ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

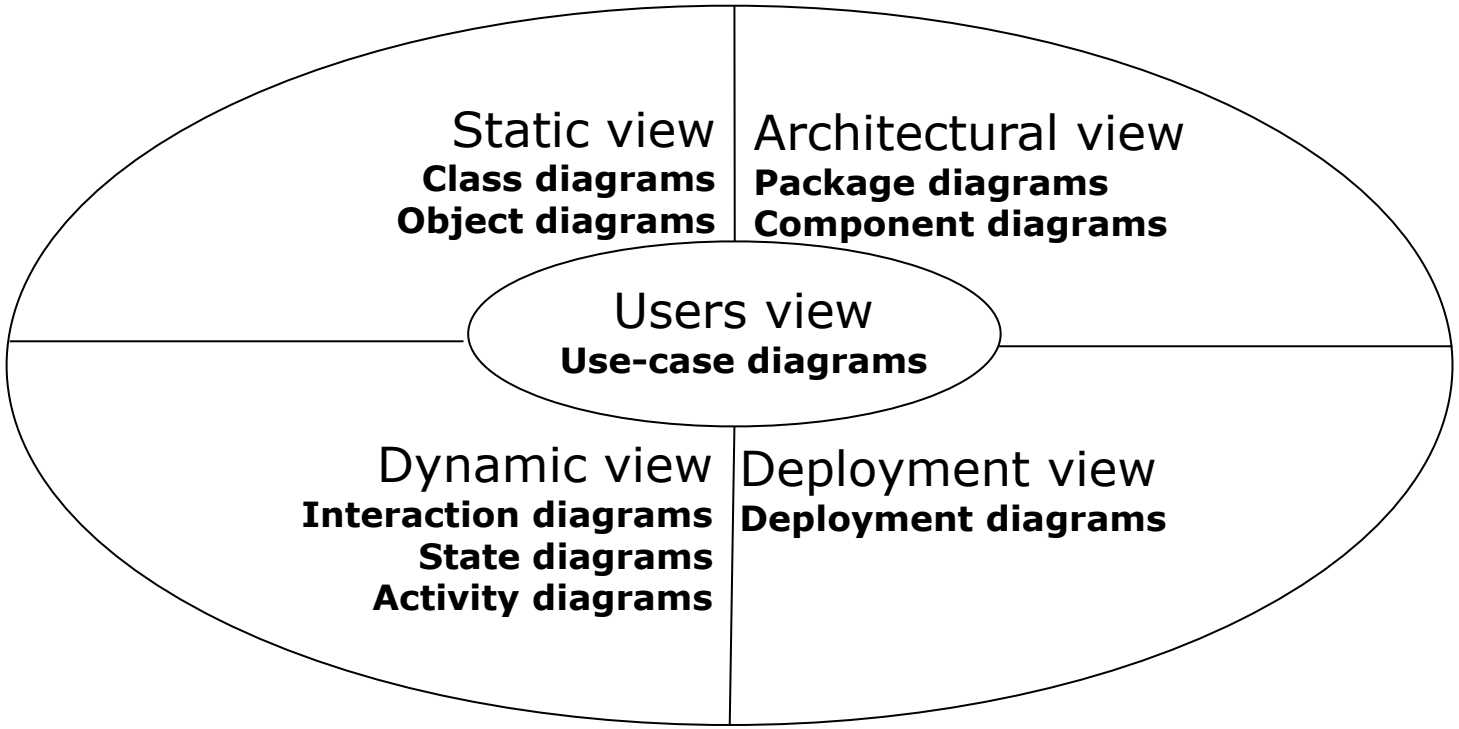
PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG PHẦN MỀM

Lê Viết Trương
Khoa Khoa học máy tính

Mô hình hóa kiến trúc

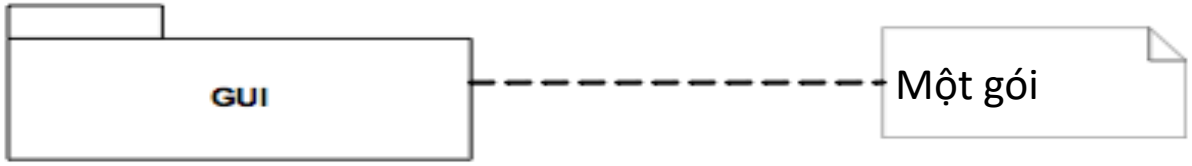
- Biểu đồ gói
- Biểu đồ thành phần
- Biểu đồ triển khai

Khung nhìn

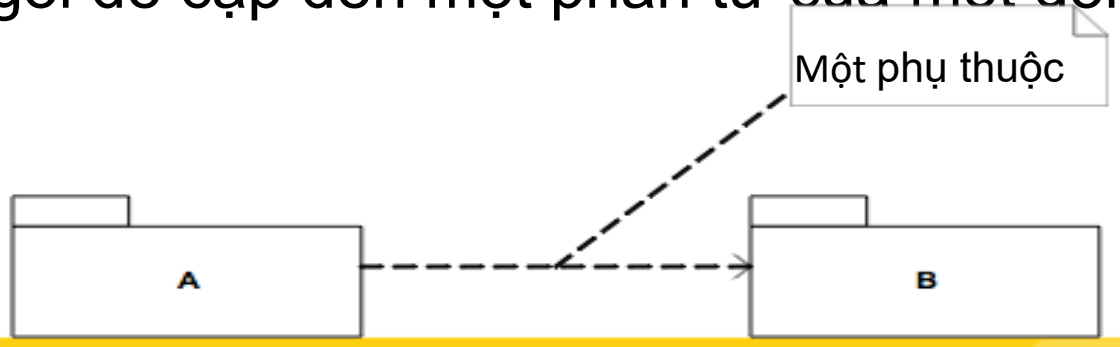

Static view
Class diagrams
Object diagrams
Architectural view
Package diagrams
Component diagrams
Users view
Use-case diagrams
Dynamic view
Interaction diagrams
State diagrams
Activity diagrams
Deployment view
Deployment diagrams

Biểu đồ gói

- Một gói cho phép nhóm các phần tử có liên quan
  - Một số lớp liên quan được nhóm lại với nhau thành một gói
  - Một số gói liên quan được nhóm thành một gói khác
- Ký hiệu

