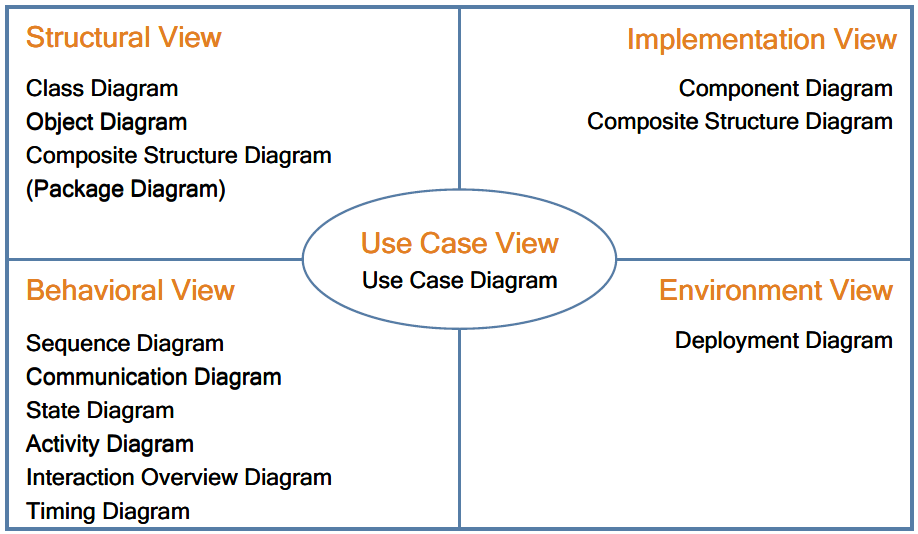
**NHẬP MÔN KỸ THUẬT PHẦN MỀM**

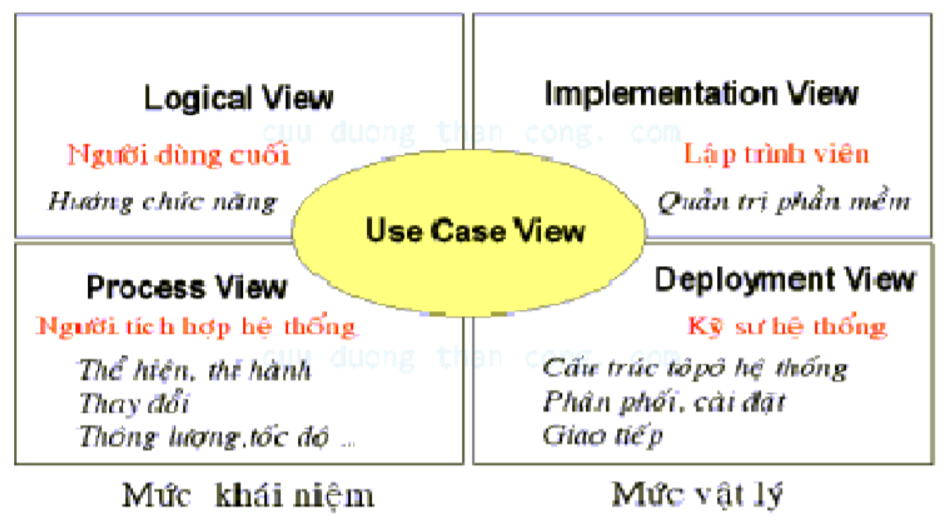
**Bài 08 - Lý thuyết 05:**

**Ngôn ngữ mô hình hóa thống nhất - Unified Modeling Language (UML)**

1. **Nội dung:**
   * Khái niệm về UML
   * Phân tích về thiết kế UML
2. **Khái niệm về UML**
   1. **Unified Modeling Language (UML)**
   * UML là ngôn ngữ đồ họa tiêu chuẩn
   * Sử dụng cho rất nhiều lĩnh vực kỹ thuật trong đó có công nghệ phần mềm.
   * UML là ngôn ngữ ký hiệu sử dụng trong lập trình hướng đối tượng.
   * UML biểu diễn, quy định cụ thể, xây dựng và tài liệu về hệ thống phần mềm
   * Vì sao nên sử dụng UML? => Vì UML là:
   * Ký hiệu đồ họa
   * Một bức tranh đáng giá ngàn lời nói
   * Ngôn ngữ giao tiếp chuẩn
   * Cung cấp nhiều sơ đồ để chụp kiến trúc khác nhau
   * Thúc đẩy việc tái sử dụng các thành phần
   * UML là một ngôn ngữ tiêu chuẩn cho phép biểu diễn, quy định cụ thể, xây dựng và tài liệu về hệ thống phần mềm
   1. **UML định nghĩa 13 sơ đồ mô tả 4+1 quan điểm kiến trúc**
   * Mô hình khung nhìn kiến trúc 4 + 1 được đề xuất bởi Philippe Kruchten, IBM
   * (4+1 architectural views model was proposed by Philippe Kruchten, IBM)

