

Systems Analysis & Design

Phân tích và thiết kế hệ thống

CS183 Spring Semester 2008



Dr. Jonathan Y. Clark

Dịch: kuam aka 

Email: j.y.clark@surrey.ac.uk

Course Website: www.computing.surrey.ac.uk/personal/st/J.Y.Clark/teaching/sad/cs183.html

Course Textbook:

Systems Analysis and Design With UML 2.0

An Object-Oriented Approach, Second Edition




Chapter 2:

Introduction to Object-Oriented Systems
Analysis & Design with the Unified Modeling
Language

Chương 2:

Giới thiệu về phân tích và phân tích hệ thống hướng
đối tượng với ngôn ngữ mô hình hóa thống nhất (UML)



Adapted from slides © 2005
John Wiley & Sons, Inc.

Objectives

Mục tiêu



- ☑ Understand the basic characteristics of object-oriented systems.
- ☑ Hiểu được những thành phần cơ bản của một hệ thống hướng đối tượng
- ☑ Be familiar with the Unified Modeling Language (UML), V.2.0.
- ☑ Làm quen với Ngôn ngữ Mô hình hóa Thống nhất UML phiên bản 2.0

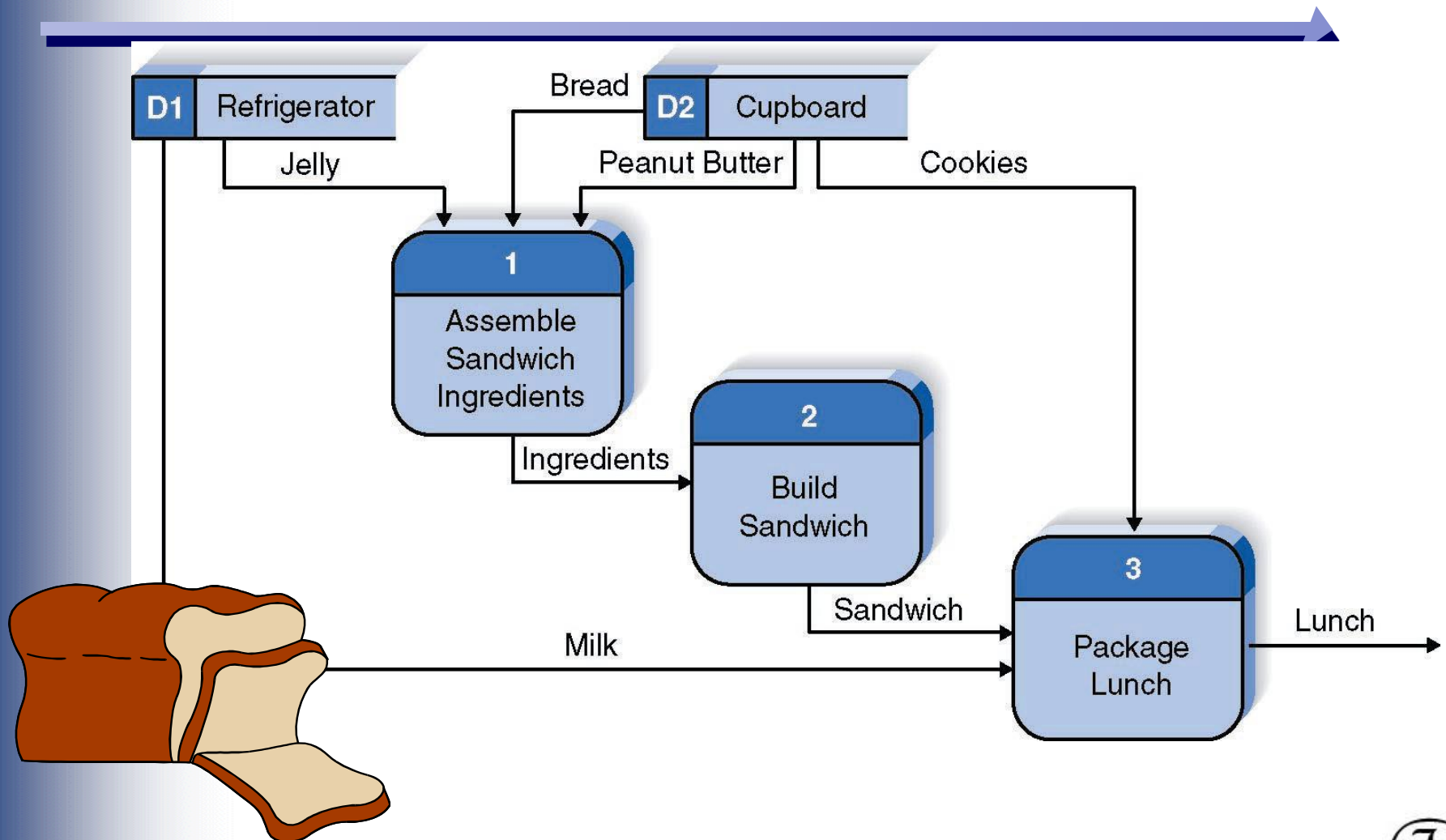
Non-Object-Oriented...

Không hướng đối tượng...