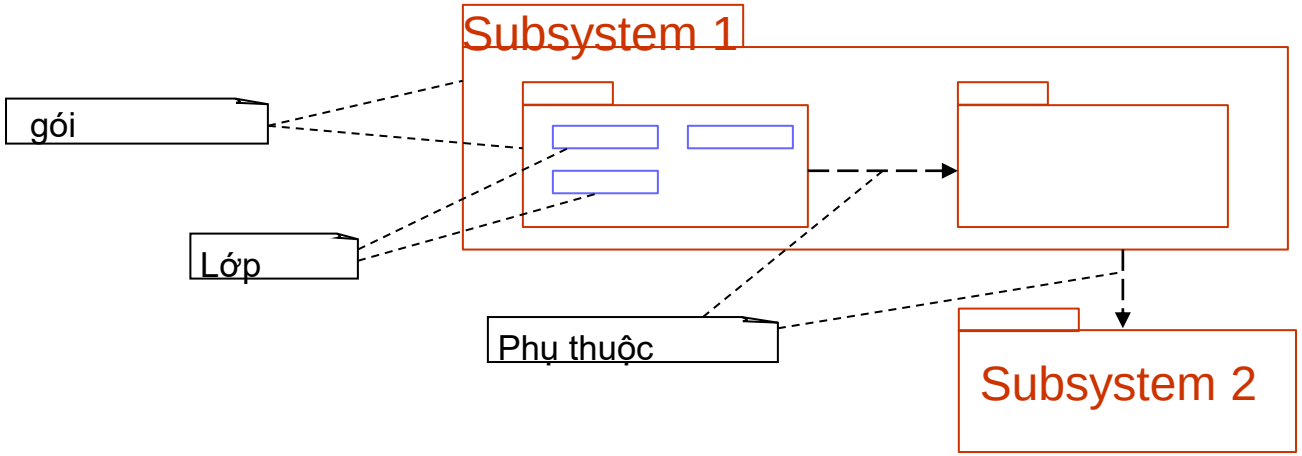


- Phụ thuộc
  - Một gói có thể phụ thuộc vào một gói khác
    - Ví dụ, một gói đề cập đến một phần tử của một gói khác
- Ký hiệu



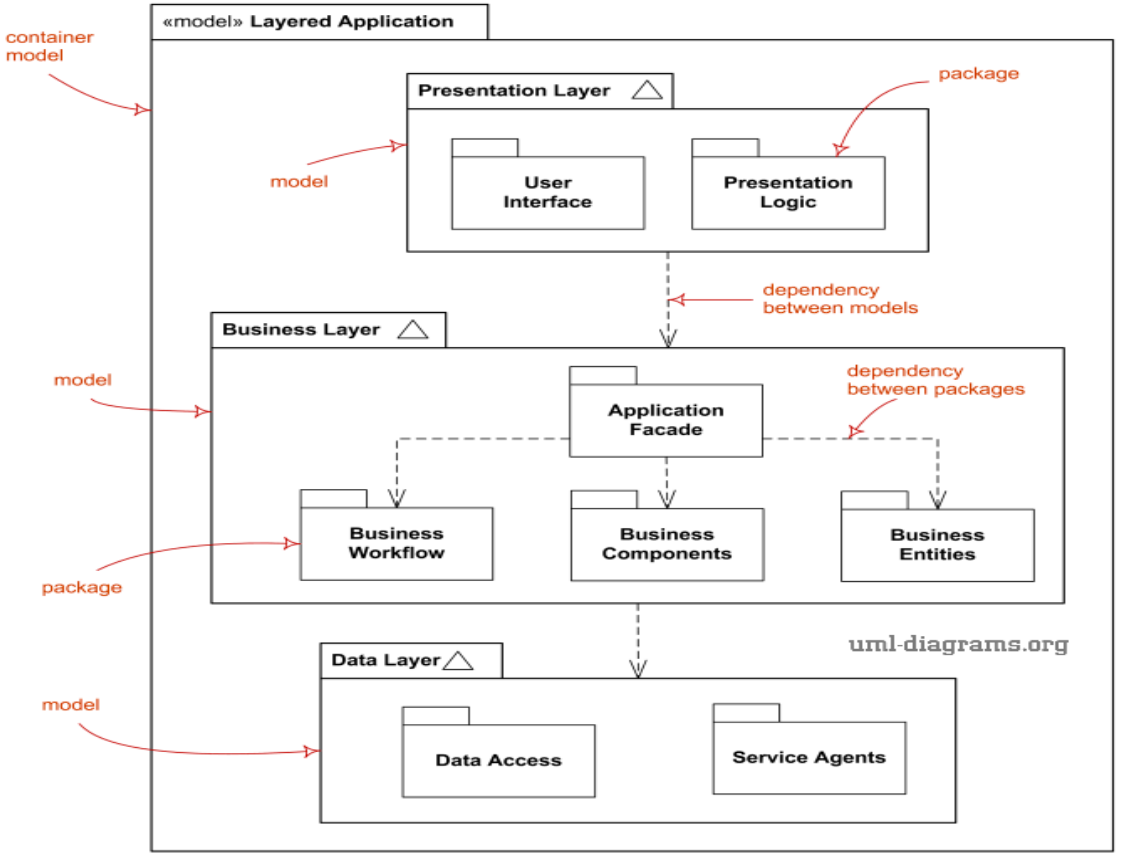
Biểu đồ gói

- Ví dụ



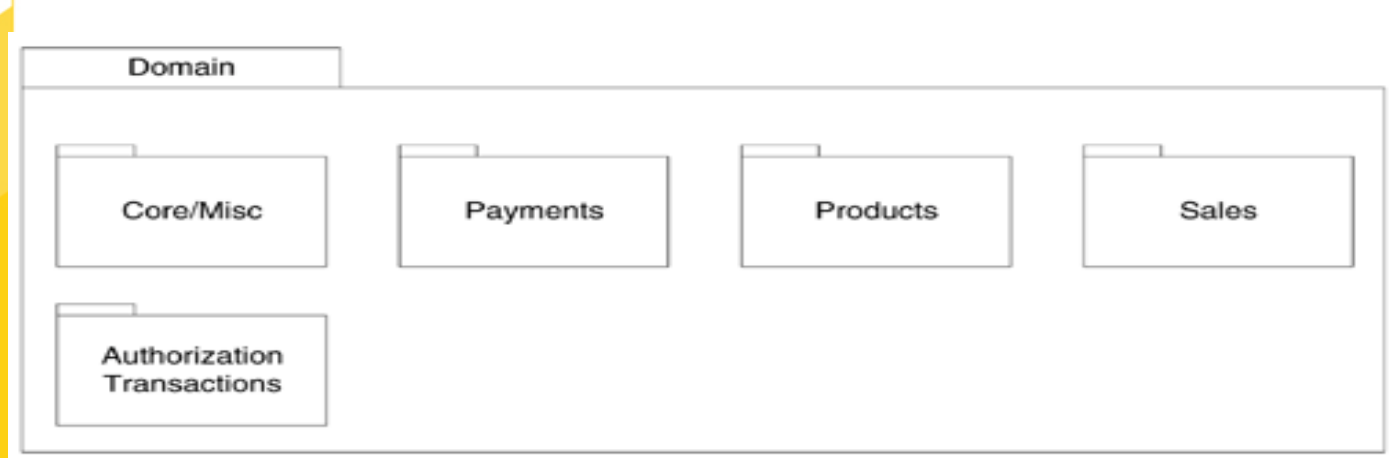
Biểu đồ gói

- Tại sao sử dụng gói?
  - Dễ dàng quản lý, hiểu và thao tác
  - Giảm sự phức tạp
  - Phát triển lặp đi lặp lại: các nhà phát triển, các nhóm khác nhau làm việc đồng thời trên các gói khác nhau
- Ví dụ