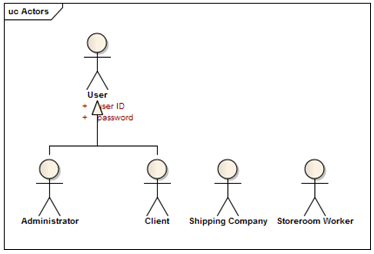


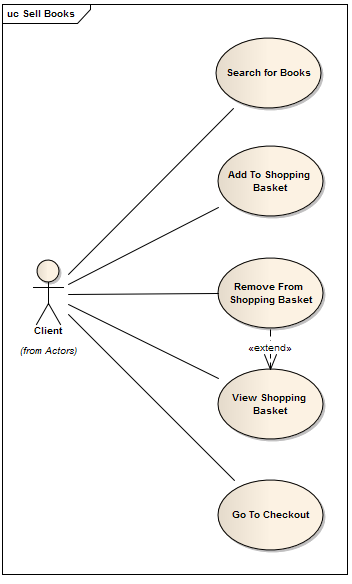
* 1. Sơ đồ lớp (Class Diagram)
  2. Sơ đồ đối tượng (Object Diagram)
  3. Sơ đồ cộng tác (Composite Structure Diagram)
  4. Sơ đồ gói (Package Diagram)
  5. **Sơ đồ tình huống sử dụng (Use Cases Diagram)**
  6. Sơ đồ trình tự (Sequence Diagram)
  7. Sơ đồ liên lạc (Communication Diagram)
  8. Sơ đồ trạng thái (State Diagram)
  9. Sơ đồ hoạt động (Activity Diagram)
  10. Sơ đồ tương tác (Interaction Overview Diagram - UML 2.0)
  11. Sơ đồ phối hợp thời gian (Timing Diagram - UML 2.0)
  12. Sơ đồ thành phần (Component Diagram)
  13. *Sơ đồ cộng tác (Composite Structure Diagram)*
  14. Sơ đồ triển khai (Deployment Diagram)

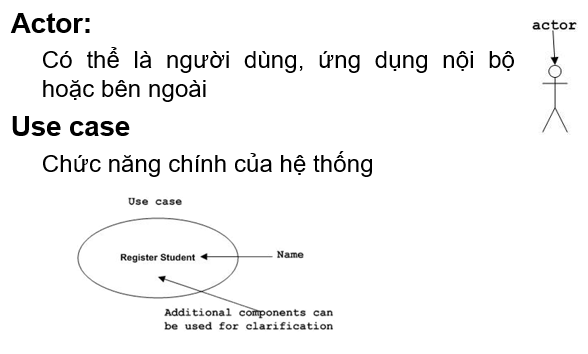


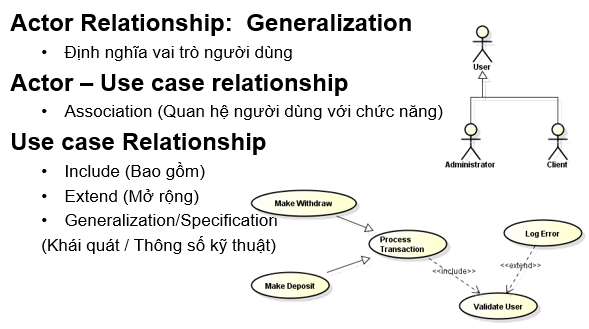
1. **Phân tích về thiết kế UML**
   1. **UML – Use case diagram**
   * Mô tả các chức năng được cung cấp bởi hệ thống
   * Mỗi trường hợp sử dụng cho một chức năng riêng biệt của sự tương tác giữa người sử dụng (người hoặc máy) và hệ thống.
   * Bao gồm: actors, use cases, và relationships
     + ***Ví dụ:***