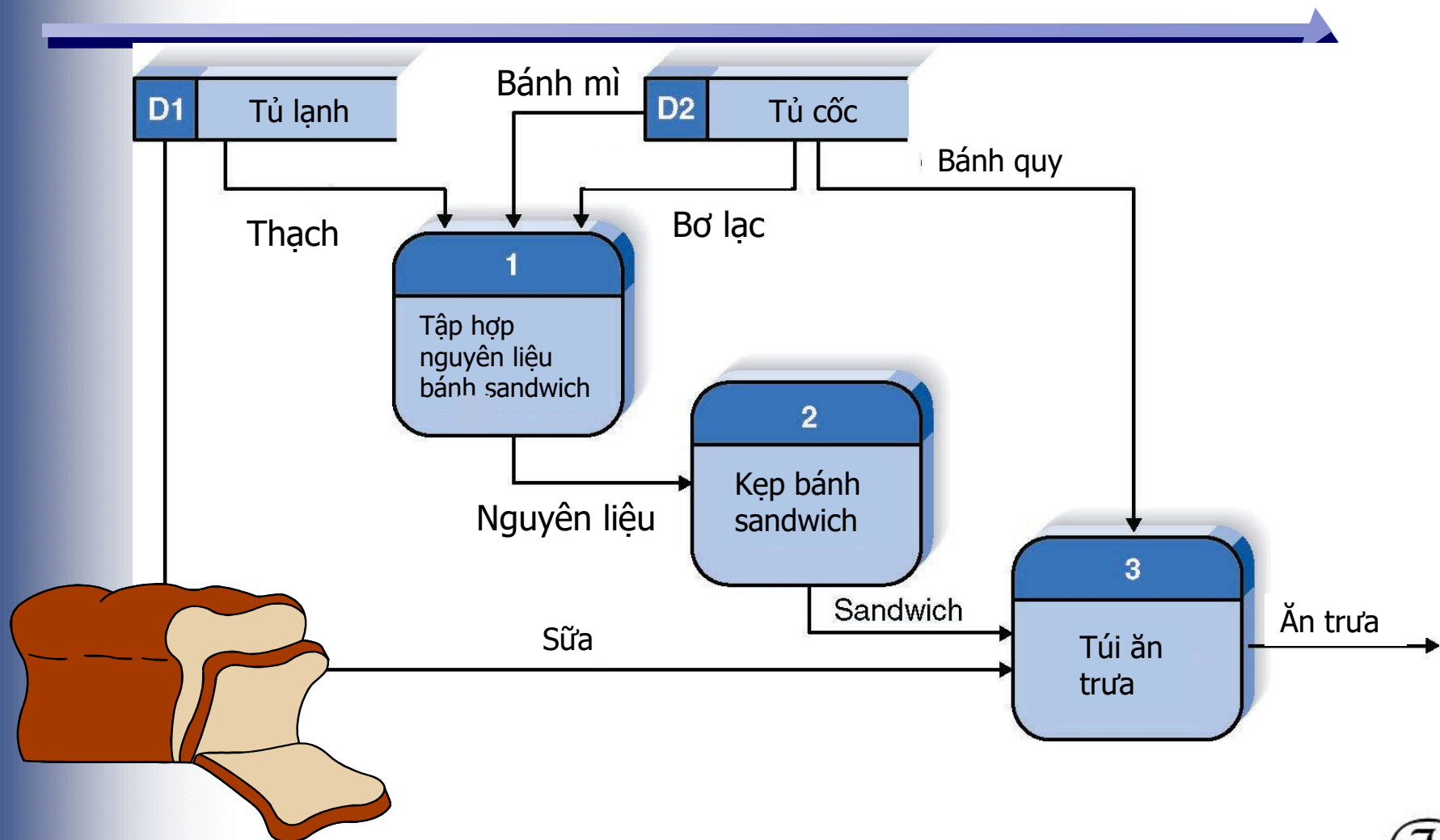


- ☑ Process models
- ☑ Mô hình tiến trình
 - Based on behaviour and actions
 - Dựa vào hành vi và hành động
- ☑ Data Models
- ☑ Mô hình dữ liệu
 - Based on static (fixed) representations of data
 - Dựa vào những thể hiện tĩnh (cố định) của dữ liệu