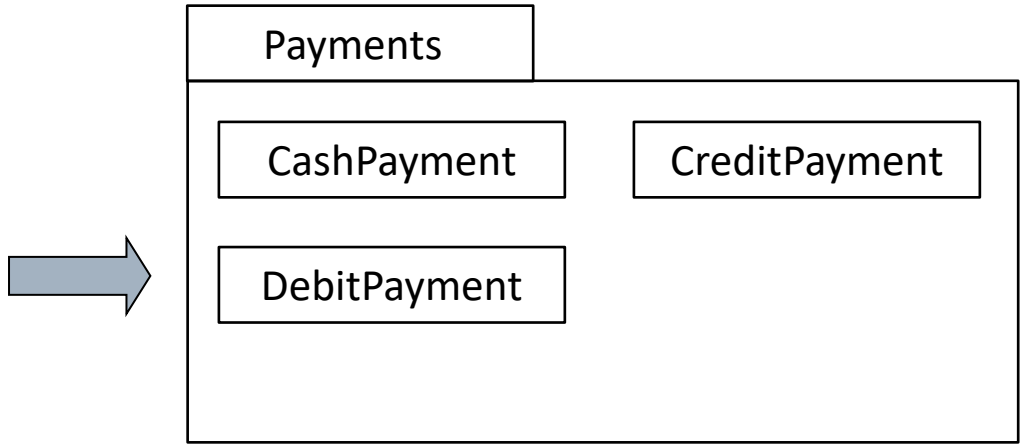


Biểu đồ gói

- Nguyên tắc tổ chức của gói
  - Sự gắn kết chức năng
    - Các lớp / giao diện được nhóm lại có liên quan chặt chẽ về mục đích, dịch vụ, cộng tác, chức năng
      - Ví dụ: tất cả các thành phần của gói "thanh toán" đều liên quan đến việc thanh toán các sản phẩm



Gói “Domain”



Gói “Payments”

Biểu đồ gói

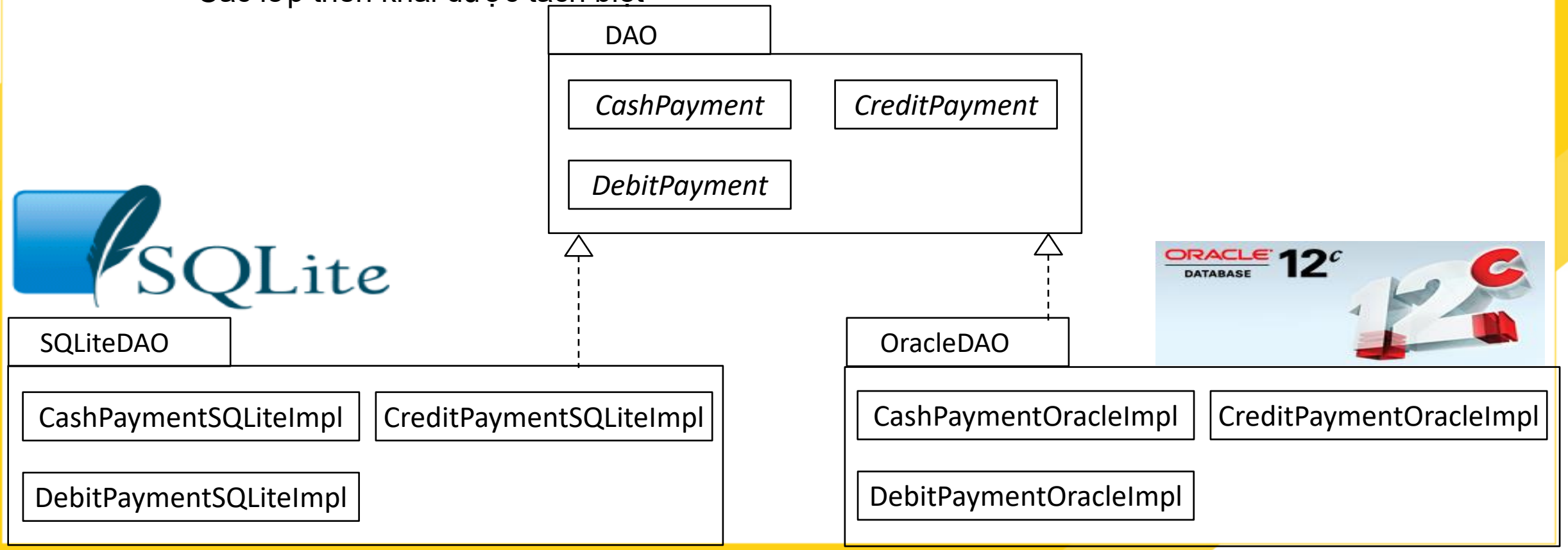
- Nguyên tắc tổ chức của gói
  - Sự gắn kết chức năng
    - **Sự gắn kết của một gói được định lượng bằng**

$$C = \frac{\textbf{Số lượng các mối quan hệ nội bộ}}{\textbf{Số phần tử}}$$

      - Một giá trị nhỏ của C có thể nói lên rằng
        - Gói chứa quá nhiều mục không liên quan, không được tổ chức tốt
        - Gói chứa các mục không liên quan với mục đích của nhà thiết kế
          - ví dụ như gói "công cụ"
        - Gói chứa một tập hợp con các phần tử có tính gắn kết cao, nhưng toàn bộ không gắn kết

Biểu đồ gói

- Nguyên tắc tổ chức của gói(tiếp theo)
  - Gói giao diện
    - Các giao diện liên quan được đặt trong một gói
    - Các lớp triển khai được tách biệt

