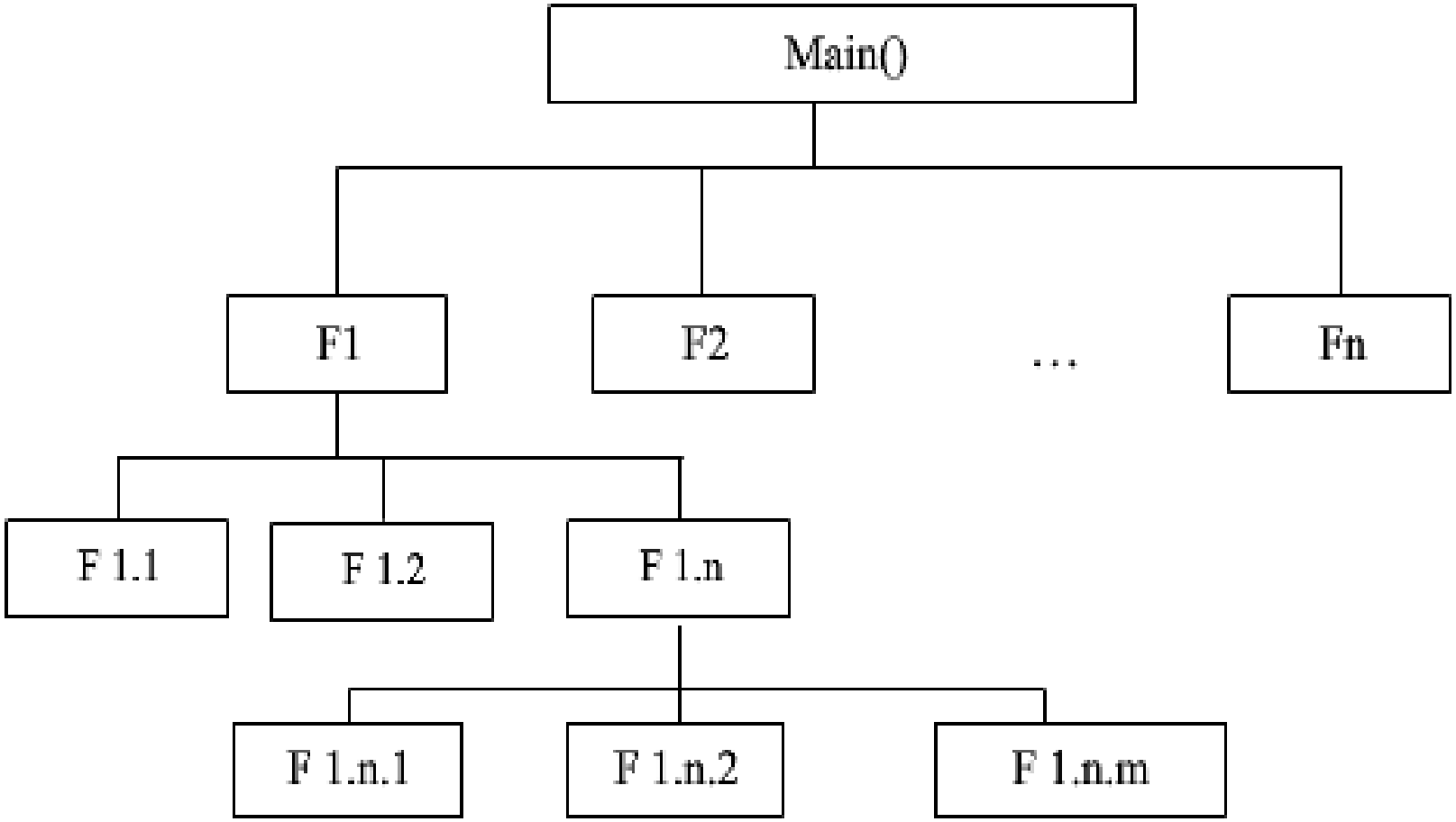
- Theo cách tiếp cận này thì chương trình được xem là một dãy các công việc.
- Ví dụ: "Hệ thống quản lý thư viện": quản lý bạn đọc, cho mượn sách, nhận trả sách, thông báo nhắc trả sách…(gần như không thay đổi).

Phương pháp hướng chức năng(tt)

- Mỗi công việc sẽ được thực hiện bởi một số hàm nhất định.
- Trọng tâm của cách tiếp cận này là các hàm chức năng.
- Phần mềm được phân tích ra thành các chức năng nhỏ hơn. (đến khi nào thì dừng?) =>(phân rã đến cấp hàm trong ngôn ngữ lập trình – có thể lập trình được).

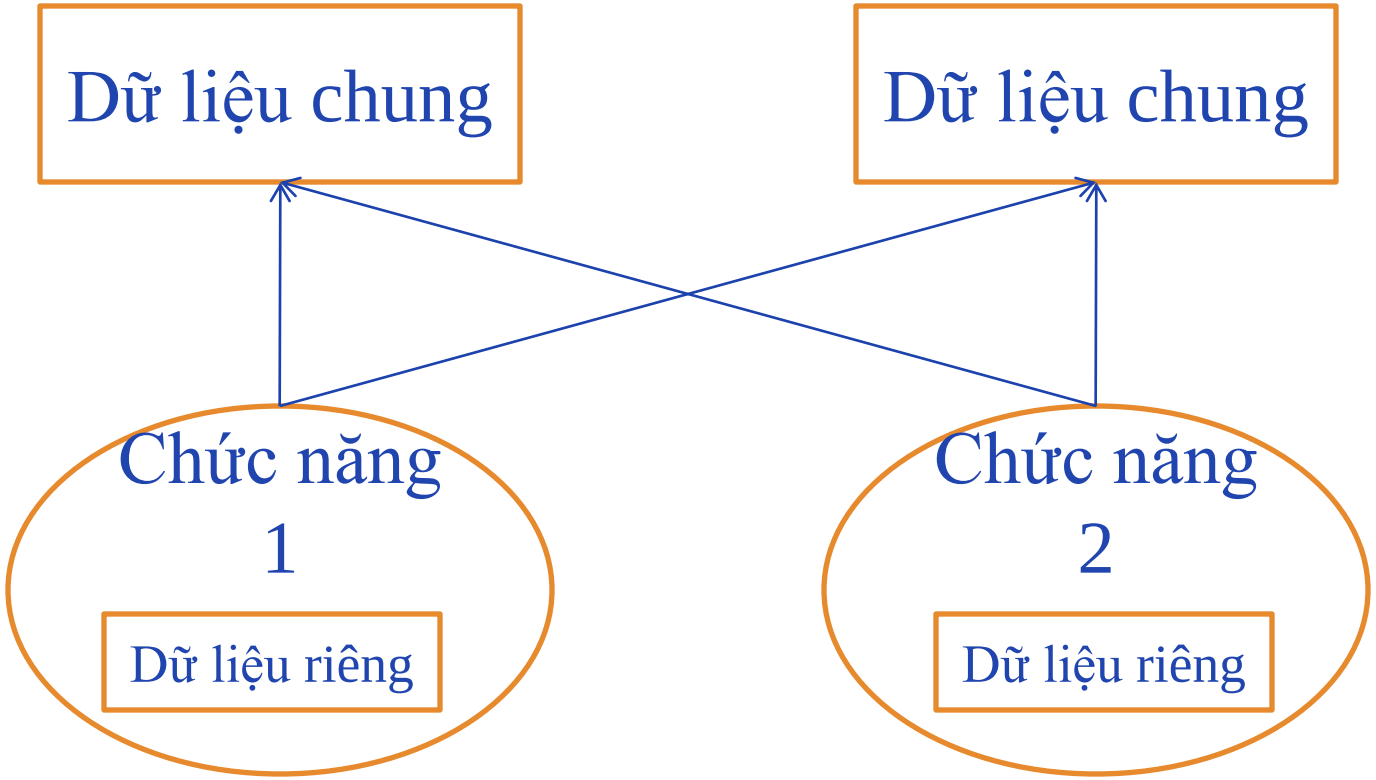
Phương pháp hướng chức năng(tt)

Mô hình PTTK hướng chức năng



Phương pháp hướng chức năng(tt)

- Các chức năng trao đổi với nhau bằng cách truyền tham số hoặc sử dụng dữ liệu chung.



Mô hình quan hệ giữa các chức năng trong hệ thống

Phương pháp hướng chức năng(tt)

- Sử dụng biến toàn cục → bất lợi trong thiết kế và lập trình.
- Dự án lớn, phức tạp, nhiều nhóm tham gia →sự thay đổi dữ liệu chung sẽ ảnh hưởng đến hiệu suất lao động, dự án.

❖Tính mở và tính thích nghi của hệ thống thấp. Vấn đề nâng cấp và duy trì hệ thống phần mềm là khó khăn.