A "Simple" Process for Making Lunch



Các bước làm bữa trưa đơn giản

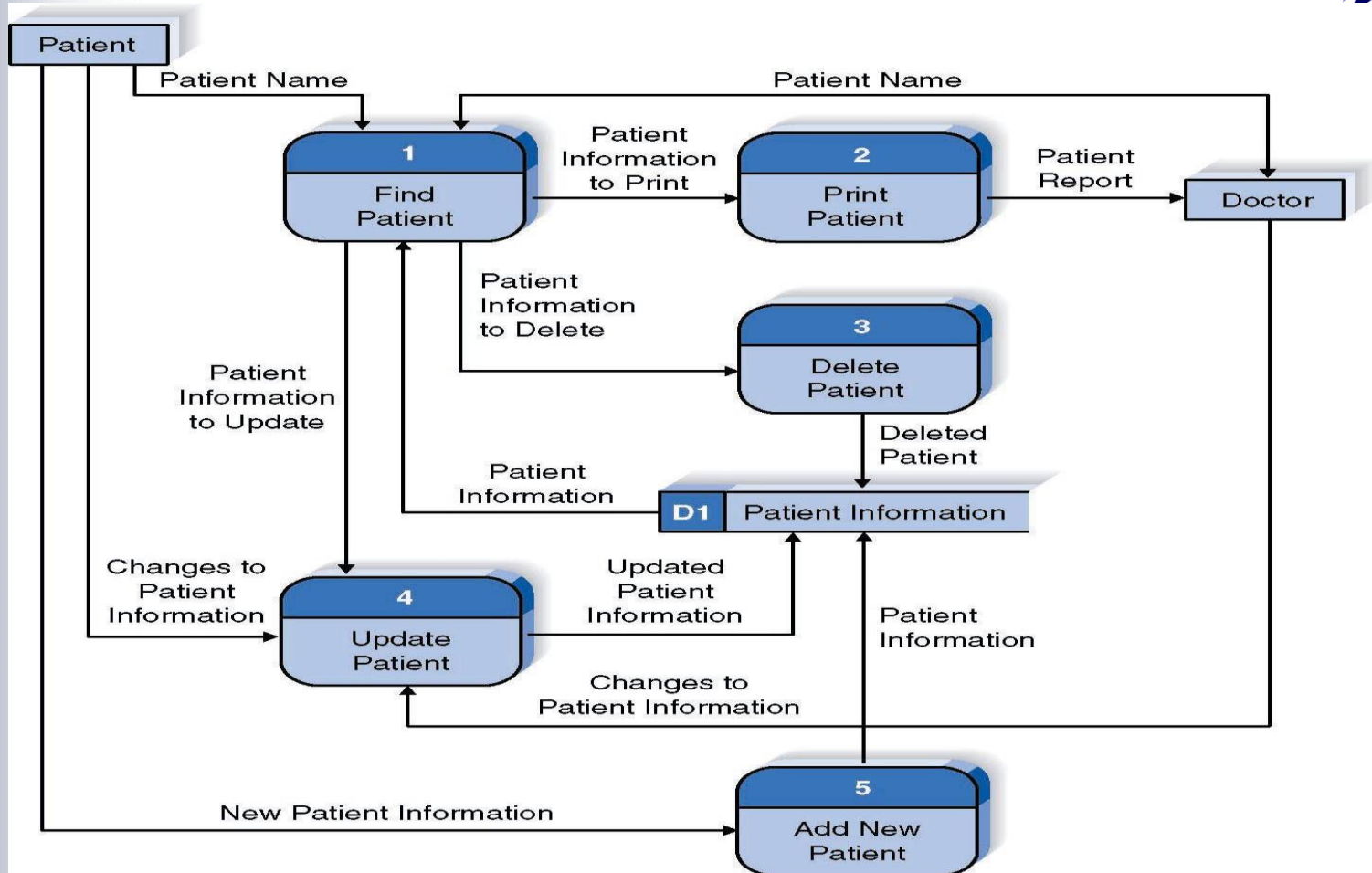


Process Modelling: Mô hình hóa tiến trình

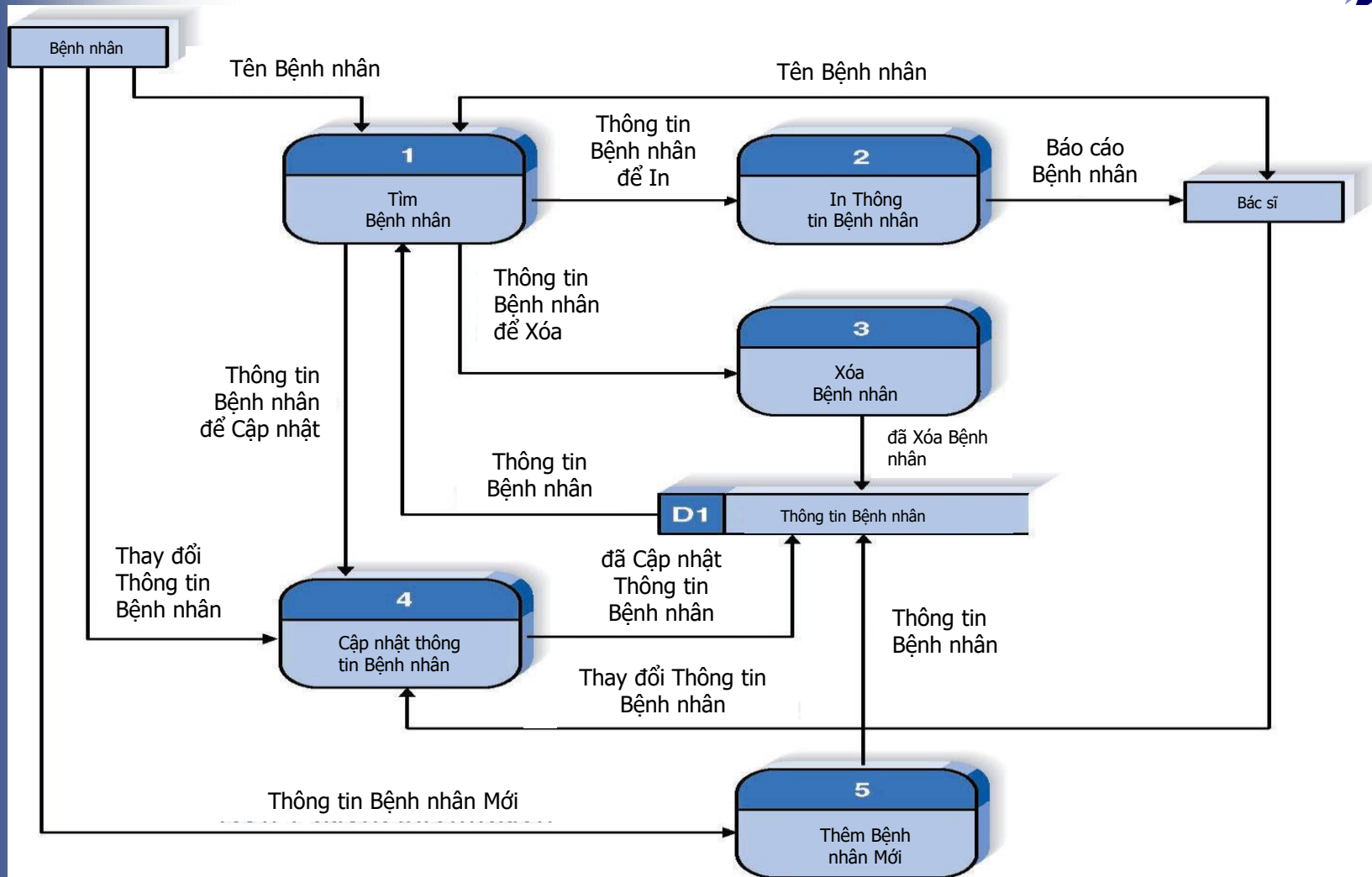


Data Flow Diagrams Sơ đồ Luồng Dữ liệu (DFD)

Reading a DFD



Xem một DFD



Data Modelling:

Mô hình hóa dữ liệu



Entity-Relationship Diagrams (ERDs)

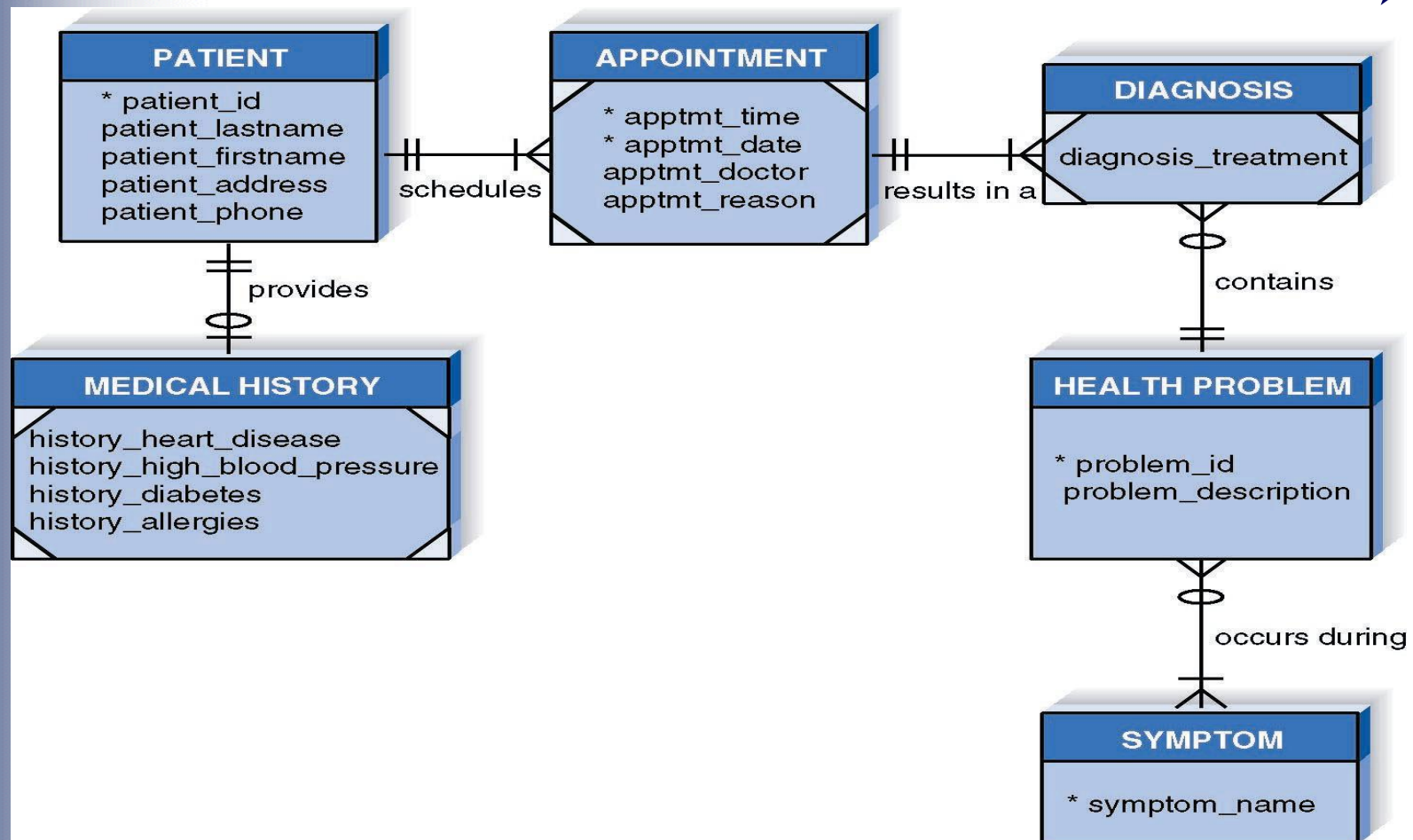
Sơ đồ Thực thể-Liên kết (ERDs)

What Is an ERD?

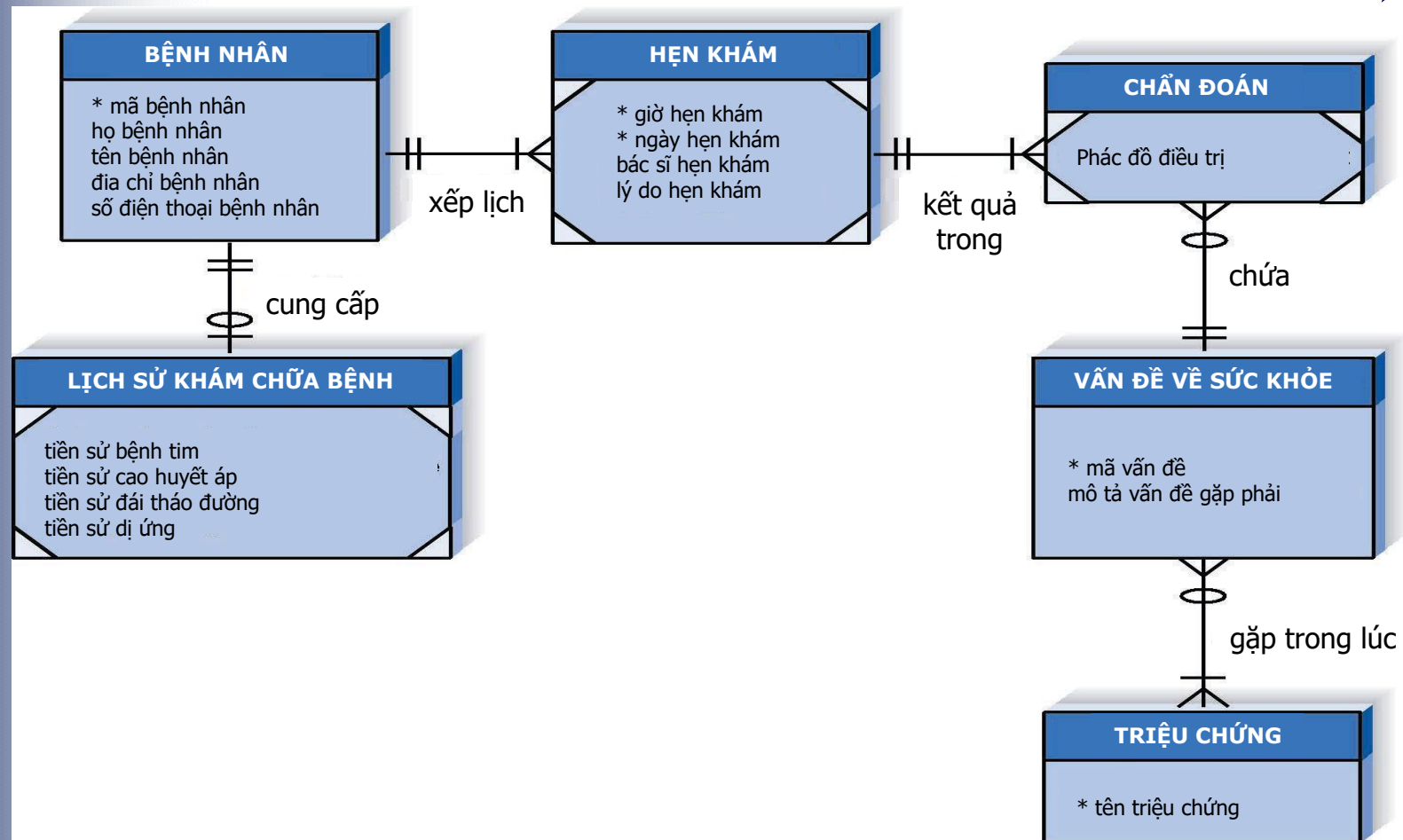
ERD là gì?