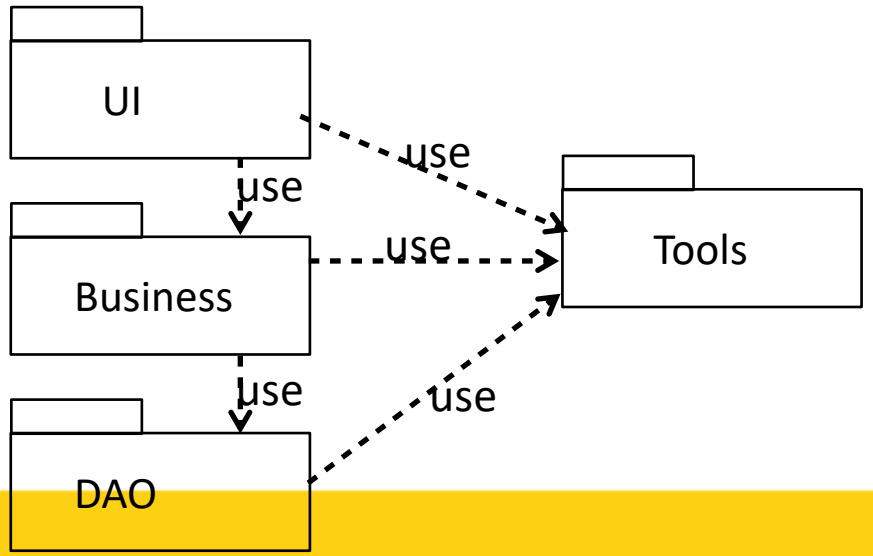


Biểu đồ gói

- Nguyên tắc tổ chức của gói(tiếp theo)
  - Gói các thành phần không ổn định
    - Các phần tử ổn định được nhóm trong một gói
    - Các phần tử không ổn định được nhóm lại với nhau trong một gói
      - Đây là những mục thường được sửa đổi và phân phối lại
      - Để giảm tác động đến các phần tử ổn định
      - Ví dụ: một gói bao gồm hai mươi lớp trong đó mười lớp thường được sửa đổi và phân phối lại. Tốt nhất là tách chúng thành hai gói: một bao gồm mười lớp ổn định, gói kia không ổn định.

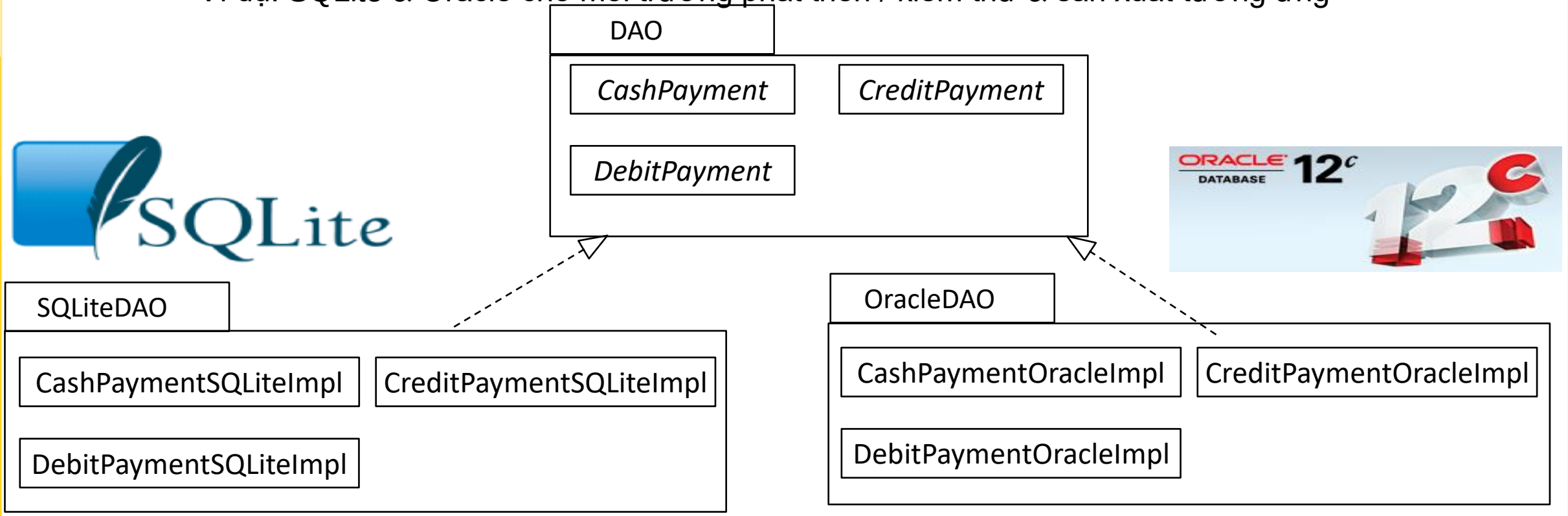
Biểu đồ gói

- Nguyên tắc tổ chức của gói(tiếp theo)
  - Càng phụ thuộc càng ổn định
    - Gói càng phụ thuộc thì càng phải ổn định
      - Vì những thay đổi trên các gói này có ảnh hưởng lớn đến các gói khác
      - Ví dụ, một gói “công cụ” cần phải ổn định.
    - Một số cách để cải thiện độ ổn định của gói
      - Nó chỉ chứa các giao diện hoặc các lớp trừu tượng
      - Nó không phụ thuộc vào các gói khác hoặc chỉ phụ thuộc vào các gói rất ổn định
      - Nó chứa mã ổn định (được cài đặt tốt)



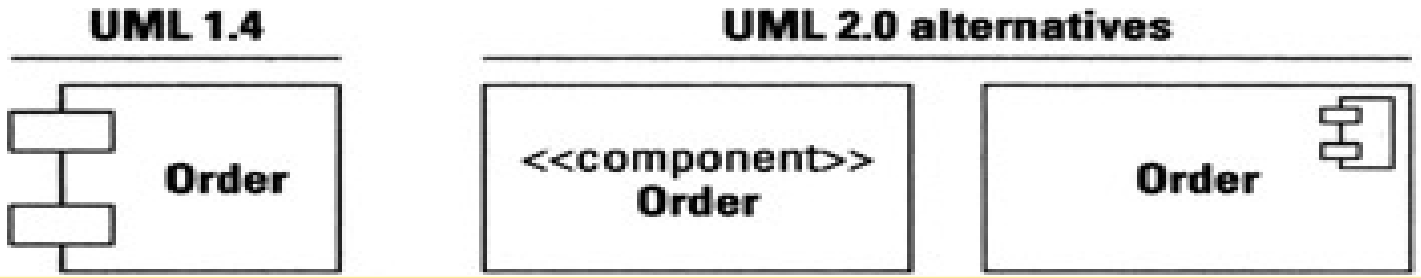
Biểu đồ gói

- Nguyên tắc tổ chức của gói(tiếp theo)
  - **Gói các phần tử độc lập**
    - Các phần tử được sử dụng độc lập hoặc trong các ngữ cảnh khác nhau được tách thành các gói khác nhau
    - Ví dụ: SQLite & Oracle cho môi trường phát triển / kiểm thử & sản xuất tương ứng



