


ÔN TẬP XÂY DỰNG PHẦN MỀM THEO MÔ HÌNH PHÂN LỚP

1. Trình bày ý nghĩa của 2 loại mô hình : sơ đồ lớp khái niệm và sơ đồ lớp thiết kế.
 - Nêu sự liên hệ giữa 2 mô hình này.
2. Trình bày ý nghĩa của 2 loại sơ đồ : sơ đồ hoạt động (activity diagram) và sơ đồ tuần tự (sequence diagram).

Activity Diagram

- Biểu đồ hoạt động nắm bắt hành động và các kết quả của chúng.
- Biểu đồ hoạt động tập trung vào công việc được thực hiện trong khi thực thi một thủ tục (hàm), các hoạt động trong một lần thực thi một trường hợp sử dụng hoặc trong một đối tượng.
- Biểu đồ hoạt động là một biến thể của biểu đồ trạng thái và có một mục tiêu tương đối khác, đó là nắm bắt hành động (công việc và những hoạt động phải được thực hiện) cũng như kết quả của chúng theo sự biến đổi trạng thái. Các trạng thái trong biểu đồ hoạt động (được gọi là các trạng thái hành động) sẽ chuyển sang giai đoạn kế tiếp khi hành động trong trạng thái này đã được thực hiện xong (mà không xác định bất kỳ một sự kiện nào theo như nội dung của biểu đồ trạng thái).
- Một sự điểm phân biệt khác giữa biểu đồ hoạt động và biểu đồ trạng thái là các hành động của nó được định vị trong các *luồng* (swimlane). Một luồng sẽ gom nhóm các hoạt động, chú ý tới khái niệm người chịu trách nhiệm cho chúng hoặc chúng nằm ở đâu trong một tổ chức. Một biểu đồ hoạt động là một phương pháp bổ sung cho việc miêu tả tương tác, đi kèm với trách nhiệm thể hiện rõ các hành động xảy ra như thế nào, chúng làm gì (thay đổi trạng thái đối tượng), chúng xảy ra khi nào (chuỗi hành động), và chúng xảy ra ở đâu (luồng hành động).

 *Biểu đồ hoạt động có thể được sử dụng cho nhiều mục đích khác nhau, ví dụ như:*

- Để nắm bắt công việc (hành động) sẽ phải được thực thi khi một thủ tục được thực hiện. Đây là tác dụng thường gặp nhất và quan trọng nhất của biểu đồ hoạt động.
- Để nắm bắt công việc nội bộ trong một đối tượng.
- Để chỉ ra một nhóm hành động liên quan có thể được thực thi ra sao, và chúng sẽ ảnh hưởng đến những đối tượng nằm xung quanh chúng như thế nào.
- Để chỉ ra một trường hợp sử dụng có thể được thực thể hóa như thế nào, theo khái niệm hành động và các sự biến đổi trạng thái của đối tượng.

- Để chỉ ra một doanh nghiệp hoạt động như thế nào theo các khái niệm công nhân (tác nhân), qui trình nghiệp vụ (workflow), hoặc tổ chức và đối tượng (các khía cạnh vật lý cũng như tri thức được sử dụng trong doanh nghiệp).
- Biểu đồ hoạt động có thể được coi là một loại Flow chart. Điểm khác biệt là Flow Chart bình thường ra chỉ được áp dụng đối với các qui trình tuần tự, biểu đồ hoạt động có thể xử lý cả các qui trình song song.