- ☑ A picture showing the information (is...) created, stored, and used by a business system.
- ☑ Là một bản vẽ cho ta thấy những thông tin được tạo ra, lưu trữ và sử dụng bởi một hệ thống nghiệp vụ.
- ☑ Entities generally represent people, places, and things of interest to the organization.
- ☑ Thực thể nói chung để biểu diễn con người, địa điểm, hay những gì ta quan tâm.
- ☑ Lines between entities show relationships between entities.
- ☑ Những đường thẳng nối giữa các thực thể là liên kết giữa chúng

An ERD Example



Ví dụ về ERD



Entities and Instances

Thực thể và bản thể

Entity Thực thể	Example Instances Ví dụ về bản thể
<div>Patient Bệnh nhân</div>	John Smith Susan Jones Peter Todd Dale Turner Pat Turner

Object-Oriented Approaches

Tiếp cận kiểu hướng đối tượng



- ❑ Combine processes and data
- ❑ Kết hợp quy trình và dữ liệu
- ❑ Are more 'natural'
- ❑ “Tự nhiên” hơn

Basic Characteristics of Object Oriented Systems

Các thành phần cơ bản của hệ thống hướng đối tượng



- ☑ Classes and Objects
- ☑ Lớp và Đối tượng
- ☑ Methods and Messages
- ☑ Phương thức và Thông điệp
- ☑ Encapsulation and Information Hiding
- ☑ Đóng gói và Che giấu Thông tin
- ☑ Inheritance
- ☑ Kế thừa
- ☑ Polymorphism
- ☑ Đa hình

Helpful Hint....'Compile'

Mẹo để nhớ..."Compile"

- ☑ C Classes
- ☑ O Objects
- ☑ M Methods and Messages
- ☑ P Polymorphism
- ☑ I Inheritance
- ☑ (Last, but not least)
- ☑ E Encapsulation

- ☑ C Lớp
- ☑ O Đối tượng
- ☑ M Phương thức và Thông điệp
- ☑ P Đa hình
- ☑ I Kế thừa
- ☑ (L Cuối cùng, nhưng không kém phần quan trọng)
- ☑ E Đóng gói

Classes and Objects

Lớp và đối tượng



- ❑ Class – Template to define specific instances or objects
- ❑ Lớp – Mẫu để ta định nghĩa các bản thể hay đối tượng cụ thể
- ❑ Object – Instantiation of a class
- ❑ Đối tượng – Thể hiện cụ thể của một lớp
- ❑ Attributes – Describes the object
- ❑ Thuộc tính – Để mô tả đối tượng
- ❑ Behaviours – specify what object can do
- ❑ Hành vi – xác định đối tượng có thể làm gì