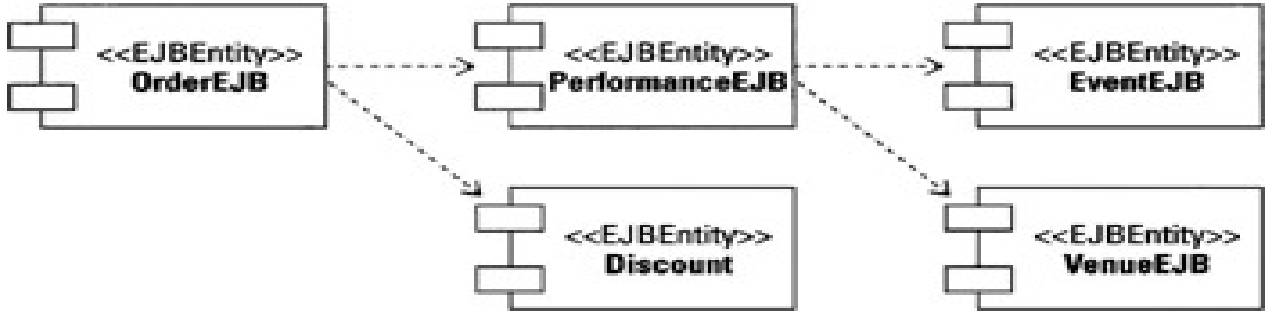
Biểu đồ thành phần

- Biểu đồ thành phần mô hình hóa **khung nhìn kiến trúc** của hệ thống, việc cài đặt vật lý của hệ thống
  - Các lớp mô tả tổ chức logic, trong khi các thành phần mô tả các cài đặt vật lý
- Biểu đồ thành phần xác định các mô-đun phần mềm vật lý và mối quan hệ của chúng với nhau
- Các thành phần có thể đại diện cho bất kỳ thứ gì từ một lớp đơn lẻ đến các ứng dụng, hệ thống con và hệ thống
- Phần mềm triển khai các thành phần
- Phần tạo tác có thể là bất kỳ loại mã nào có thể nằm trong bất kỳ loại mã nguồn bộ nhớ, tập tin nhị phân, tập lệnh, tập tin thực thi, cơ sở dữ liệu hoặc ứng dụng
- Sự phụ thuộc đại diện cho các loại mối quan hệ tồn tại giữa các thành phần trên biểu đồ Thành phần
- Ký hiệu

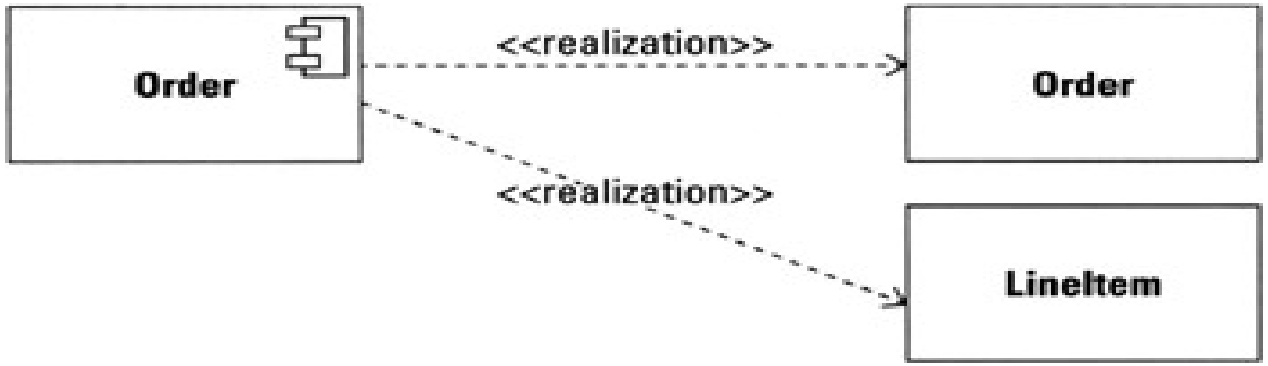


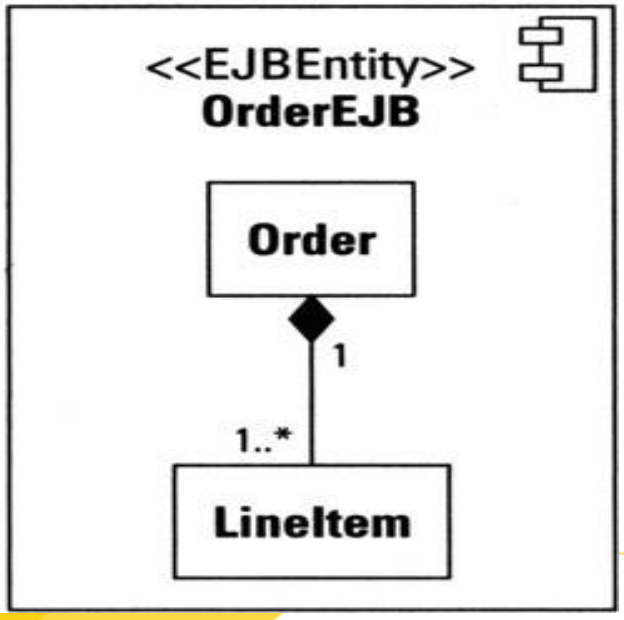
Biểu đồ thành phần

- Mô hình hóa các phụ thuộc



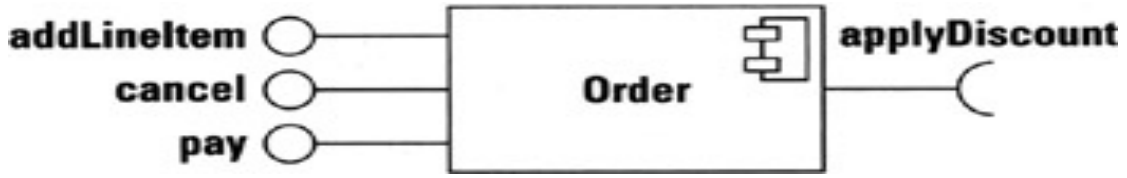
- Một thành phần là một trừu tượng, một đại diện của một yêu cầu đối với phần mềm vật lý.
- Hiện thực đề cập đến bất kỳ việc cài đặt một yêu cầu nào.
- Thành phần có thể chứa các thực hiện (cài đặt)