❖Khả năng tái sử dụng hạn chế

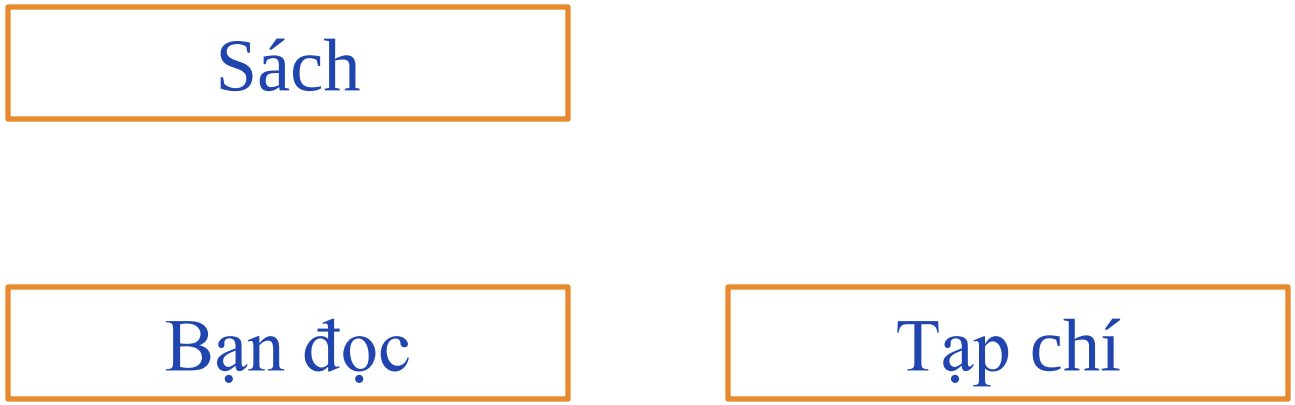
❖ Không hỗ trợ cơ chế kế thừa.

Phương pháp hướng đối tượng

- Cả hệ thống được coi như một thực thể được tổ chức từ tập các đối tượng (thực thể) và các đối tượng đó trao đổi với nhau thông qua việc gửi và nhận (phương thức).

Phương pháp hướng đối tượng (tt)

- Phân tích hệ thống thành các đơn thể đơn giản, dễ hiểu.
- Ví dụ: "Hệ thống quản lý thư viện", ta có các lớp đối tượng sau:



Ưu điểm của PP hướng đối tượng

- Các chức năng của hệ thống được biểu diễn thông qua cộng tác của đối tượng → việc tiến hóa thay đổi các chức năng không ảnh hưởng đến cấu trúc tĩnh của phần mềm.
- Tính mở và tính thích nghi của hệ thống cao hơn (chỉ thay đổi những lớp đối tượng có liên quan, hoặc bổ sung lớp đối tượng).
- Khả năng tái sử dụng cao (sử dụng cơ chế kế thừa).

Ngôn ngữ mô hình hóa UML

Tổng quan về UML

- UML được xây dựng dựa vào:
  - Cách tiếp cận của Booch
  - Kỹ thuật mô hình hướng đối tượng OMT (Object Modeling Technique) của Rumbaugh
  - Công nghệ phần mềm hướng đối tượng OOSE (Object Oriented Software Engineering) của Jacobson

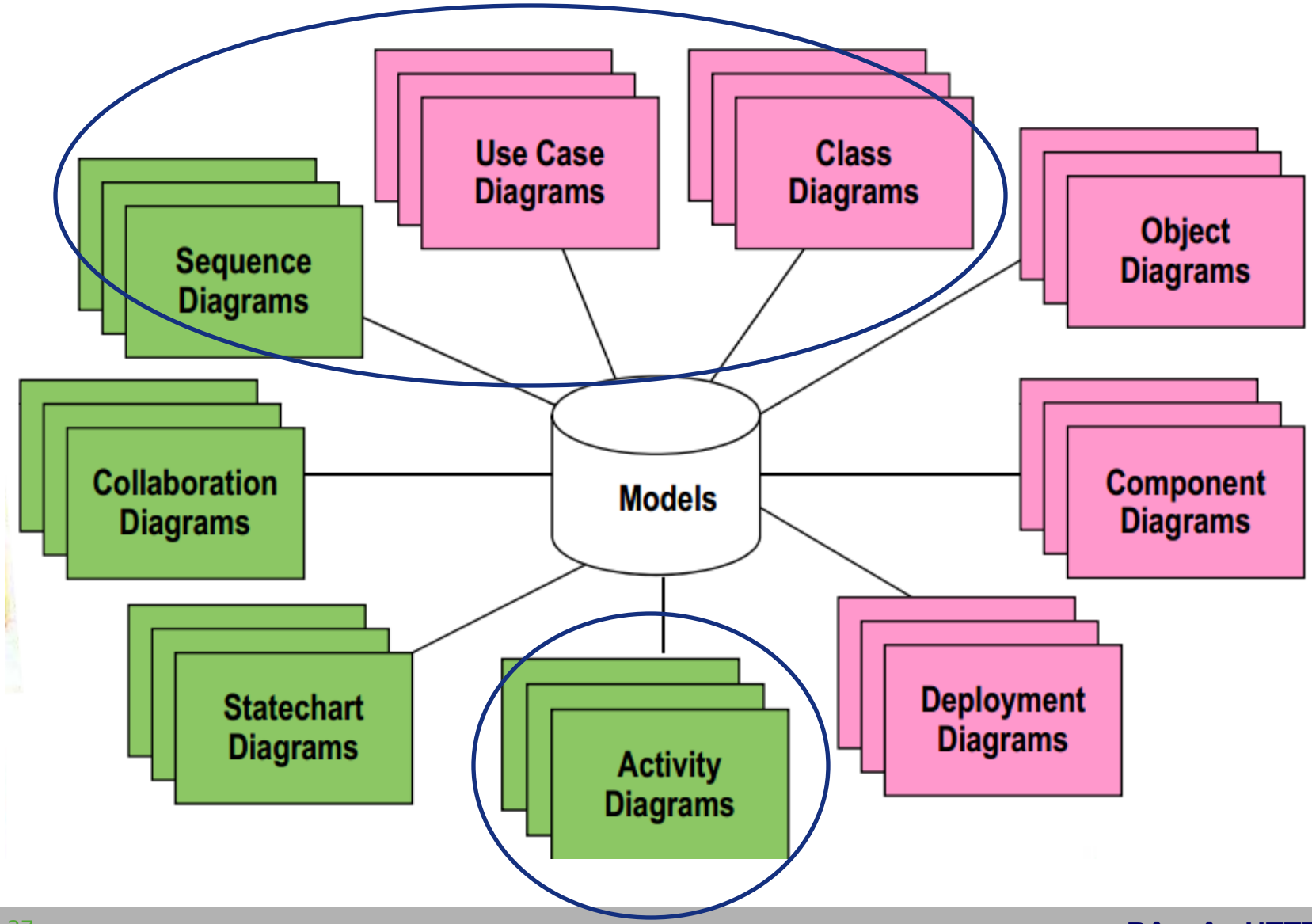
Tổng quan về UML (tt)

- UML là một ngôn ngữ mô hình hóa hướng đối tượng (modeling language)
- UML là ngôn ngữ để viết kế hoạch chi tiết phần mềm → phù hợp cho việc mô hình hóa các hệ thống.

Tổng quan về UML (tt)

- Các mô hình xây dựng bởi UML có thể ánh xạ tới một ngôn ngữ lập trình cụ thể như : Java, C++, VB... thậm chí cả các bảng trong một CSDL quan hệ hay CSDL hướng đối tượng.