Classes and Objects

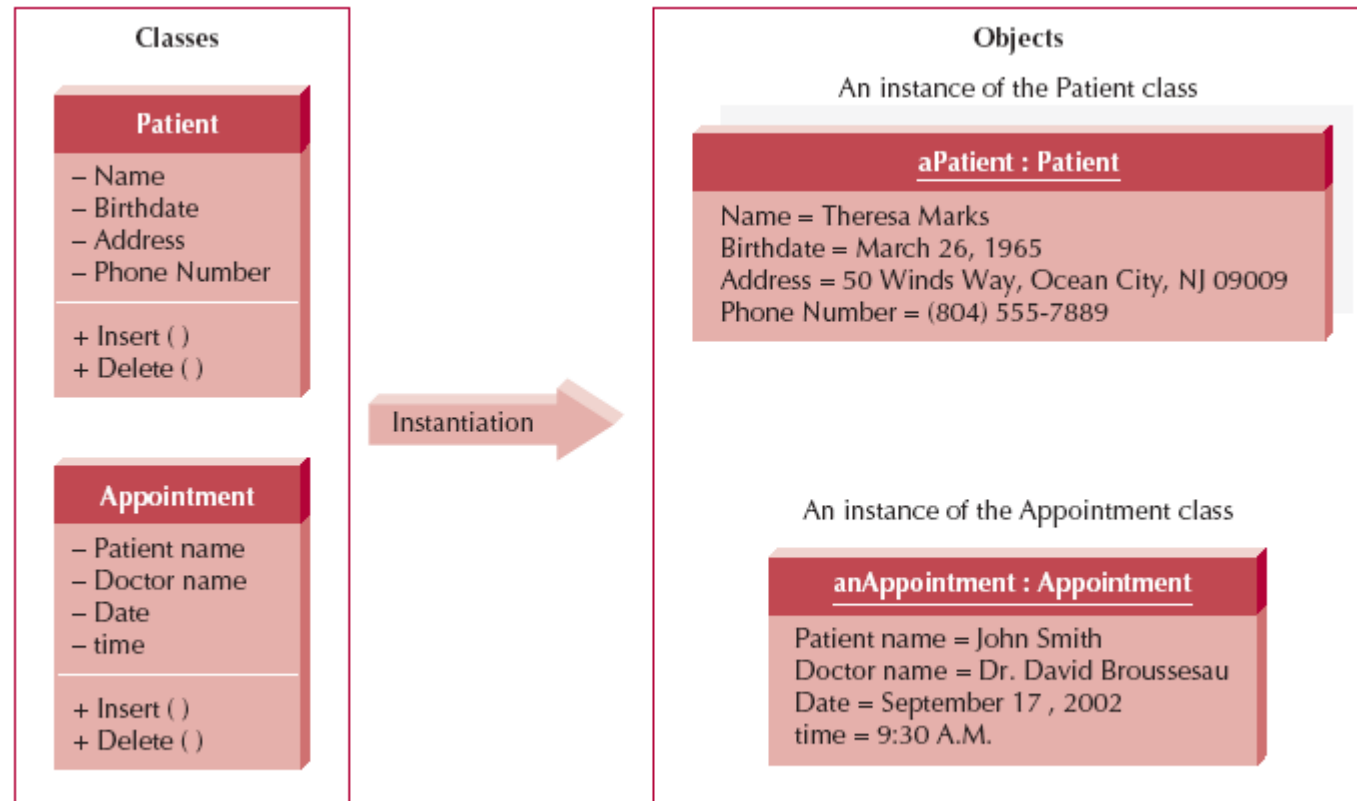


FIGURE 2-1 Classes and Objects

Lớp và Đối tượng

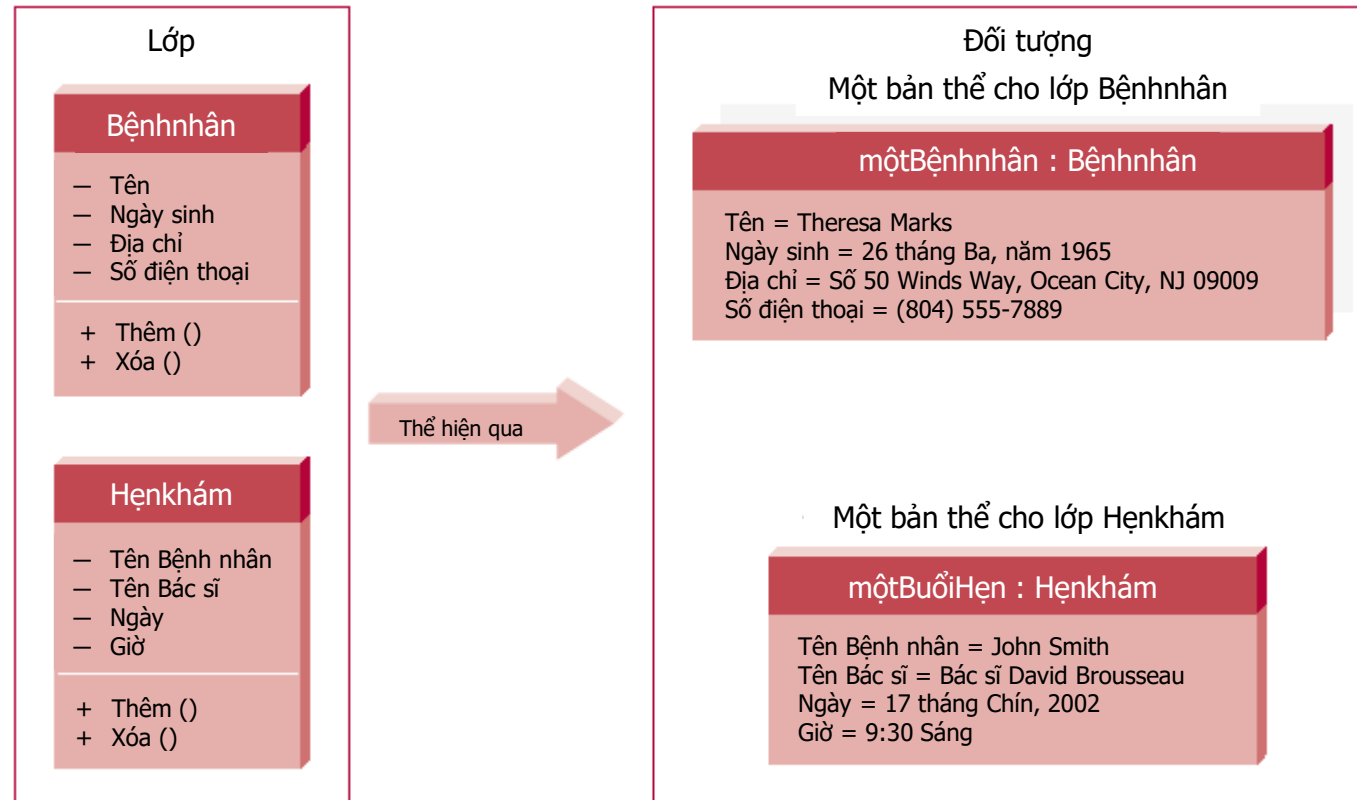


FIGURE 2-1 Classes and Objects

Methods and Messages

Phương thức và thông điệp



- ☑ Methods implement an object's behavior
- ☑ Phương thức thể hiện hành vi của đối tượng
 - Analogous to a function or procedure
 - Tương tự với hàm hay thủ tục
- ☑ Messages are sent to trigger methods
- ☑ Thông điệp được gửi đi để kích hoạt phương thức
 - Procedure call from one object to the next
 - Là thủ tục gọi từ đối tượng này sang đối tượng khác

Messages and Methods

Thông điệp và phương thức

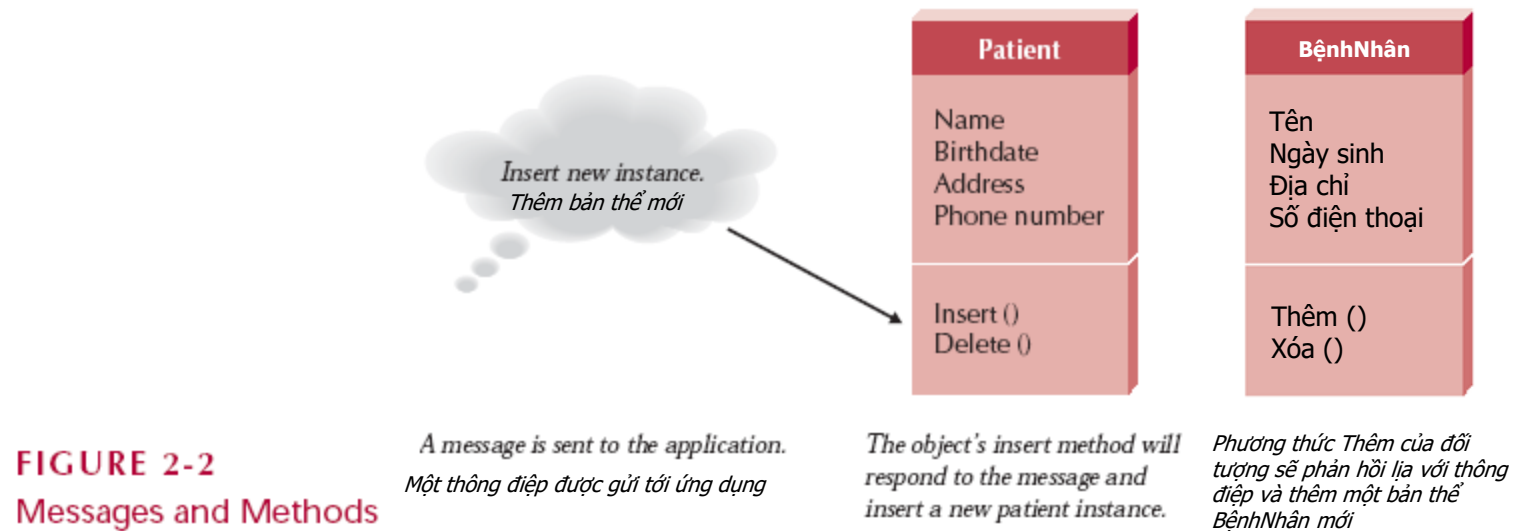


FIGURE 2-2
Messages and Methods

Encapsulation and Information Hiding

Đóng gói và che giấu thông tin



- ☑ Encapsulation

- ☑ Đóng gói

- combination of data and process into an entity
- Hợp nhất dữ liệu và quy trình thành một thực thể

- ☑ Information Hiding

- ☑ Che giấu thông tin

- Only the information required to use a software module is published to the user
- Chỉ những thông tin cần thiết để sử dụng phần mềm mới được công khai.

- ☑ Reusability is the Key Point

- ☑ Khả năng tái sử dụng là điểm mấu chốt

- an object is used by calling methods
- Một đối tượng được dùng bằng cách gọi phương thức

Inheritance

Kế thừa



- ☑ Superclasses or general classes are at the top of a hierarchy of classes
- ☑ Lớp cha hay lớp khái quát sẽ đứng trên đỉnh của cây kế thừa.
- ☑ Subclasses or specific classes are at the bottom
- ☑ Lớp con hay lớp cụ thể sẽ nằm ở đáy
- ☑ Subclasses inherit attributes and methods from classes higher in the hierarchy
- ☑ Lớp con sẽ kế thừa các thuộc tính và phương thức của lớp đứng cao hơn trong cây kế thừa.