Sequence Diagram

- Biểu đồ trình tự được sử dụng chủ yếu để thể hiện mối tương tác giữa các đối tượng trong thứ tự trình tự mà các mối tương quan này xảy ra. Giống như biểu đồ lớp, các chuyên viên phát triển thường nghĩ các biểu đồ trình tự là dành riêng đối với họ. Tuy nhiên, nhân viên kinh doanh của một tổ chức có thể tìm ra biểu đồ trình tự hiệu quả để trao đổi các công việc gần đây hoạt động thế nào bằng cách trình bày các đối tượng công việc đa dạng tác động với nhau thế nào. Bên cạnh tập hợp tài liệu các sự kiện của một tổ chức, một biểu đồ trình tự ở cấp độ kinh doanh có thể được sử dụng như là một tài liệu cần thiết để trao đổi các yêu cầu cho việc thực thi hệ thống trong tương lai. Trong giai đoạn các yêu cầu của một dự án, các chuyên viên phân tích có thể tận dụng các trường hợp tới mức độ cao hơn bằng cách đưa ra một mức cải tiến chính thức hơn. Khi xảy ra như vậy, sử dụng các tình huống được cải tiến vào một hoặc nhiều các biểu đồ trình tự.
- Nhân viên kỹ thuật của một tổ chức có thể tìm các biểu đồ trình tự hiệu quả qua việc chứng minh một hệ thống tương lai sẽ hoạt động thế nào. Trong giai đoạn thiết kế, các chuyên viên kiến trúc và chuyên viên phát triển có thể sử dụng biểu đồ để ép các mối tương quan đối tượng của hệ thống, do đó lọc ra các thiết kế hệ thống tổng thể.
- Một trong các sử dụng chính của biểu đồ trình tự là trong sự chuyển đổi từ các yêu cầu được thể hiện như là sử dụng các tình huống tới các cải tiến tiếp theo và mức chính thức hơn của sự cải tiến. Sử dụng các trường hợp được cải tiến thường xuyên hơn vào một hoặc nhiều biểu đồ trình tự. Ngoài các sử dụng trong việc thiết kế hệ thống mới, biểu đồ trình tự có thể được sử dụng để dẫn chứng các đối tượng trong một hệ thống đang tồn tại (gọi là "hợp lệ") gần đây tương tác thế nào. Sự dẫn chứng này rất hữu ích khi chuyển đổi một hệ thống tới nhân sự hoặc tổ chức khác.

3. Trình bày ý nghĩa và chức năng của mỗi lớp trong kiến trúc 3 lớp. Cho ví dụ minh họa.

- **GUI layer**: Lớp này làm nhiệm vụ giao tiếp với người dùng cuối để thu thập dữ liệu và hiển thị kết quả/dữ liệu thông qua các thành phần trong giao diện người sử dụng.
- **Business Logic Layer**: đây là layer xử lý chính các dữ liệu trước khi được đưa lên hiển thị trên màn hình hoặc xử lý các dữ liệu trước khi chuyển xuống Data Access Layer để lưu dữ liệu xuống cơ sở dữ liệu. Đây là nơi để kiểm tra ràng buộc các yếu tố nghiệp vụ, tính toán, xử lý các yêu cầu và lựa chọn kết quả trả về cho GUI Layers.

- **Data Access Layer:** Lớp này thực hiện các nghiệp vụ liên quan đến lưu trữ và truy xuất dữ liệu của ứng dụng như đọc, lưu, cập nhật cơ sở dữ liệu,..

 **Ví dụ: Mô tả hoạt động của mô hình 3-layer với ứng dụng liệt kê danh sách điểm sinh viên:**

- Từ màn hình form quản lý sinh viên gồm có 1 combobox chọn lớp, 1 gridview để hiển thị danh sách sinh viên và 1 button để thực hiện lệnh liệt kê danh sách.
 - (1) Người dùng chọn combobox lớp trên GUI và ấn button liệt kê
 - (2) GUI kiểm tra yêu cầu chọn combobox hợp lệ và gửi mã lớp (**) vừa chọn sang BUS xử lý yêu cầu hiển thị danh sách điểm sinh viên
 - (4) Tại BUS vì yêu cầu từ GUI khá đơn giản nên BUS sẽ không xử lý gì mà sẽ gửi mã lớp sang DAL lấy danh sách điểm.
 - (5) Tại DAL sau khi đã nhận được yêu cầu lấy danh sách điểm từ mã lớp, DAL sẽ tương tác với hệ quản trị CSDL (6) qua các lệnh mở tập tin, kết nối, truy vấn,... để lấy được danh sách điểm (7) với mã số yêu cầu, DAL tiếp tục gửi danh sách (**) này sang BUS để xử lý (7)
 - (8) Tại BUS sau khi nhận được danh sách điểm từ DAL gửi sang, BUS thực hiện nghiệp vụ của mình bằng cách tính điểm trung bình, kết luận đậu/rớt của từng sinh viên (tất cả xử lý về mặt nghiệp vụ), sau đó gửi danh sách điểm đã xử lý (**) sang GUI để hiển thị (9)
 - (9) 1 lần nữa GUI có thể kiểm tra tính hợp lệ của dữ liệu và hiển thị thông tin và thông báo lên màn hình cho người dùng (10)

(**) Trong 1 số trường hợp vì lượng thông tin gửi nhiều, ví dụ như 1 sinh viên gồm nhiều thuộc tính như họ tên, tuổi, ngày sinh,... ta có thể dùng DTO để chuyển đổi tượng hoặc danh sách đối tượng giữa các tầng với nhau cho tiện dụng.