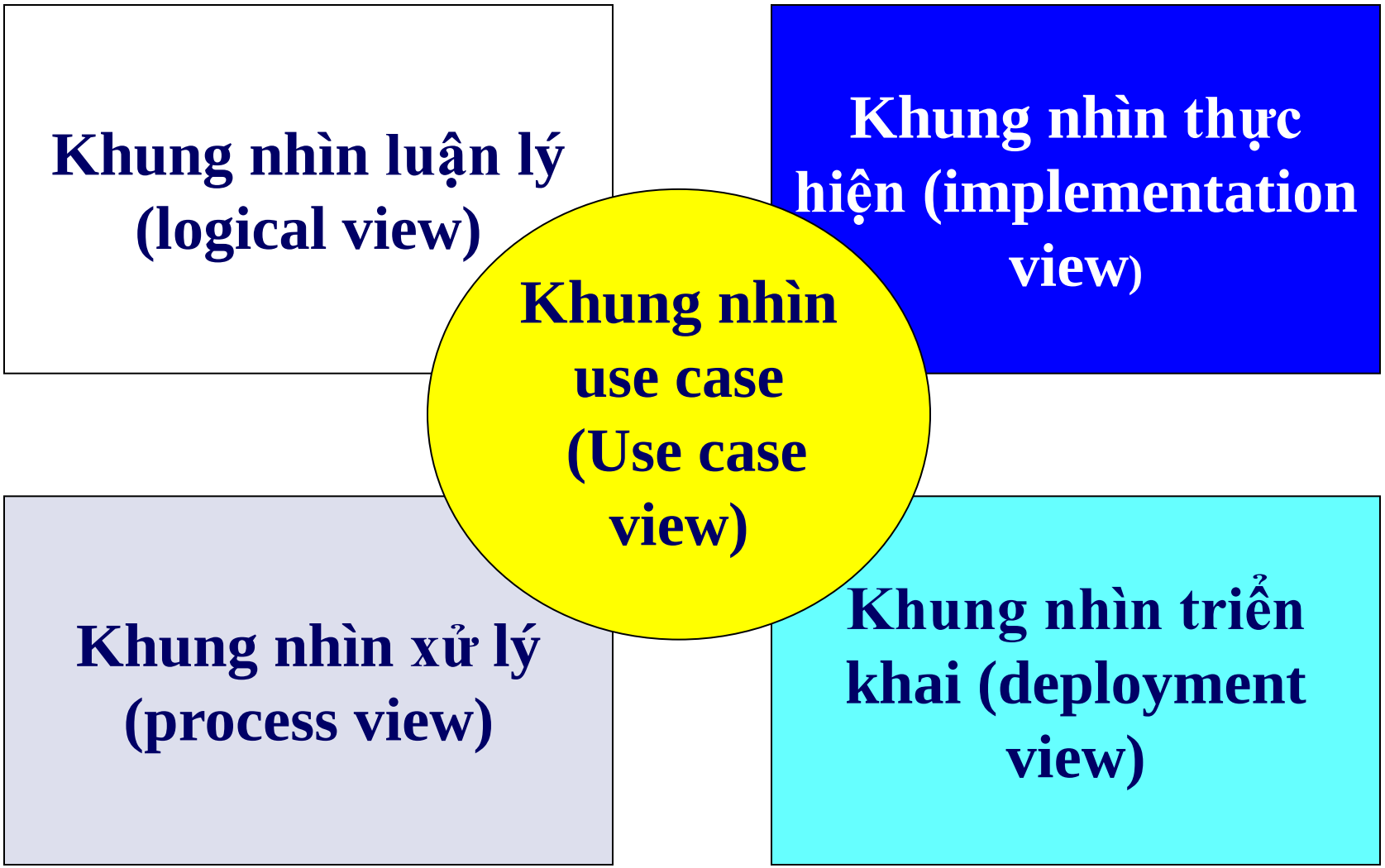
Biểu đồ UML


Use Case Diagrams
Class Diagrams
Sequence Diagrams
Object Diagrams
Collaboration Diagrams
Models
Component Diagrams
Statechart Diagrams
Activity Diagrams
Deployment Diagrams

Các biểu đồ trong UML

- Biểu đồ cấu trúc
  - Object diagram (Biểu đồ đối tượng)
  - Class diagram (Biểu đồ lớp)
  - Component diagram (Biểu đồ thành phần)
  - Deployment diagram (Biểu đồ triển khai)
- Biểu đồ hành vi
  - Use case diagram (Biểu đồ use case)
  - Sequence diagram (Biểu đồ trình tự)
  - Collaboration diagram (Biểu đồ hợp tác)
  - State Diagram (Biểu đồ trạng thái)
  - Activity diagram (Biểu đồ hoạt động)

Các khung nhìn (view) của UML



Khung nhìn Use case

- Chứa các use case mô tả hành vi của hệ thống dưới góc nhìn của người dùng cuối, nhà phân tích hay người kiểm thử hệ thống.
- Là góc nhìn từ ngoài vào hệ thống, không xét tổ chức bên trong của phần mềm, mà chỉ làm rõ các chức năng chính của hệ thống
- Khi bắt đầu dự án, lược đồ use case được dùng để thống nhất hệ thống giữa khách hàng và nhà phát triển hệ thống

Khung nhìn lý luận, thiết kế

- Là cách nhìn của những nhà thiết kế hệ thống.
- Thể hiện tĩnh qua các biểu đồ lớp, biểu đồ đối tượng và thể hiện động qua biểu đồ tương tác, biểu đồ máy trạng thái, biểu đồ hoạt động
- Để tạo khung nhìn thiết kế thường theo hai bước.
  - Bước 1: nhận ra các lớp phân tích (analysis class) độc lập với ngôn ngữ lập trình
  - Bước 2: chuyển các lớp phân tích thành các lớp thiết kế (design class) phụ thuộc theo ngôn ngữ.

Khung nhìn xử lý

- Chia hệ thống thành các tiến trình (process) và luồng (thread), mô tả việc đồng bộ hóa và các xử lý đồng thời.
- Phản ánh các lộ trình điều khiển, các quá trình thực hiện
- Được thể hiện cùng với các biểu đồ như góc nhìn thiết kế

Khung nhìn thực hiện

- Là khung nhìn đối với dạng phát hành của phần mềm (hệ thống vật lý).
- Bao gồm các component và file tương đối độc lập, có thể lắp ráp theo nhiều cách để hệ thống chạy được.
- Với UML sắc thái tĩnh của khung nhìn này thể hiện qua biểu đồ thành phần, sắc thái động thể hiện qua biểu đồ tương tác, máy trạng thái, biểu đồ hoạt động

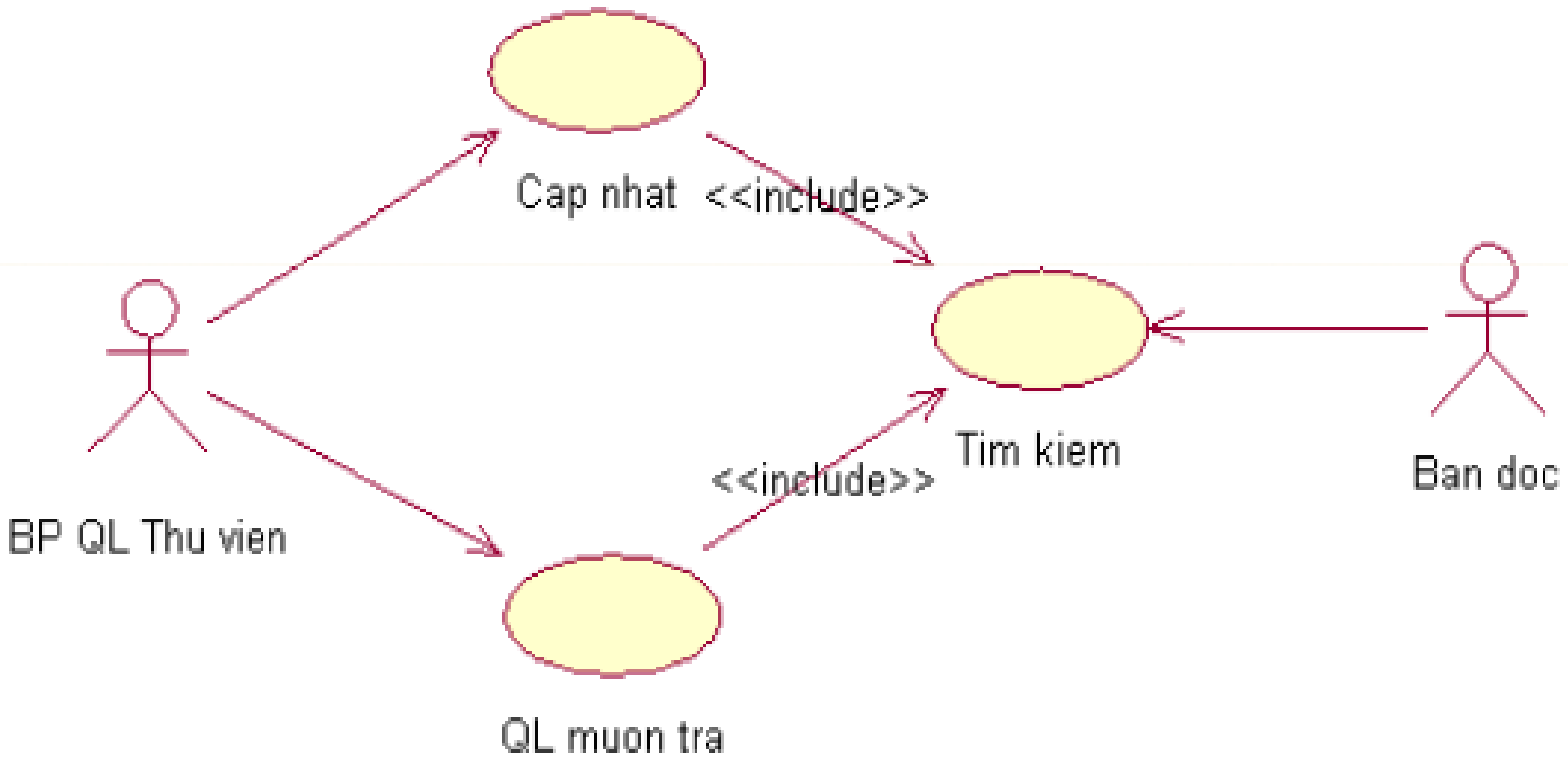
Khung nhìn triển khai

- Chỉ ra cấu hình phần cứng mà hệ thống sẽ chạy trên đó. Nó thể hiện sự phân tán, cài đặt các phần tạo nên kiến trúc vật lý của hệ thống. Biểu đồ được sử dụng là Deployment Diagram.
- Có thể vận dụng 5 góc nhìn trên 1 cách tách biệt tùy theo sự quan tâm của từng loại người đến với hệ thống, tuy vậy các khung nhìn trên phải có sự tương hợp