Class Hierarchy

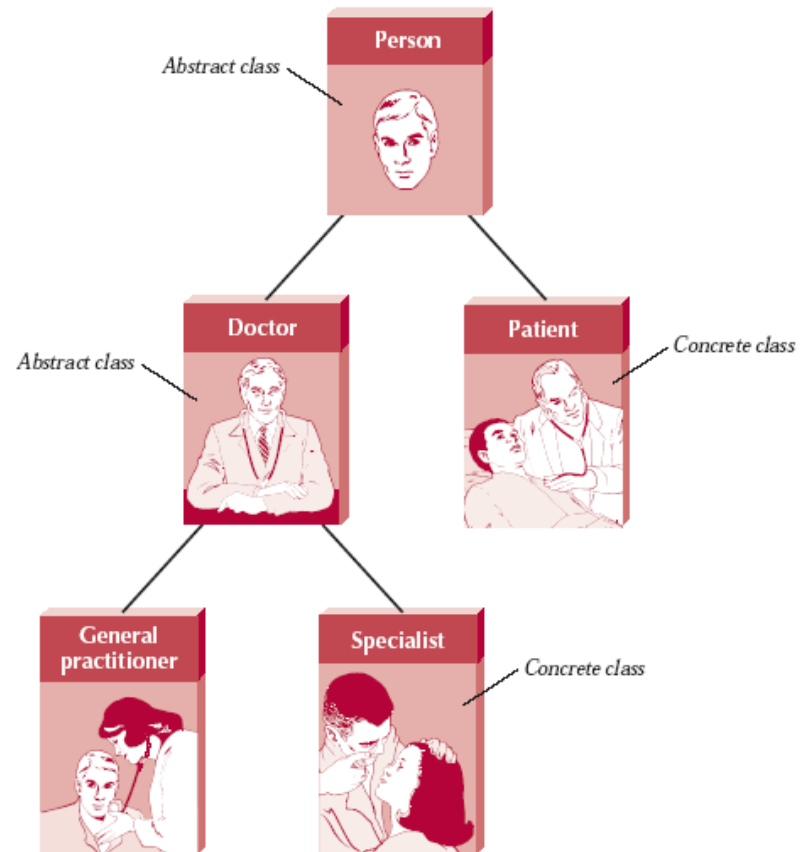


FIGURE 2-3
Class Hierarchy

Cây kế thừa

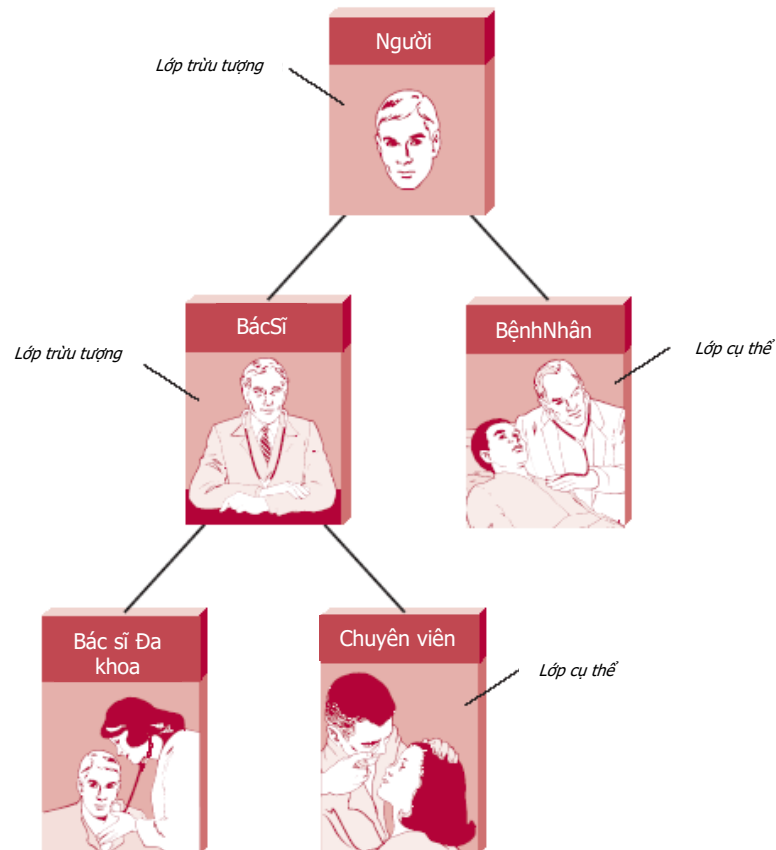


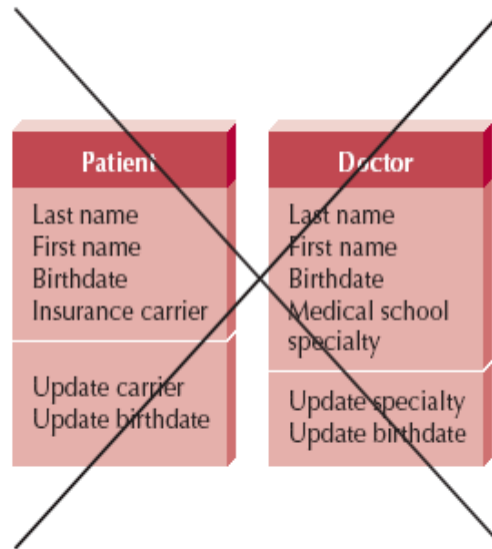
FIGURE 2-3
Class Hierarchy

Inheritance

Kế thừa

Cái hình này
khỏi dịch nhé :v

Without Inheritance
Không kế thừa



With Inheritance
Có kế thừa

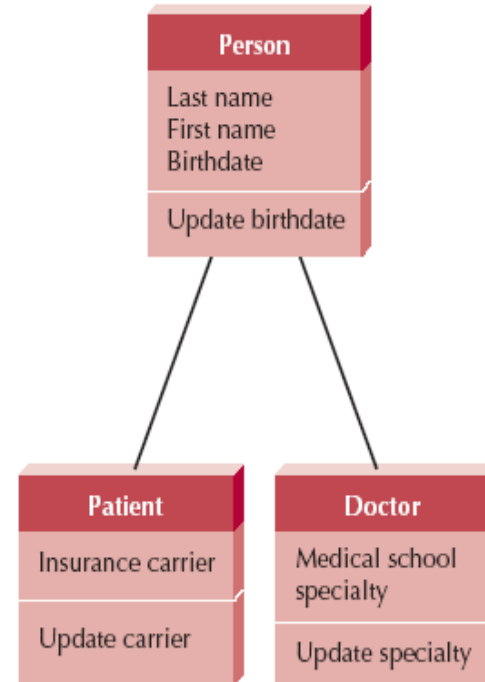


FIGURE 2-4
Inheritance

Polymorphism

Đa hình

- ☑ A message can be interpreted differently by different classes of objects
- ☑ Một thông điệp có thể được hiểu khác nhau với từng lớp khác nhau
- ☑ e.g. A 'Create_Record' message is essentially the same thing, but causes 'Create_Patient_Record' by a 'Patient_Database' object, or 'Create_Doctor_Record' by a 'Healthcare_Staff_Database' object
- ☑ Ví dụ. Thông điệp "Tạo_Báo_cáo" về cơ bản chỉ được hiểu theo một cách, nhưng sẽ là "Tạo_Báo_cáo_Bệnh_Nhân" nếu gửi tới đối tượng "Cơ_sở_dữ_liệu_Bệnh_nhân", hay "Tạo_Báo_cáo_Bác_sĩ" nếu gửi tới đối tượng "Cơ_sở_dữ_liệu_đội_ngũ_y_bác_sĩ"