*BookShop Sample: Actors + Sell Book Use Cases (bán sách)*

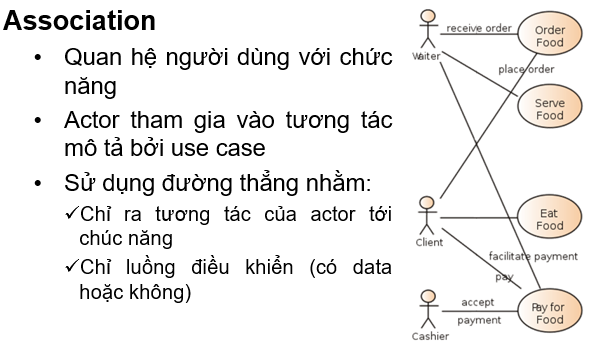


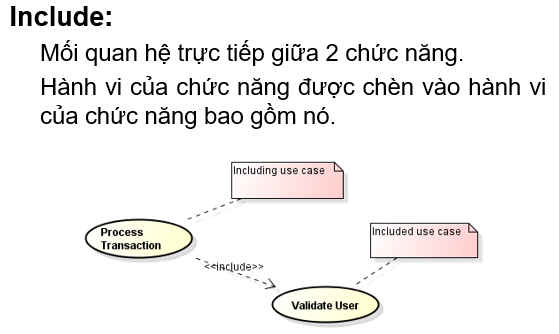


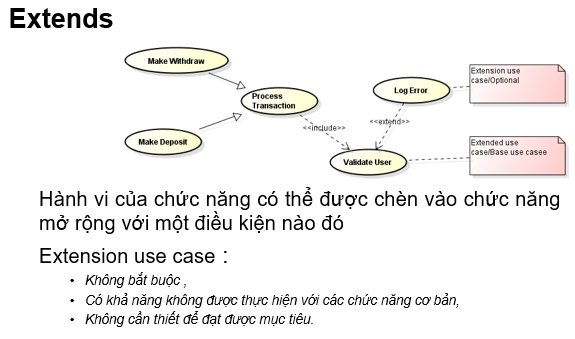


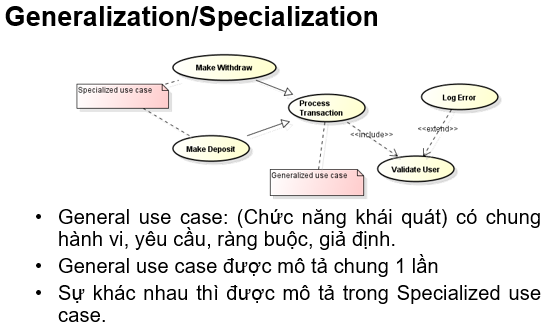


* + ***Actor relationship: generalization:*** tức là tổng quát hoá: như hình bên: khách hàng và người quản trị hê thống sẽ được tổng quát hoá = 1 actor là người sử dụng. Việc sử dụng relationship này rất hữu dụng khi bạn muốn định nghĩa những chức năng trùng nhau của actor, ví dụ: cả khách hàng và người quản trị hê thống đều phải login và có ***account/password***





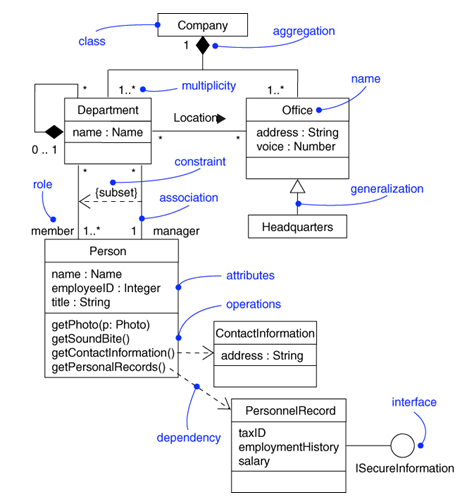




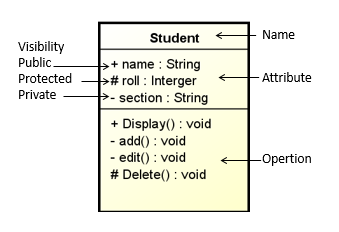
* + - ***Vẽ use case diagram, phải xác định:***
  + Các chức năng của hệ thống chính là use case
  + Actor (người dùng)
  + Quan hệ giữa use cases và actors
    - ***Use case diagram tốt phải:***
  + Tên của use case rất quan trọng.
    - Đặt tên có thể xác định các chức năng được thực hiện
    - Đưa tên phù hợp cho Actor.
  + Thể hiện mối quan hệ và sự phù thuộc rõ ràng.
  + Không bao gồm tất cả các loại quan hệ
  + Mục đích chính của Use case diagram để nắm bắt yêu cầu.
  + Khi cần thiết làm rõ một số điểm quan trong cần có ghi chú

**THẢO LUẬN MỞ VỀ USE CASE**

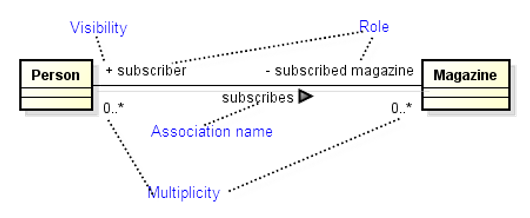
* 1. **UML – Class diagram**
     + ***Mô tả cấu trúc hệ thống***
  + Mô tả các Class của hệ thống, thuộc tính của chúng và mối quan hệ giữa các Class.
    - ***Mục đích:***
  + Phân tích và thiết kế các View tỉnh của ứng dụng.
  + Thể hiện sự thống nhất giữa các thành phần trong View tỉnh.
  + Mô tả sự phản hồi của hệ thống.
  + Cơ sở cho các thành phần và triển khai sơ đồ.