4. Nêu các thành phần của hệ thống khi xây dựng cấu trúc đa giao diện.

- Các công cụ để xây dựng hệ thống.

5. Trình bày các mô hình phân lớp mà bạn tìm hiểu (MVC, MVP, MVVM, ...)

 **MVC**

• **Tổng quan:**

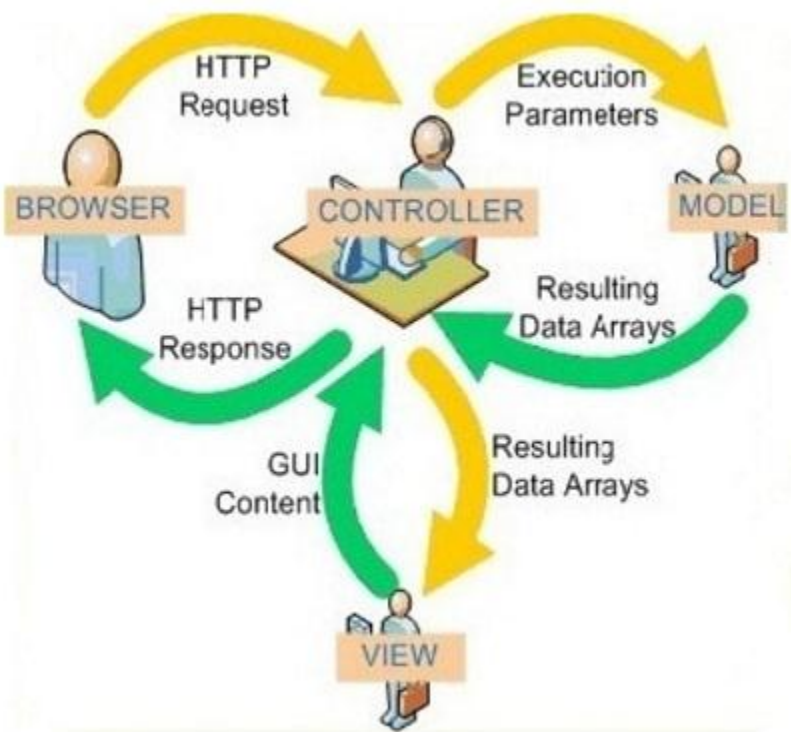
Vào những năm 70 của thế kỷ 20, tại phòng thí nghiệm Xerox PARC ở Palo Alto. Sự ra đời của giao diện đồ họa (Graphical User Interface - GUI) và lập trình hướng đối tượng (Object Oriented Programming - OOP) cho phép lập trình viên làm việc với những thành phần đồ họa như những đối tượng đồ họa có thuộc tính và phương thức riêng của nó. Không dừng lại ở đó, những nhà nghiên cứu ở Xerox PARC còn đi xa hơn nữa khi họ cho ra đời cái gọi là kiến trúc MVC (viết tắt của Model – View – Controller).

Kiến trúc này ngày càng được phát triển và hoàn thiện nhằm giải quyết các vấn đề phát sinh cũng như các giải pháp cho quá trình phát triển phần mềm.

- **Các thành phần trong MVC**

Trong kiến trúc này, hệ thống được chia thành 3 tầng tương ứng đúng với tên gọi của nó (Model – View – Controller). Ở đó nhiệm vụ cụ thể của các tầng được phân chia như sau:

1. **Model (Tầng dữ liệu):** là một đối tượng hoặc một tập hợp các đối tượng biểu diễn cho phần dữ liệu của chương trình. Nó được giao nhiệm vụ cung cấp dữ liệu cho cơ sở dữ liệu và lưu dữ liệu vào các kho chứa dữ liệu. Tất cả các nghiệp vụ logic được thực thi ở Model. Dữ liệu vào từ người dùng sẽ thông qua View đến Controller và được kiểm tra ở Model trước khi lưu vào cơ sở dữ liệu. Việc truy xuất, xác nhận, và lưu dữ liệu là một phần của Model



Hình 1: Luồng xử lý của mô hình MVC

2. **View (Tầng giao diện):** là phần giao diện với người dùng, bao gồm việc hiện dữ liệu ra màn hình, cung cấp các menu, nút bấm, hộp đối thoại, chọn lựa ..., để người dùng có thể thêm, xóa, sửa, tìm kiếm và làm các thao tác khác đối với dữ liệu trong hệ thống.. Thông thường, các thông tin cần hiển thị được lấy từ thành phần Models.