Biểu đồ use case

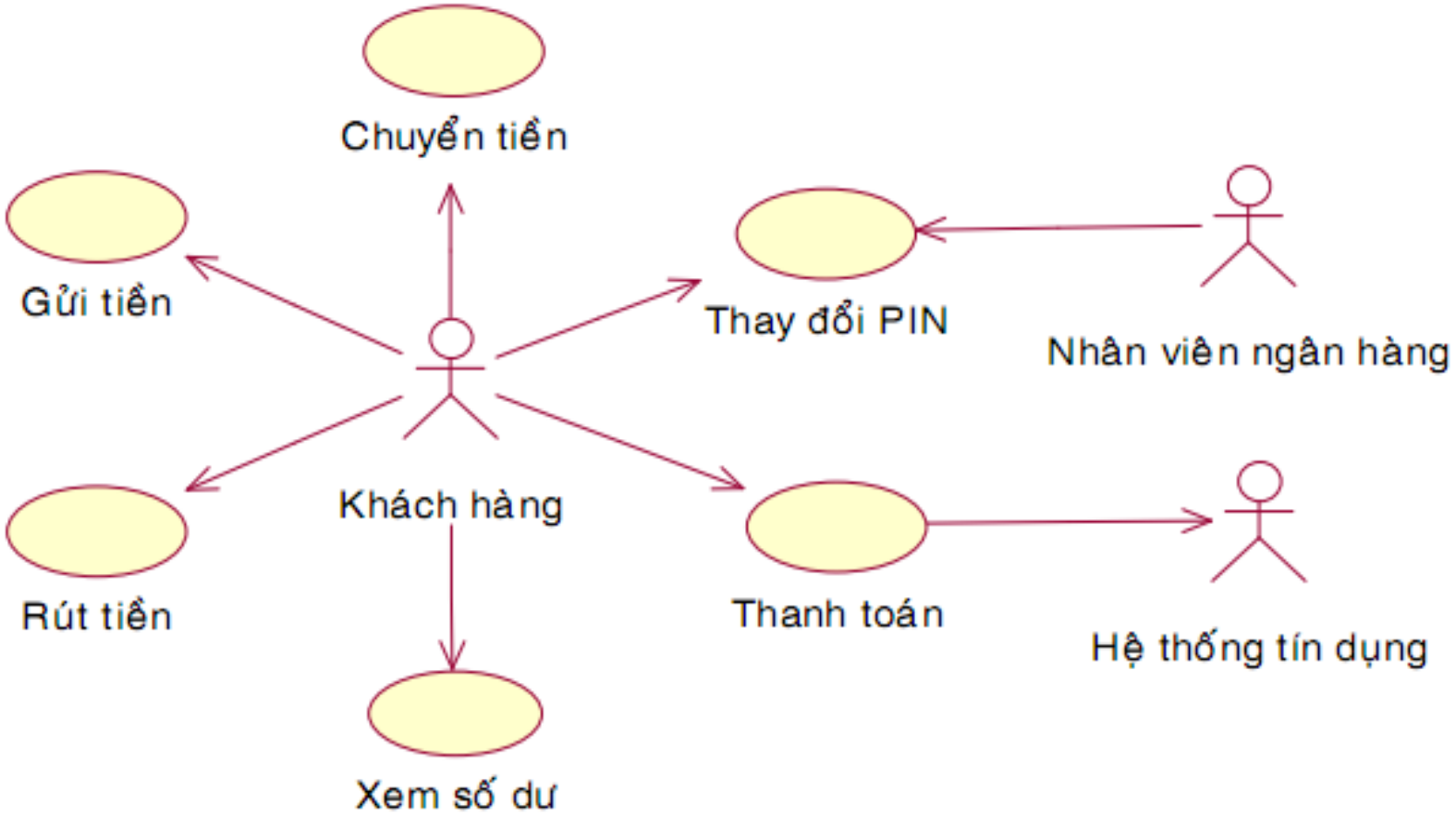
- Một biểu đồ Use case (Use case diagram) mô tả các tình huống tiêu biểu của hệ thống.
- Mỗi use case mô tả chức năng hệ thống cần phải xem xét từ quan điểm của người sử dụng.
- Tác nhân (actor): là con người hay hệ thống thực khác cung cấp thông tin hay tác động đến hệ thống.
- Biểu đồ use case: là tập hợp các tác nhân, các use case và mối quan hệ giữa chúng.

Biểu đồ use case (tt)



Biểu đồ UC Quản lý mượn trả sách

Biểu đồ use case (tt)

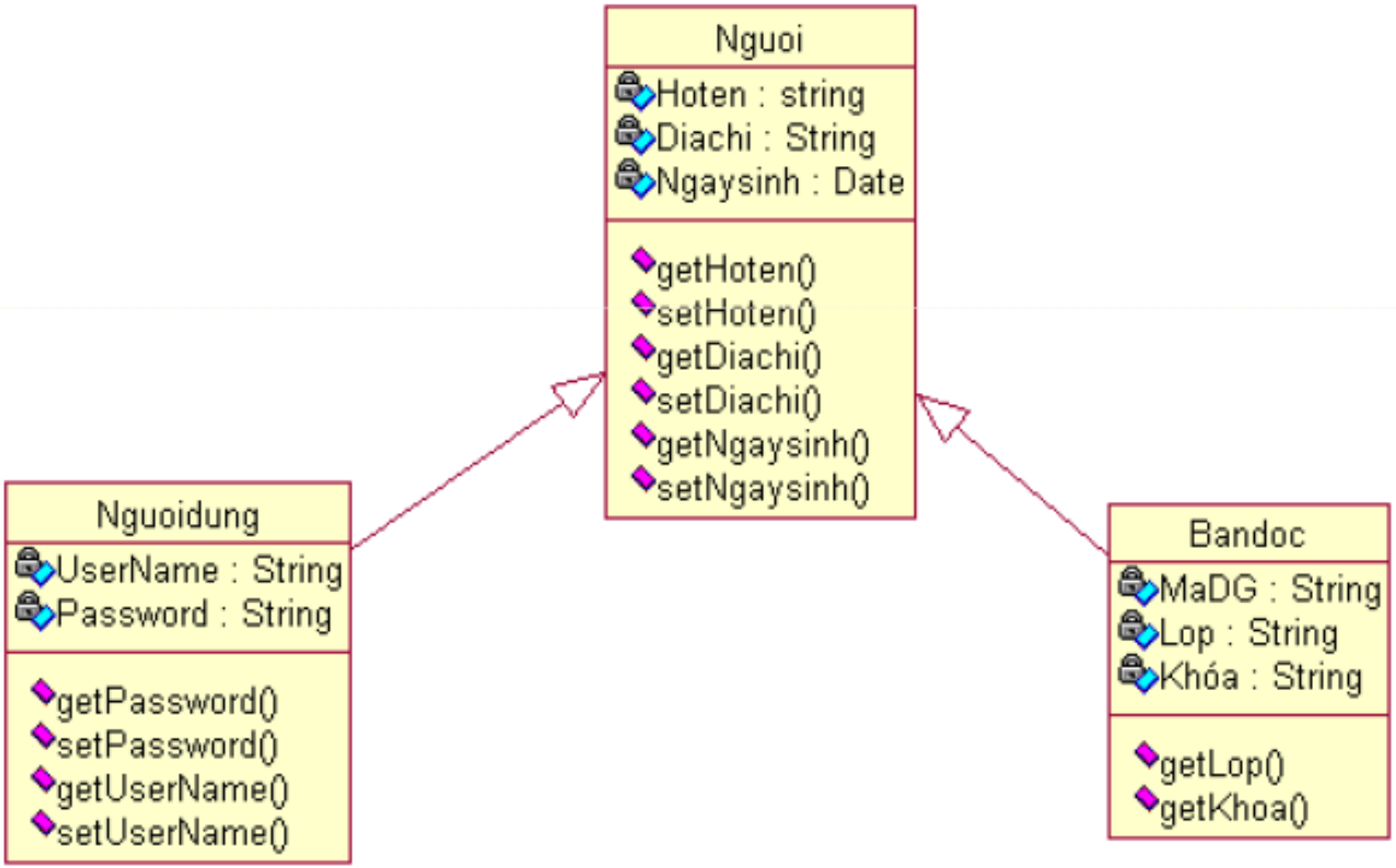


Biểu đồ UC của ATM

Biểu đồ lớp

- Biểu đồ lớp (Class diagram) là một nhóm đối tượng có chung một số thuộc tính và phương thức tạo thành.
- Biểu đồ lớp bao gồm các lớp (thuộc tính và phương thức) và mối quan hệ giữa chúng