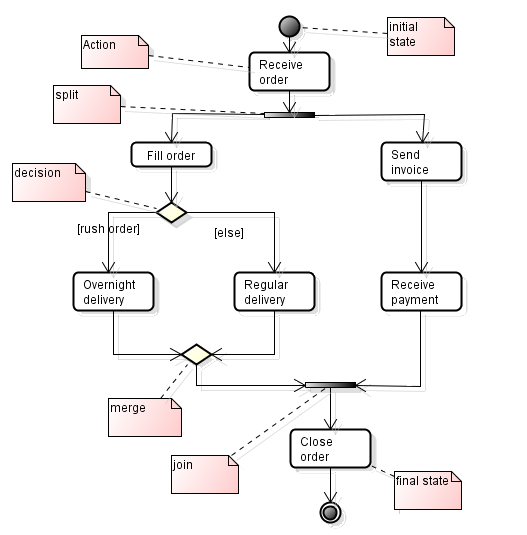
* + - ***Class***
  + Đại diện cho đối tượng (Object)
  + Có các thuộc tính
  + Và các function.



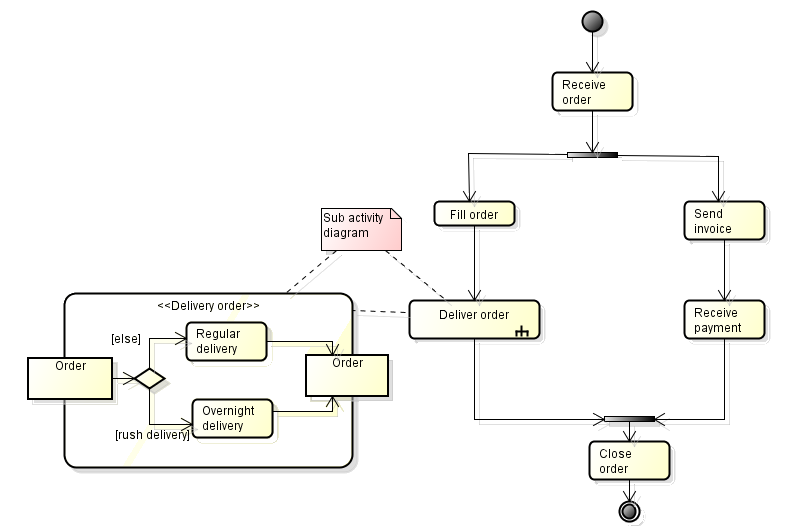
* + - ***Thể hiện mối quan hệ giữa các Class***



* + - ***Class diagram tốt phải:***
  + Tên class, tên diagram nên mô tả được các đối tượng tồn tại trong hệ thống.
  + Thành phần, mối quan hệ nên được định nghĩa rõ ràng.
  + Attributes và methods nên được định nghĩa rõ ràng.
  + Trong mỗi Class: Tối thiểu các thuộc tính.
    - Những thuộc tính không cần thiết có thể làm sơ đồ phức tạp hơn.
  + Dùng các ghi chú khi được yêu cầu.
  + Diagram có thể được vẽ trên giấy thường và làm lại nhiều lần để làm cho nó chính xác hơn.
  1. **UML – Activity diagram**
  + Activity diagrams là sơ đồ quy trình công việc từng bước của các hành động với sự hỗ trợ cho sự lựa chọn, lặp đi lặp lại và đồng thời
  + Activity diagrams thể hiện dòng chảy hành động được kiểm soát như thế nào.
  + Activity diagrams được tạo thành từ các shape, kết nối bằng arrow (mũi tên):
    - Rounded rectangles: actions (hành động)
    - Diamonds: lựa chọn
    - Bars: bắt đầu (split) hoặc kết thúc (join) của hành động.
    - A black circle: Bắt đầu (initial state) của workflow
    - An encircled black circle: kết thúc (final state) của workflow.
  + ***Example of an activity diagram***



* + ***Sub activity diagram***



* 1. **Demo => Sử dụng Visio vẽ sơ đồ**
  2. **Chuẩn bị Workshop 4**
  + Sử dụng công cụ Visio thiết kế Class diagram và Active diagram theo Requirement của đề tài