Biểu đồ thành phần

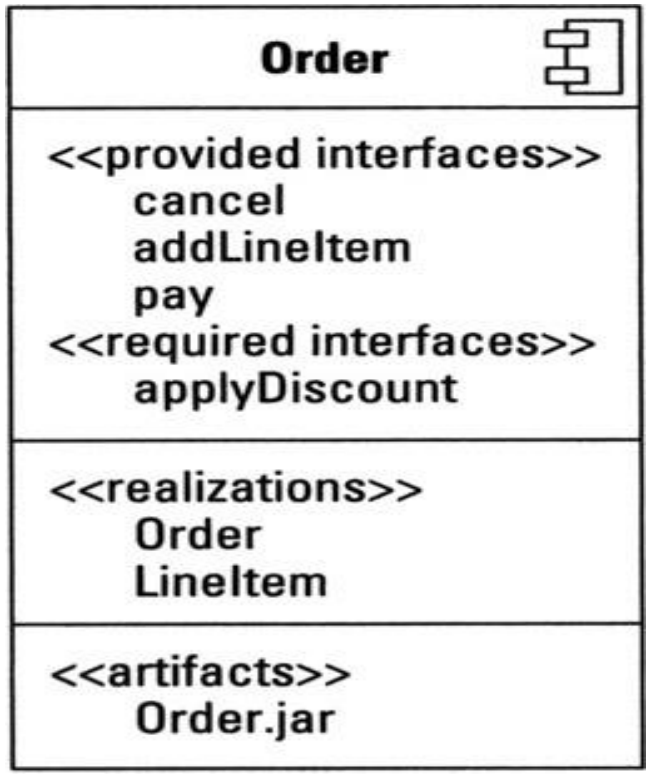
- Một tính năng cơ bản của các thành phần là khả năng xác định các giao diện
- Giao diện có hai loại: *bắt buộc* và *cung cấp*
  - *Giao diện được cung cấp*: xác định cách một thành phần khác phải yêu cầu quyền truy cập vào dịch vụ được cung cấp
  - *Giao diện bắt buộc*: xác định một giao diện chính xác những gì nó cần
- Ví dụ: Thành phần Order *cung cấp* các dịch vụ "add line items to the order", "cancel the order", and "pay for the order“; và *yêu cầu* discount/chiết khấu (từ thành phần khác)



- Kết nối các giao diện được yêu cầu và được cung cấp để tạo thành mối quan hệ đối tác giữa các thành phần

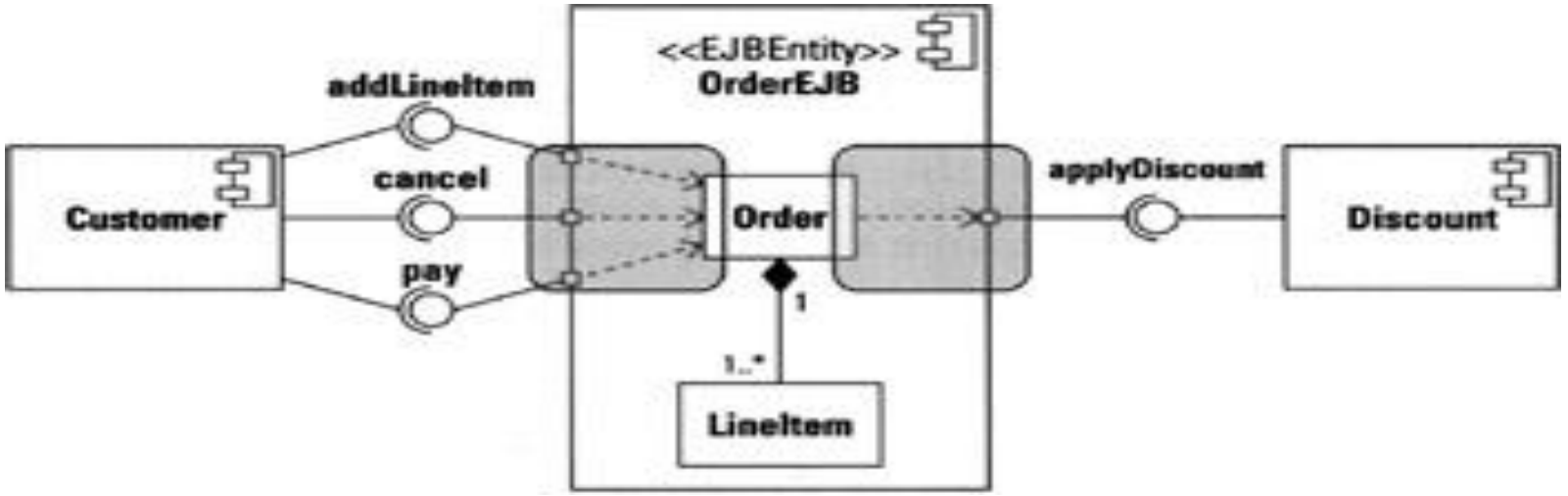
Biểu đồ thành phần

- UML 2.0 cung cấp một cách để đại diện cho tất cả thông tin được xác định cho đến nay cho một thành phần, bao gồm các giao diện, hiện thực hóa và tạo tác
  - Khung nhìn hộp trắng



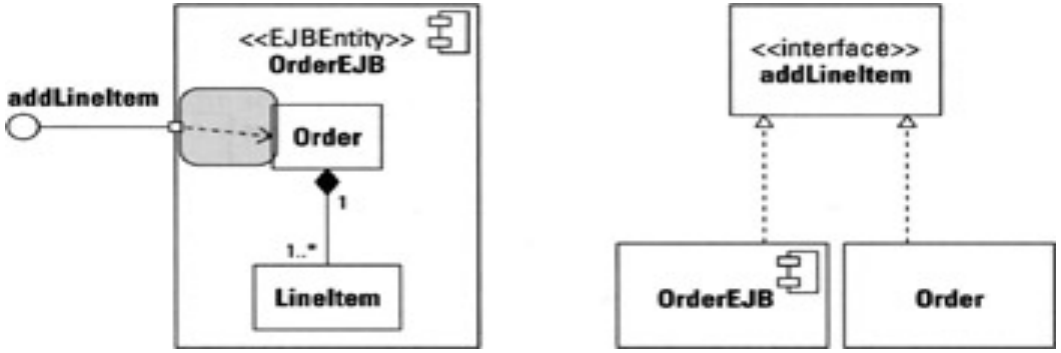
Biểu đồ thành phần

- **Các cổng** ánh xạ các giao diện của thành phần với các hiện thực hóa mà hỗ trợ chúng
- Một cổng xuất hiện dưới dạng một hình vuông nhỏ ở cạnh của thành phần

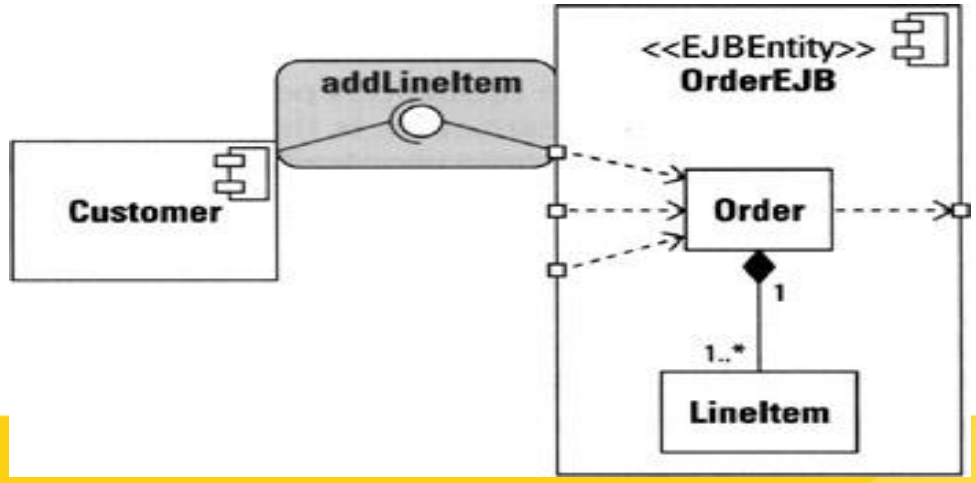


Biểu đồ thành phần

- Trình kết nối là một liên kết cho phép giao tiếp giữa hai hoặc nhiều thành phần
  - Hai loại kết nối: kết nối ủy quyền và kết nối lắp ráp
- Một trình kết nối ủy quyền ánh xạ một yêu cầu từ một cổng đến một cổng khác hoặc đến một hiện thực hóa cung cấp việc cài đặt cho yêu cầu