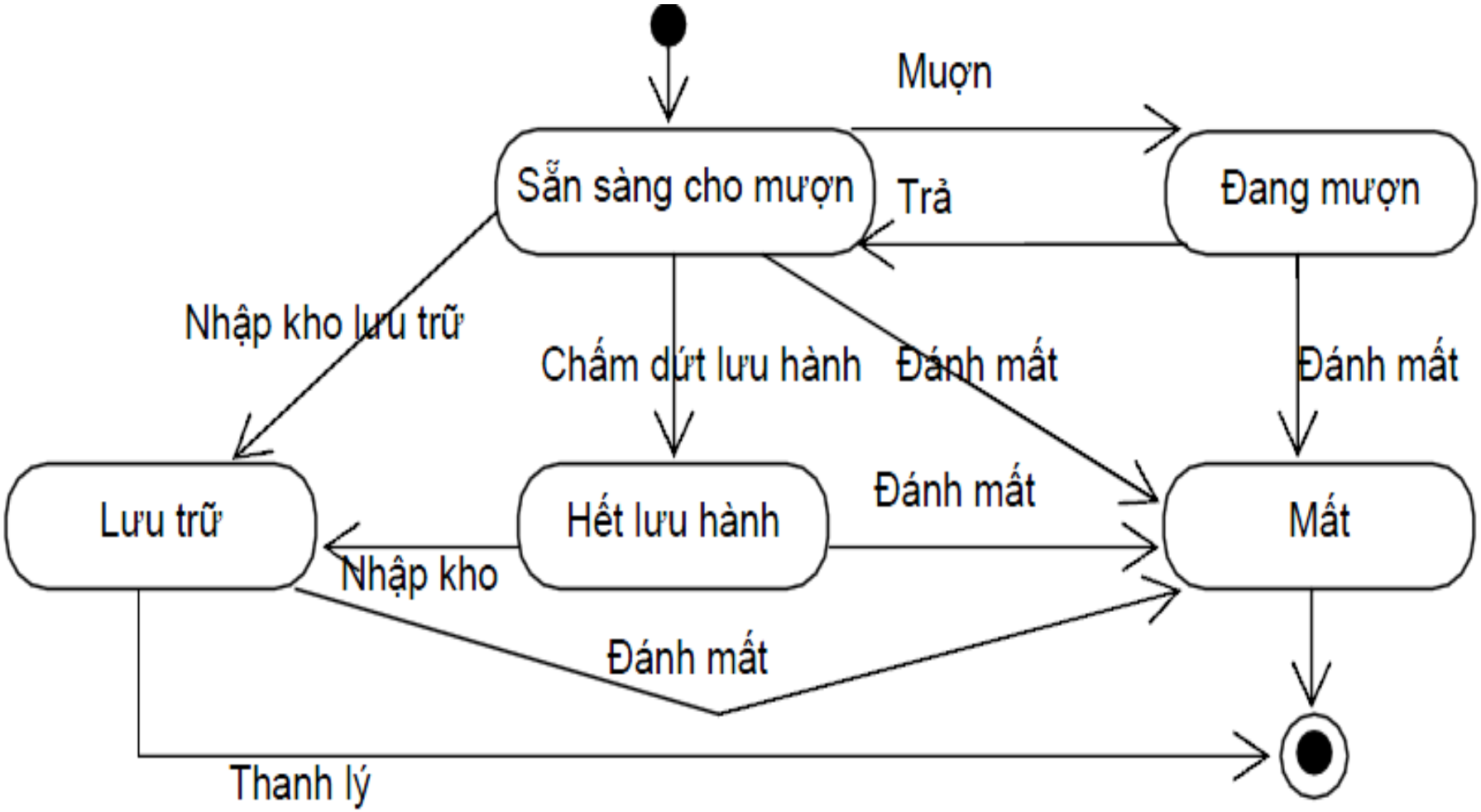
Biểu đồ lớp (tt)



Biểu đồ trạng thái

- Biểu đồ trạng thái (State Diagram) Tương ứng với mỗi lớp sẽ chỉ ra các trạng thái mà đối tượng của lớp có thể có và sự chuyển tiếp giữa những trạng thái đó.

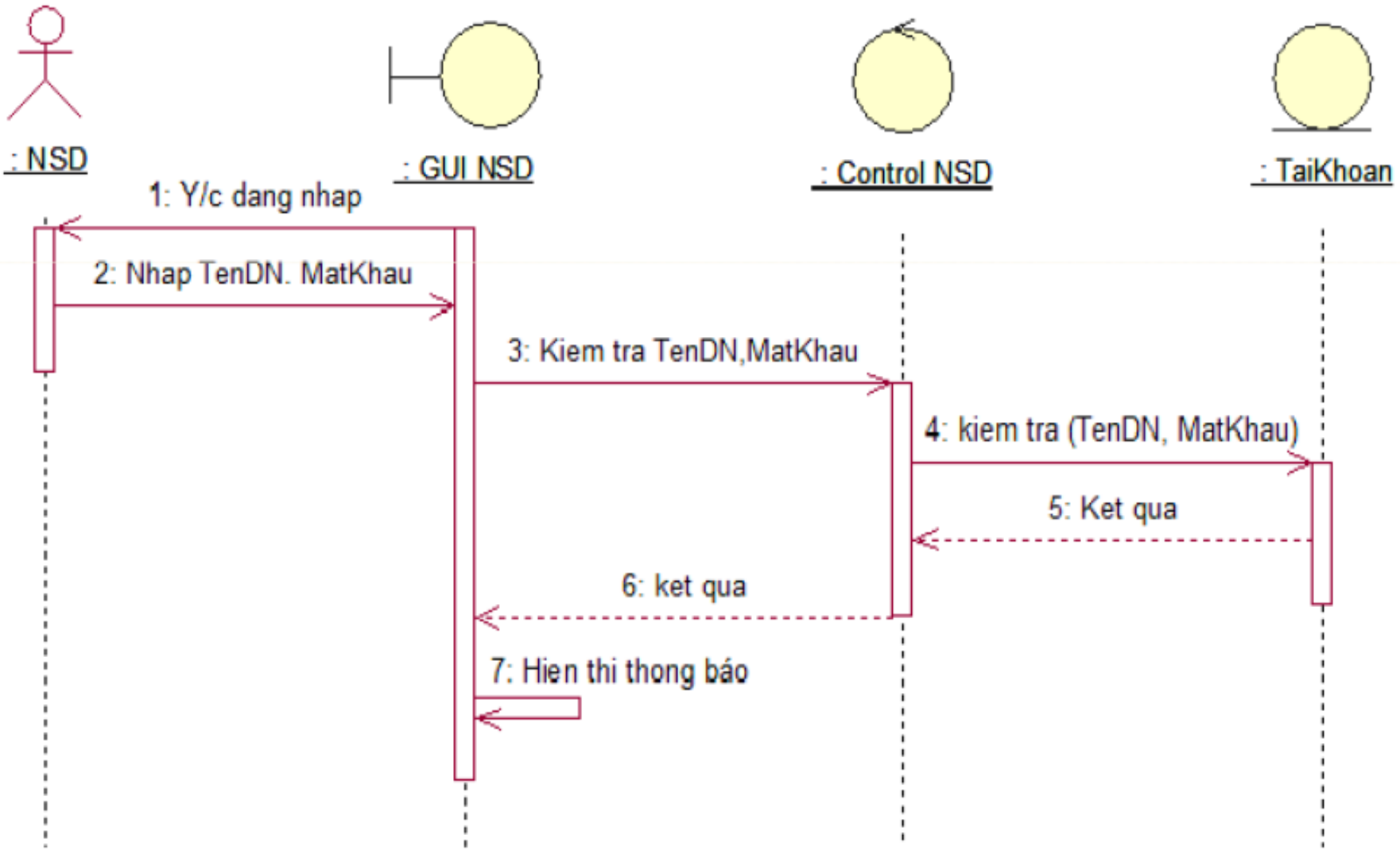
Biểu đồ trạng thái



Biểu đồ trình tự, cộng tác

- Biểu đồ trình tự (Sequence diagram) là biểu đồ cộng tác (Collaboration diagram) biểu diễn mối quan hệ giữa các đối tượng. Biểu đồ trình tự thêm vào chiều thời gian nhằm thể hiện trực quan thứ tự trao đổi của các thông điệp.