Polymorphism & Encapsulation

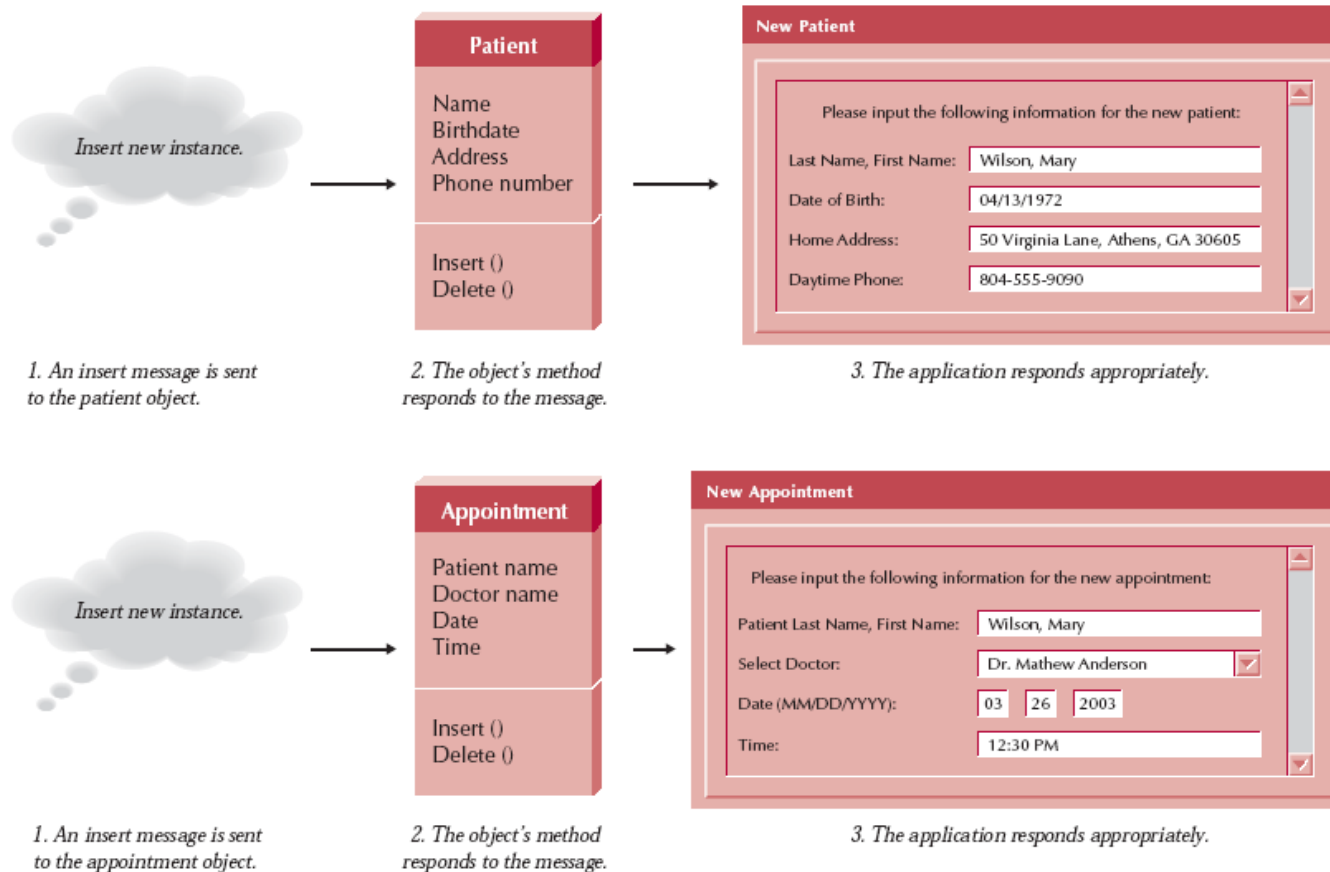


FIGURE 2-5 Polymorphism and Encapsulation

Đa hình và Đóng gói

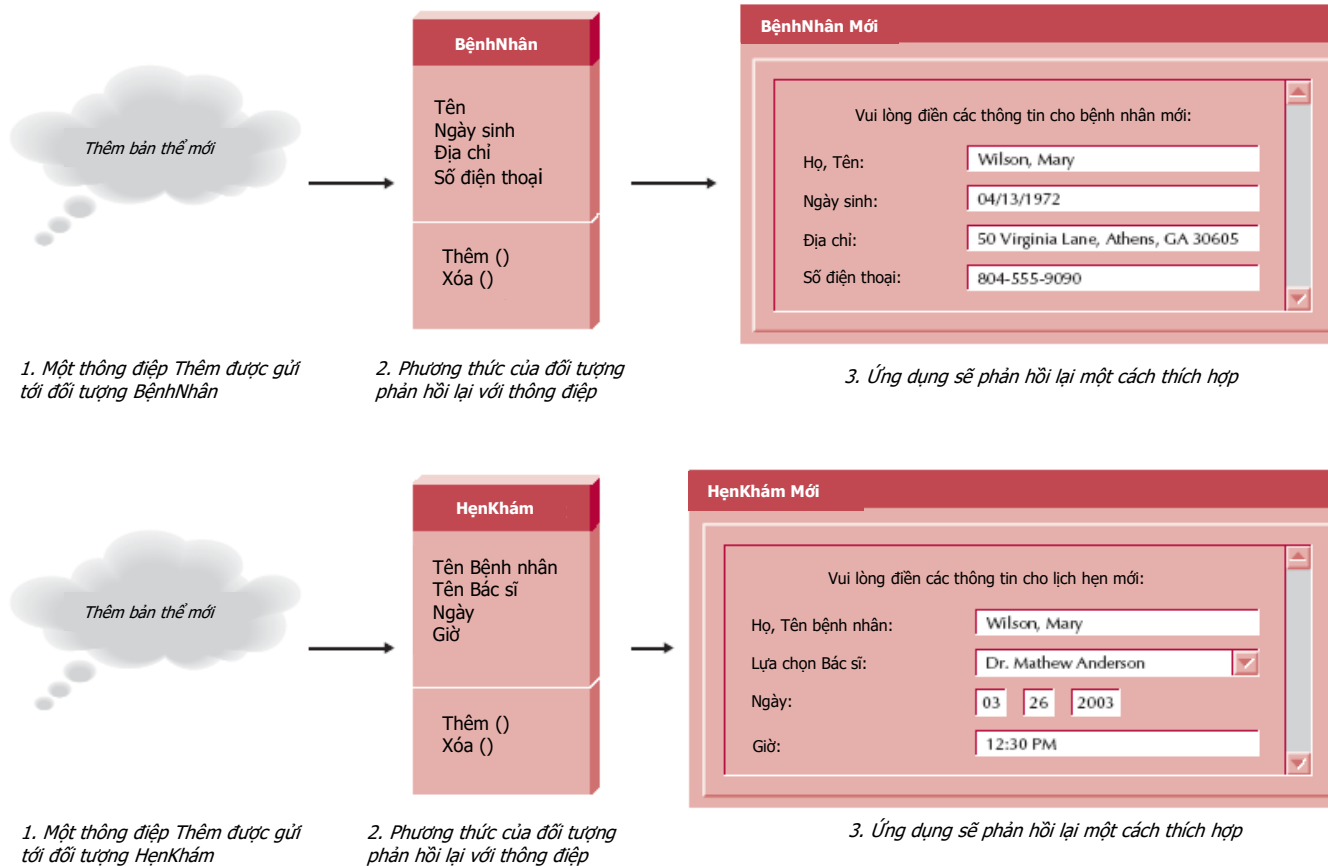


FIGURE 2-5 Polymorphism and Encapsulation

Benefits of the Object Approach

Concept	Supports	Leads to
Classes, objects, methods, and messages	<ul style="list-style-type: none"> ■ A more realistic way for people to think about their business ■ Highly cohesive units that contain both data and processes 	<ul style="list-style-type: none"> ■ Better communication between user and analyst-developer ■ Reusable objects ■ Benefits from having a highly cohesive system (see cohesion in Chapter 13)
Encapsulation and information hiding	<ul style="list-style-type: none"> ■ Loosely coupled units 	<ul style="list-style-type: none"> ■ Reusable objects ■ Fewer ripple effects from changes within an object or in the system itself ■ Benefits from having a loosely coupled system design (see coupling in Chapter 13)
Inheritance	<ul style="list-style-type: none"> ■ Allows us to use classes as standard templates from which other classes can be built 	<ul style="list-style-type: none"> ■ Less redundancy ■ Faster creation of new classes ■ Standards and consistency within and across development efforts ■ Ease in supporting exceptions
Polymorphism and Dynamic Binding	<ul style="list-style-type: none"> ■ Minimal messaging that is interpreted by objects themselves 	<ul style="list-style-type: none"> ■ Simpler programming of events ■ Ease in replacing or changing objects in a system ■ Fewer ripple effects from changes within an object or in the system itself
Use-case driven and use cases	<ul style="list-style-type: none"> ■ Allows users and analysts to focus on how a user will interact with the system to perform a single activity 	<ul style="list-style-type: none"> ■ Better understanding and gathering of user needs ■ Better communication between user and analyst
Architecture centric and functional, static, and dynamic views	<ul style="list-style-type: none"> ■ Viewing the evolving system from multiple points of view 	<ul style="list-style-type: none"> ■ Better understanding and modeling of user needs ■ More complete depiction of information system
Iterative and incremental development	<ul style="list-style-type: none"> ■ Continuous testing and refinement of the evolving system 	<ul style="list-style-type: none"> ■ Meeting real needs of users ■ Higher quality systems