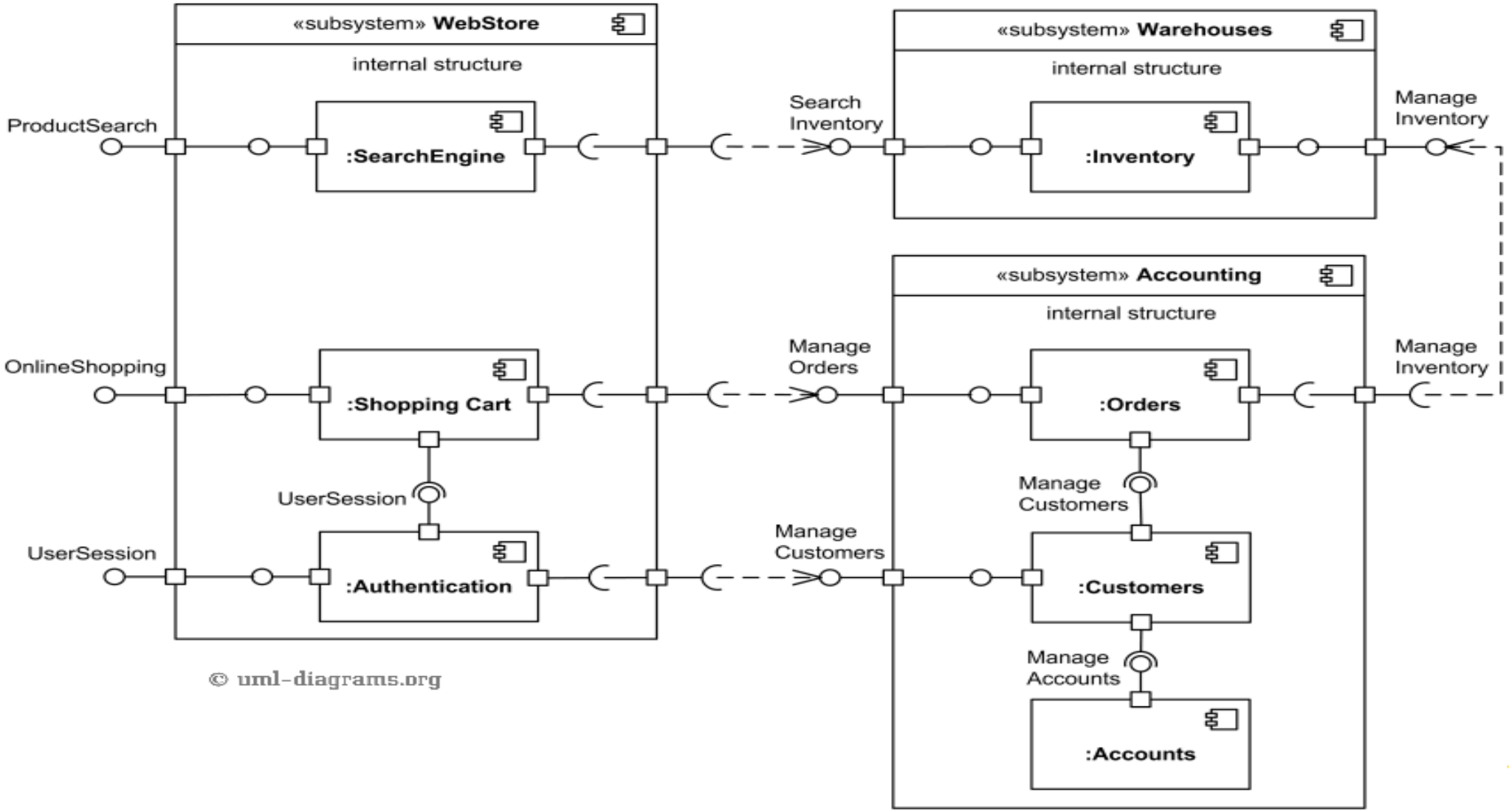


- Một trình kết nối lắp ráp được sử dụng để ánh xạ giao diện bắt buộc đến giao diện được cung cấp



Biểu đồ thành phần

- **Ví dụ**: Mua sắm trực tuyến với ba hệ thống con có liên quan - Webstore, Warehouses and Accounting