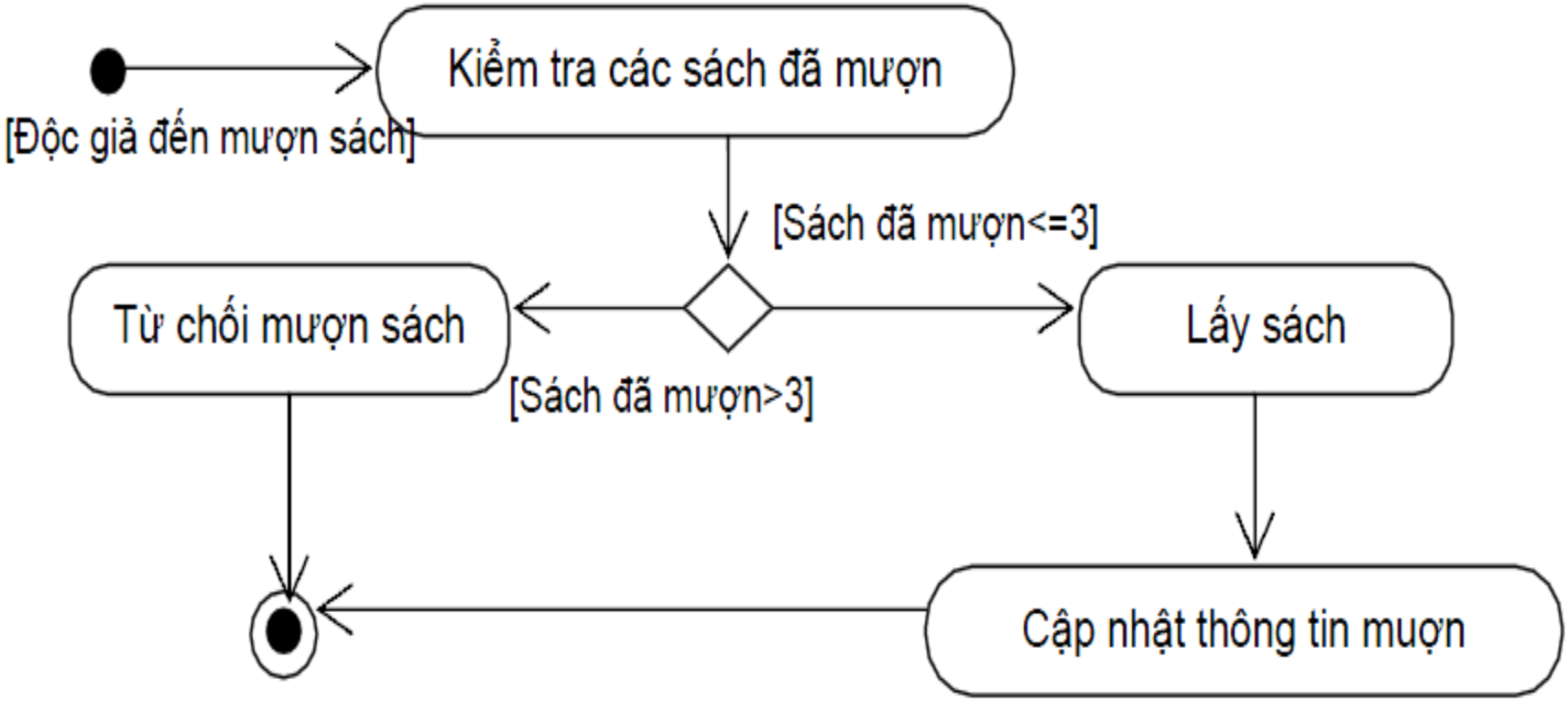
Biểu đồ trình tự (tt)



Biểu đồ hoạt động

- Biểu đồ hoạt động (Activity diagram) biểu diễn các hoạt động, chuyển tiếp các hoạt động, thường được sử dụng để biểu diễn các phương thức phức tạp của lớp hoặc mô tả ca sử dụng.

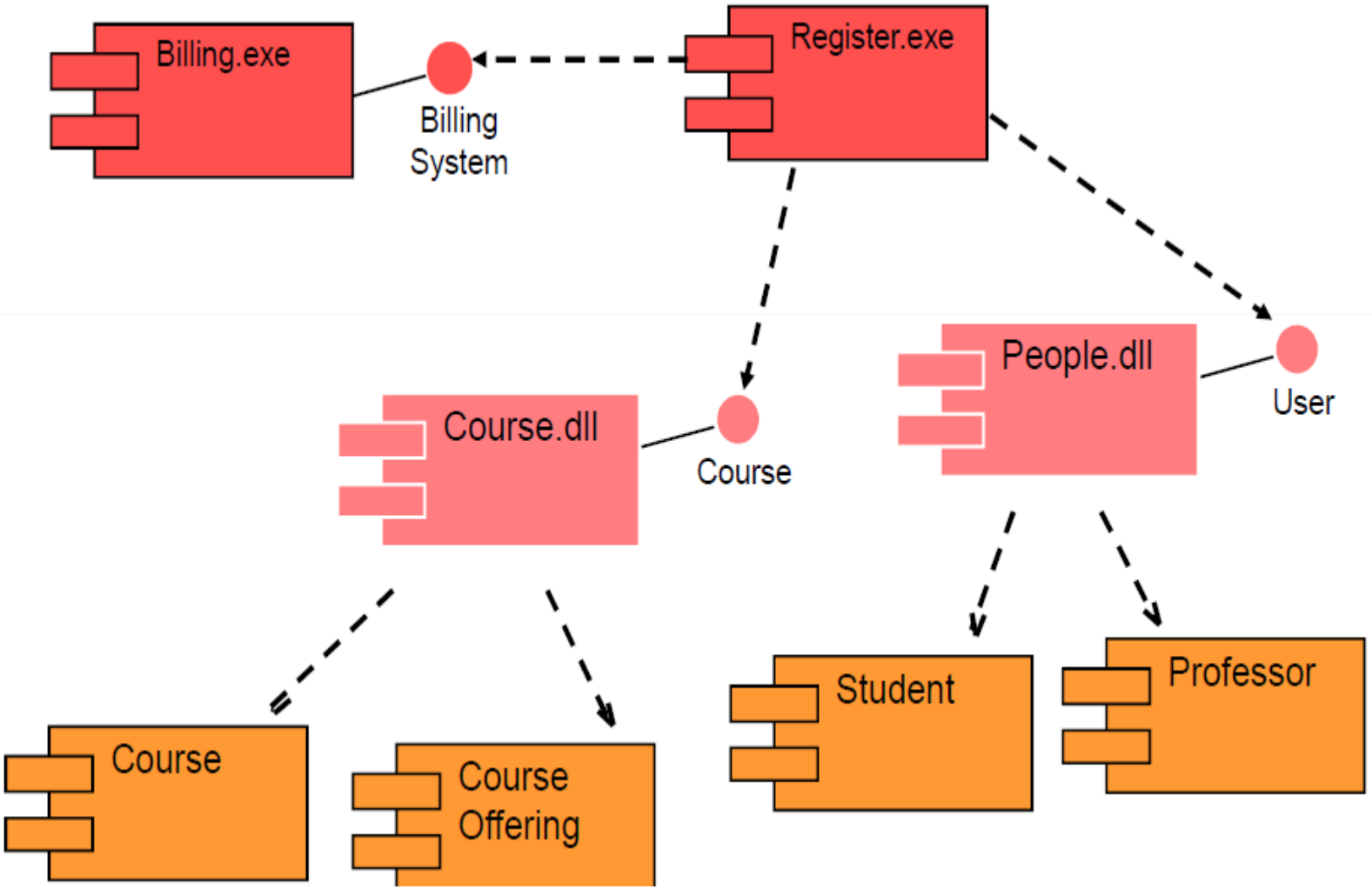
Biểu đồ hoạt động



Biểu đồ thành phần

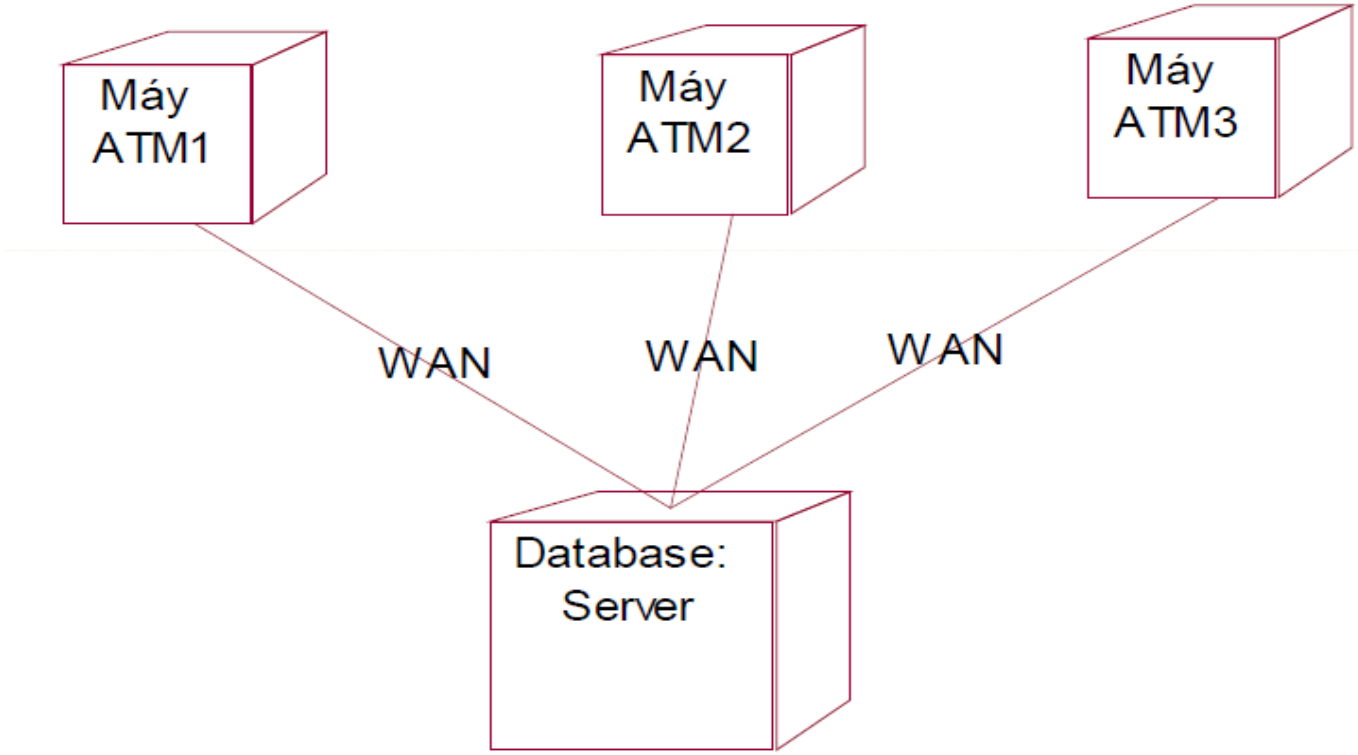
- Biểu đồ thành phần (Component diagram) biểu diễn các thành phần của hệ thống và mối liên hệ giữa các thành phần đó.
- Mục đích
  - Tổ chức mã nguồn.
  - Xây dựng các module thi hành.
  - Chỉ định cơ sở dữ liệu vật lý.
- Được phát triển bởi nhà thiết kế và lập trình.