FIGURE 2-8 Benefits of the Object Approach

Lợi ích của hướng đối tượng

Khái niệm	Lợi ích	Hướng tới
Lớp, đối tượng, phương thức và thông điệp	<ul style="list-style-type: none"> ❑ Một lối suy nghĩ thực tế hơn ❑ Các đơn vị gắn bó bền chặt, vừa chứa cả dữ liệu vừa chứa cả quy trình 	<ul style="list-style-type: none"> ❑ Kết nối người dùng và nhà phát triển tốt hơn ❑ Tái sử dụng đối tượng ❑ Lợi ích từ việc có một hệ thống cố kết cao (xem cố kết là gì ở Chương 13)
Đóng gói và che giấu thông tin	<ul style="list-style-type: none"> ❑ Các đơn vị liên kết yếu 	<ul style="list-style-type: none"> ❑ Tái sử dụng đối tượng ❑ Giảm thiểu các hiệu ứng phụ khi ta thay đổi giá trị của đối tượng hay các giá trị trong hệ thống ❑ Lợi ích từ hệ thống liên kết yếu (xem tính liên kết ở Chương 13)
Kế thừa	<ul style="list-style-type: none"> ❑ Cho chúng ta sử dụng những lớp làm mẫu để từ đó có thể xây dựng các lớp khác 	<ul style="list-style-type: none"> ❑ Giảm phần dư thừa ❑ Tạo lớp mới nhanh hơn ❑ Mang tính cơ bản và bền vững, phát triển đa nền tảng ❑ Dễ dàng hỗ trợ khi gặp ngoại lệ
Đa hình và liên kết động	<ul style="list-style-type: none"> ❑ Giảm thiểu lượng thông điệp chỉ riêng đối tượng đó hiểu 	<ul style="list-style-type: none"> ❑ Lập trình sự kiện dễ dàng hơn ❑ Dễ dàng thay thế hay thay đổi đối tượng trong hệ thống ❑ Giảm thiểu các hiệu ứng phụ khi ta thay đổi giá trị của đối tượng hay các giá trị trong hệ thống
Dẫn động ca sử dụng và ca sử dụng	<ul style="list-style-type: none"> ❑ Cho phép người dùng và các nhà phân tích tập trung vào cách người dùng tương tác với hệ thống để thực hiện một hoạt động nào đó 	<ul style="list-style-type: none"> ❑ Hiểu và thu thập nhu cầu người dùng tốt hơn ❑ Kết nối nhà phân tích hệ thống với người dùng
Trung tâm kiến trúc và hướng chức năng, hướng tính và hướng động	<ul style="list-style-type: none"> ❑ Cho ta nhìn một hệ thống tiên tiến dưới nhiều góc độ khác nhau 	<ul style="list-style-type: none"> ❑ Hiểu và mô hình hóa nhu cầu người dùng tốt hơn ❑ Mô tả về hệ thống thông tin đầy đủ hơn
Phát triển lặp và gia tăng	<ul style="list-style-type: none"> ❑ Kiểm thử liên tục và tinh lọc trong một hệ thống tiên tiến 	<ul style="list-style-type: none"> ❑ Biết được nhu cầu thực sự của người dùng ❑ Hệ thống có chất lượng tốt hơn

FIGURE 2-8 Benefits of the Object Approach

The Unified Modelling Language, Version 2.0

Ngôn ngữ Mô hình hóa Thống nhất, Phiên bản 2.0



- ☑ Functional Diagrams
- ☑ Biểu đồ Chức năng
- ☑ Structure Diagrams
- ☑ Biểu đồ Cấu trúc
- ☑ Behaviour Diagrams
- ☑ Biểu đồ Hành vi

- ☑ Developers
- ☑ Các nhà phát triển
 - ✦ Grady Booch
 - ✦ Ivar Jacobson
 - ✦ James Rumbaugh

Functional Diagrams

Biểu đồ chức năng



☑ Activity Diagrams

☑ Biểu đồ Hoạt động

- Illustrate business workflows
- Mô phỏng luồng công việc

☑ Use-Case Diagrams

☑ Biểu đồ Ca Sử dụng

- Capture business requirements
- Nắm bắt yêu cầu công việc
- Illustrates interaction between system and environment

▫ Mô phỏng tương tác giữa hệ thống với môi trường

Structure Diagrams

Biểu đồ Cấu trúc



- ☑ Class diagrams

- ☑ Biểu đồ Lớp

- relationship between classes
- Thể hiện liên kết giữa các lớp

- ☑ Object diagrams

- ☑ Biểu đồ Đối tượng

- Relationships between objects
- Thể hiện liên kết giữa các đối tượng

Behaviour Diagrams

Biểu đồ Hành vi



- ☒ Interaction Diagrams
- ☒ Biểu đồ tương tác...
 - Sequence diagrams
 - Biểu đồ trình tự
 - ✦ Show Time-based ordering and behaviour of objects and their activities
 - ✦ Cho ta thấy các hành vi và hoạt động của đối tượng được sắp xếp theo thời gian
- ☒ State Machines ...
- ☒ Trạng thái máy...
 - Behavioural State Machines (Statechart diagrams)
 - Trạng thái hành vi của máy (Biểu đồ trạng thái)
 - ✦ Examines behaviour of one class/object
 - ✦ Kiểm tra hành vi của một lớp/đối tượng

Object Oriented Systems Analysis and Design

Phân tích và Thiết kế Hệ thống Hướng Đối tượng



- Use-case driven
- Dẫn động ca sử dụng
- Iterative and Incremental
- Lắp và gia tăng
- Often associated with PHASED Development (a RAD methodology)
- Thường kết hợp với phương pháp phát triển theo pha (một phương pháp kiểu RAD)

Basic Method for Development of Object Oriented Systems

Phương pháp Cơ bản trong Phát triển Hệ thống Hướng Đối tượng

- ☑ Identifying business value
- ☑ Analyze feasibility
- ☑ Develop workplan
- ☑ Staff the project
- ☑ Control and direct project
- ☑ Requirements determination
- ☑ Functional modelling
- ☑ Structural modelling
- ☑ Behavioural modelling
- ☑ Moving on to design

- ☑ Xác định giá trị nhiệm vụ
- ☑ Phân tích tính khả thi
- ☑ Lập kế hoạch
- ☑ Xây dựng đội ngũ
- ☑ Quản lý và định hướng
- ☑ Xác định yêu cầu
- ☑ Mô hình hóa chức năng
- ☑ Mô hình hóa cấu trúc
- ☑ Mô hình hóa hành vi
- ☑ Tối phần thiết kế

Summary

Tổng kết



- ☑ Process oriented (Data flow diagrams) and Data oriented (Entity relationship diagrams)
- ☑ Hướng quy trình (Sơ đồ luồng dữ liệu) và Hướng dữ liệu (Sơ đồ Thực thể liên kết)
- ☑ Basic characteristics of Object Oriented Systems Analysis and Design
- ☑ Các thành phần cơ bản trong Phân tích và Thiết kế Hệ thống Hướng Đối tượng
- ☑ Introduction to Unified Modelling Language and the Unified Process
- ☑ Giới thiệu Ngôn ngữ Mô hình hóa Thống nhất và Quy trình Thống nhất