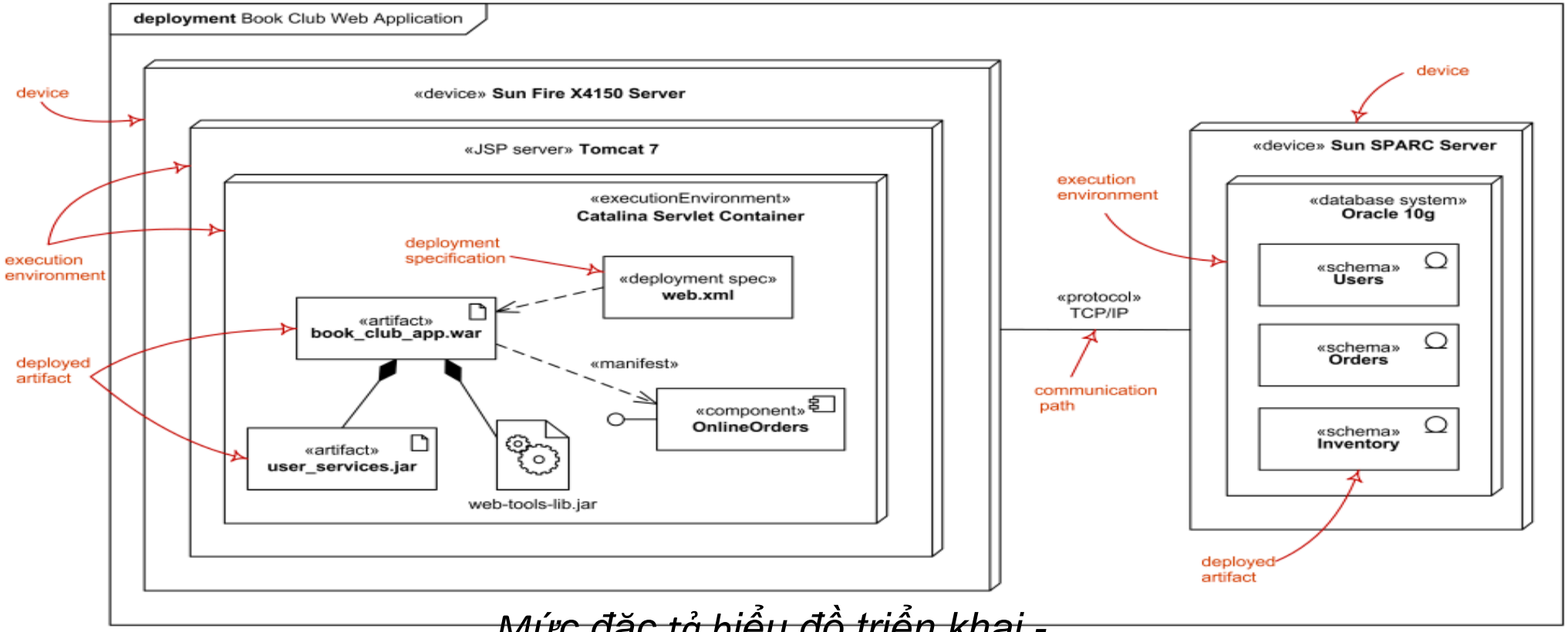


Biểu đồ triển khai

- **Biểu đồ triển khai** cho thấy kiến trúc của một hệ thống như việc triển khai (phân phối) các sản phẩm phần mềm đến các mục tiêu triển khai
  - Tạo tác đại diện cho các yếu tố cụ thể trong thế giới vật lý như tập tin thực thi, thư viện, kho lưu trữ, lược đồ cơ sở dữ liệu, tập tin cấu hình,…
  - Mục tiêu triển khai thường được đại diện bởi một **nút** là thiết bị phần cứng hoặc môi trường thực thi phần mềm. Các nút có thể được kết nối.
- Biểu đồ triển khai có thể mô tả kiến trúc ở **mức đặc tả** (còn gọi mức kiểu) hoặc ở **mức thể hiện** (tương tự như biểu đồ lớp và biểu đồ đối tượng).

Biểu đồ triển khai

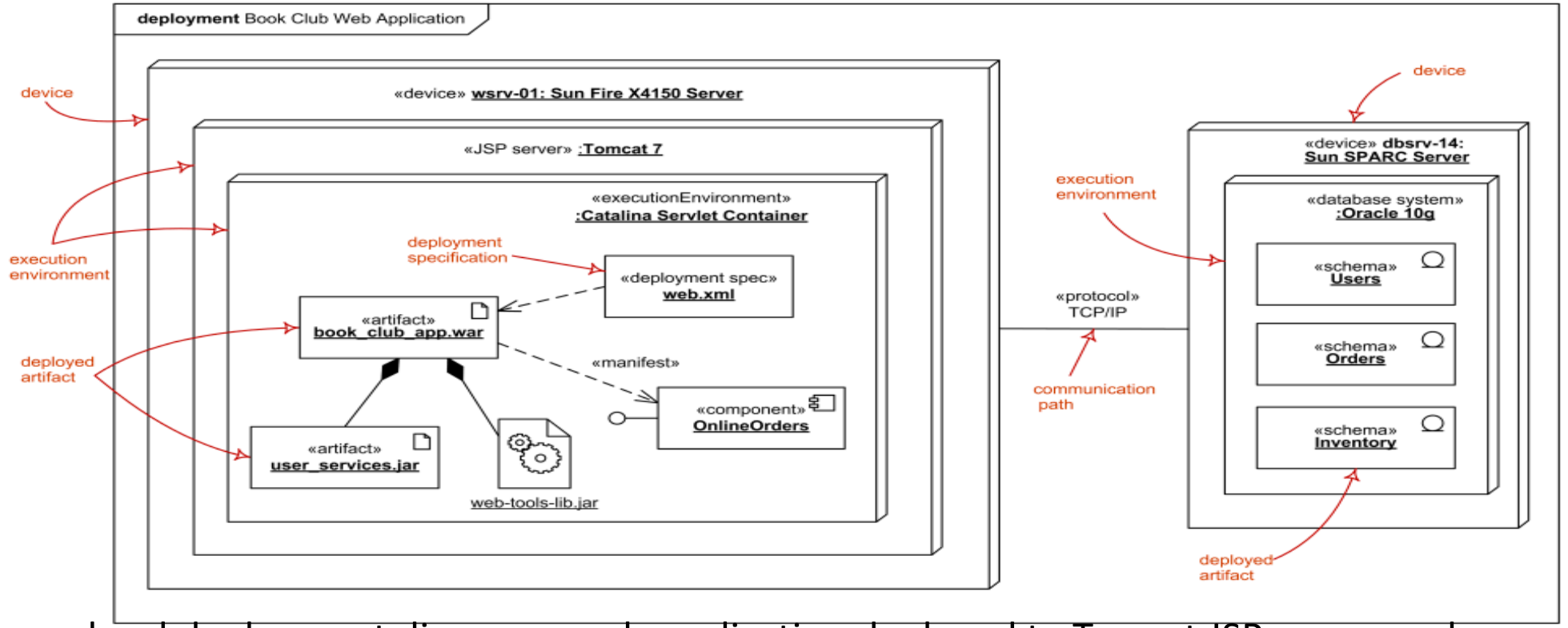
- **Mức đặc tả** (còn gọi là mức kiểu) biểu đồ triển khai cho thấy một số tổng quan về việc triển khai đồ tạo tác đến các mục tiêu triển khai mà không cần tham chiếu đến các trường hợp cụ thể của đồ vật hoặc nút.



Mức đặc tả biểu đồ triển khai - ứng dụng web được triển khai tới máy chủ Tomcat JSP và các lược đồ cơ sở dữ liệu - tới hệ thống CSDL

Biểu đồ triển khai

- Mức thể hiện biểu đồ triển khai cho thấy việc triển khai các phiên bản của đồ tạo tác đến các trường hợp cụ thể của mục tiêu triển khai. Nó có thể hữu ích, chẳng hạn như chỉ ra sự khác biệt trong việc triển khai với phát triển, môi trường dàn dựng hoặc sản xuất với tên/định danh cụ thể cho các dịch vụ hoặc thiết bị triển khai



Instance level deployment diagram - web application deployed to Tomcat JSP server and database shemas - to database system

Đồ án

- Chia nhóm 4-5 sinh viên
- Mỗi nhóm chọn một bài toán
- Xây dựng
  - Biểu đồ gói
  - Biểu đồ thành phần
  - Biểu đồ triển khai