Biểu đồ thành phần



Biểu đồ triển khai

- Biểu đồ triển khai (Deployment diagram) biểu diễn khía cạnh vật lý bằng việc mô tả các nút vật lý và các mối quan hệ giữa chúng.



Các khái niệm cơ bản của phương pháp hướng đối tượng trong UML

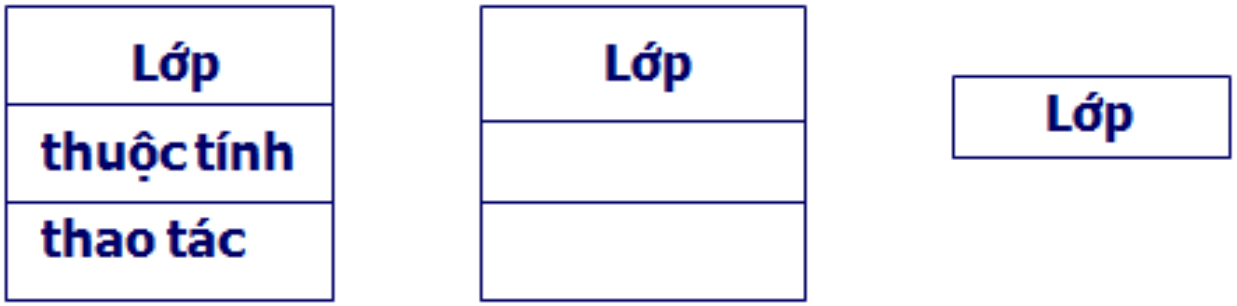
- Đối tượng.
- Lớp đối tượng.
- Thuộc tính của đối tượng.
- Thao tác và phương thức.

Đối tượng

- Đối tượng (Object): là một khái niệm, môt sự trừu tượng hóa hay một sự vật có nghĩa trong bài toán đang khảo sát.
- Biểu diễn đối tượng

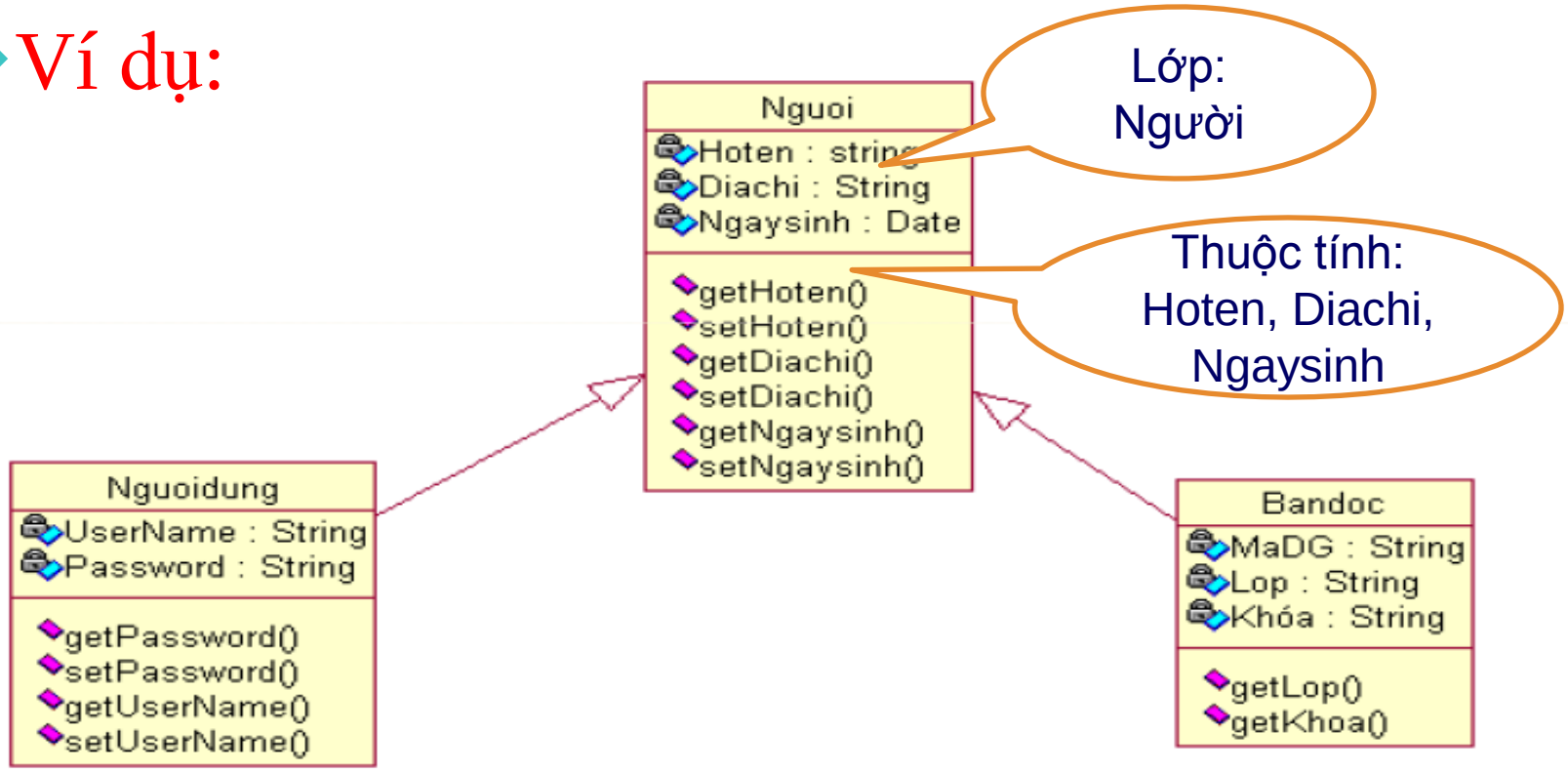
Lớp đối tượng

- Lớp đối tượng (Class): là một mô tả về một nhóm các đối tượng có những tính chất giống nhau, có chung các hành vi ứng xử (thao tác gần như nhau), có cùng mối liên quan của các đối tượng của các lớp khác và có chung ngữ nghĩa trong hệ thống phần mềm.
- Biểu diễn lớp đối tượng:



Thuộc tính của đối tượng

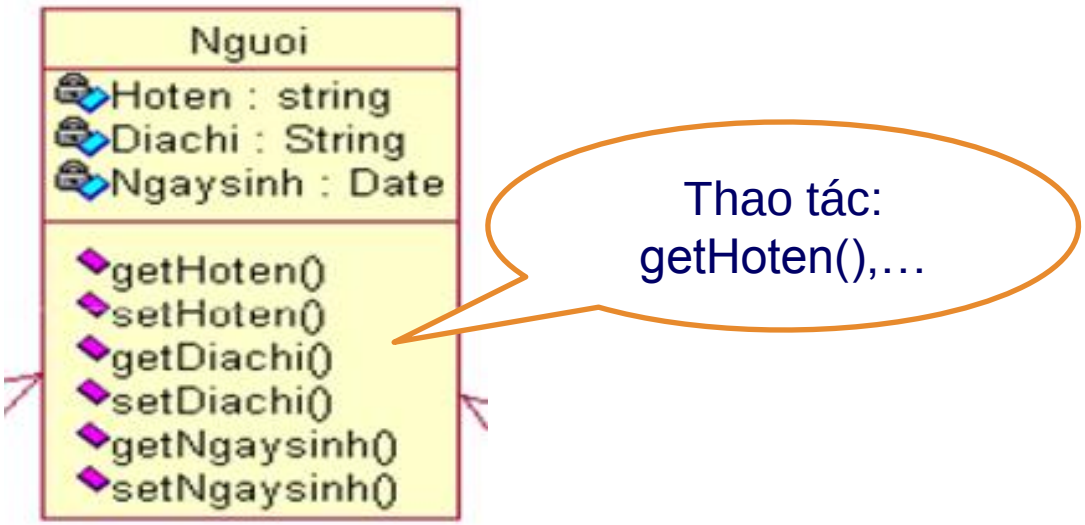
- *Thuộc tính* là một tính chất đặc trưng có tên của một lớp và nó nhận một giá trị cho mỗi đối tượng thuộc lớp đó tại mỗi thời điểm.
- Ví dụ:



Thao tác và phương thức

- Thao tác (Operation) là một hàm hay thủ tục có thể áp dụng cho hoặc bởi các đối tượng trong 1 lớp
- Phương thức (Method) là một cách thức cài đặt của một thao tác trong một lớp

Ví dụ:



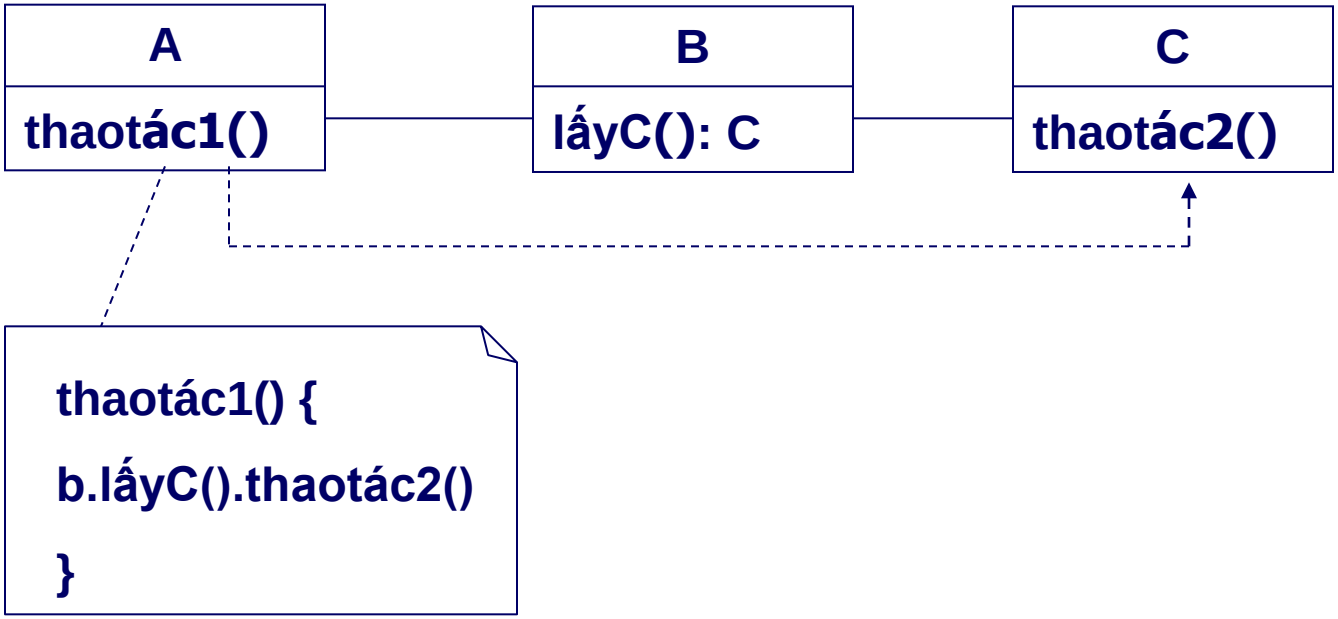
Mối quan hệ giữa các lớp

- Giữa các lớp có thể có ba mối liên quan:
  - Mối liên quan phụ thuộc.
  - Mối liên quan khái quát hóa.
  - Mối liên quan liên kết.

Mối liên hệ phụ thuộc

- Mối liên quan phụ thuộc (dependency relationship) thường dùng để diễn đạt một lớp (bên phụ thuộc) chịu ảnh hưởng của mọi thay đổi trong một lớp khác (bên độc lập), mà ngược lại thì không nhất thiết. Thường thì bên phụ thuộc cần dùng bên độc lập để đặc tả hay cài đặt cho mình. UML biểu diễn một mũi tên đứt nét (từ bên phụ thuộc sang bên độc lập).

Mối liên quan phụ thuộc (tt)



Mối liên quan khái quát hoá

- **Khái quát hoá** (generalization) là sự rút ra các đặc điểm chung của nhiều lớp để tạo thành một lớp giản lược hơn gọi là *lớp trên* (hay cha).
- Quá trình ngược lại gọi là **chuyên biệt hoá** (specialization): từ một lớp đã cho ta tăng cường thêm một số đặc điểm mới, tạo thành một lớp chuyên hơn, gọi là *lớp dưới* (hay con).

Mối liên quan khái quát hoá

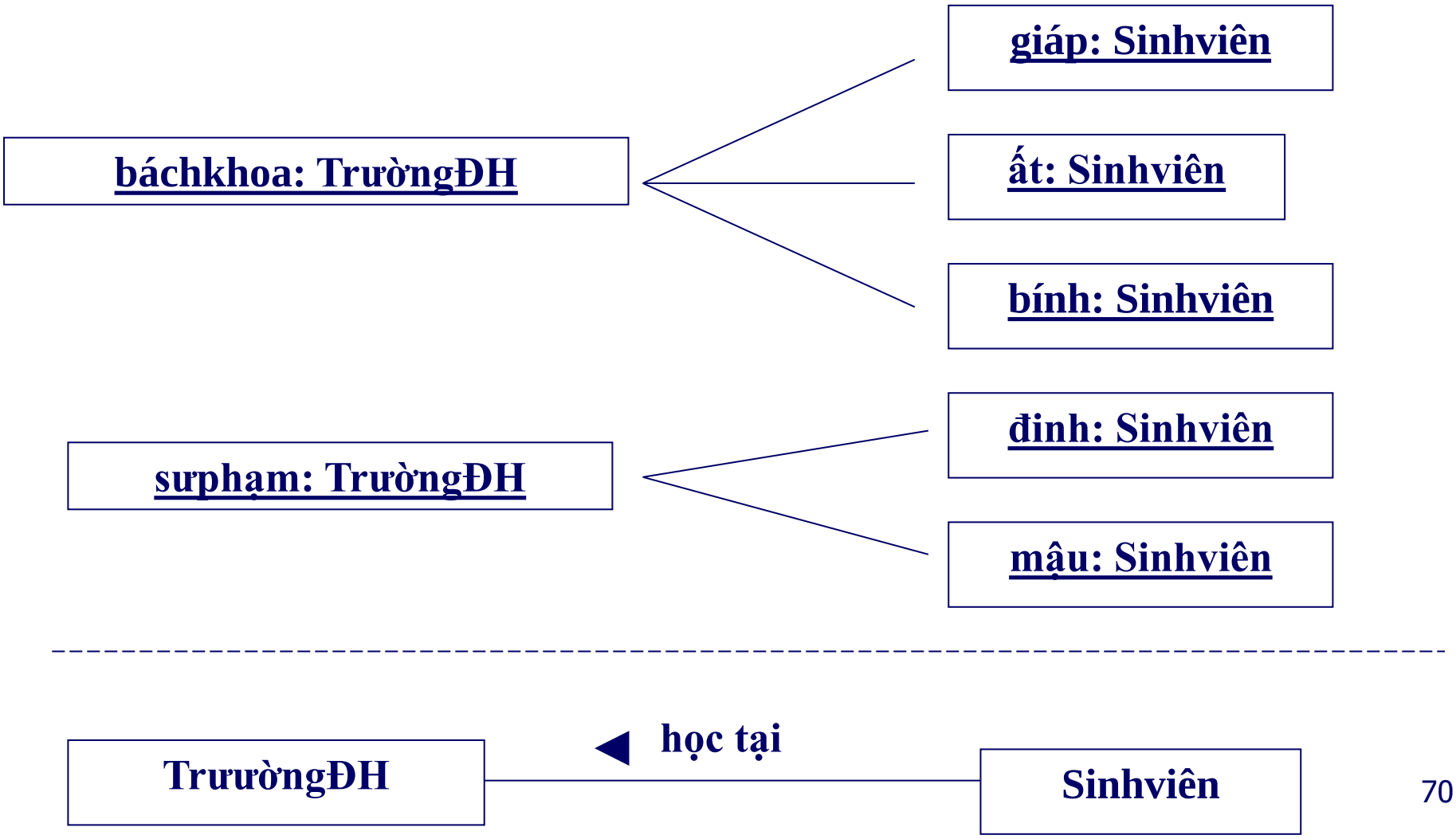
- Thuật ngữ thừa kế vốn được dùng nhiều trong các ngôn ngữ lập trình, nhằm diễn tả lớp con có mọi thuộc tính, thao tác và liên kết (sẽ nói ở dưới) được mô tả ở lớp cha. Hơn nữa lớp dưới có thể thêm thuộc tính, thao tác và liên kết mới và lại còn có thể định nghĩa lại (đè lấp) một thao tác của lớp trên (gọi đó là sự *đa hình* hay *đa xạ* (polymorphism)).
- Biểu diễn khái quát hoá:

Liên kết

Kết nối và liên kết

- Giữa các cá thể của hai lớp có thể tồn tại những sự ghép cặp, phản ánh một mối liên hệ nào đó trên thực tế. Gọi đó là một kết nối (link), ví dụ: kết nối khách hàng - hóa đơn v.v...
- Tập hợp những kết nối cùng loại (cùng ý nghĩa) giữa cá thể của hai lớp tạo thành một mối liên quan giữa hai lớp đó, gọi là một liên kết (association). Theo nghĩa đó thì đây chính là một quan hệ (hiểu theo nghĩa toán học) giữa hai tập hợp (là hai lớp).

Biểu diễn kết nối và liên kết



Phần mềm hỗ trợ PTTK hướng đối tượng

- Power Designer
- Rational Rose

Hết chương 2