3. **Controller (Tầng điều khiển):** là phần điều khiển của ứng dụng, điều hướng các nhiệm vụ (task) đến đúng phương thức (method) có chức năng xử lý nhiệm vụ đó. Nó

chịu trách nhiệm xử lý các tác động về mặt giao diện, các thao tác đối với models, và cuối cùng là chọn một view thích hợp để hiển thị ra màn hình.

- **Ưu điểm và nhược điểm:**

- 1. Ưu điểm:**

Phát triển phần mềm: Có tính chuyên nghiệp hóa, có thể chia cho nhiều nhóm được đào tạo nhiều kỹ năng khác nhau, từ thiết kế mỹ thuật cho đến lập trình đến tổ chức database. Giúp phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp..

Bảo trì: Với các lớp được phân chia theo như đã nói, thì các thành phần của một hệ thống dễ được thay đổi, nhưng sự thay đổi có thể được cô lập trong từng lớp, hoặc chỉ ảnh hưởng đến lớp ngay gần kề của nó, chứ không phát tán náo loạn trong cả chương trình.

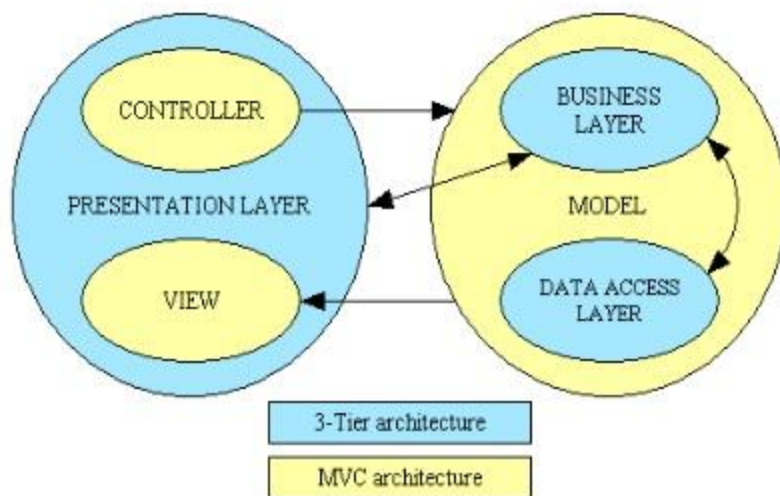
Mở rộng: Với các lớp được chia theo ba lớp như đã nói, việc thêm chức năng vào cho từng lớp sẽ dễ dàng hơn là phân chia theo cách khác.

- 2. Nhược điểm:**

Đối với dự án nhỏ việc áp dụng mô hình MC gây cồng kềnh, tốn thời gian trong quá trình phát triển.

Tốn thời gian trung chuyển dữ liệu của các tầng

- **So sánh mô hình MVC với mô hình 3 lớp**



Hình 2: So sánh mô hình MVC với mô hình 3 lớp

- 1. Điểm giống:**

Tách rời programming core/business logic ra khỏi những phụ thuộc về tài nguyên và môi trường.

Presentation Layer (PL) thể hiện giống như chức năng của View và Controller. Business Layer (BL) và Data Access Layer (DL) thể hiện giống như chức năng của Model. Như thế nhìn ở góc độ này, thì MVC tương đương với 3-layer (tất nhiên có chồng chéo như hình vẽ)

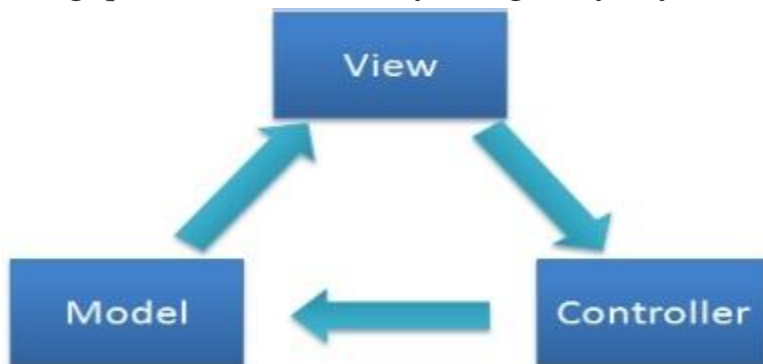
2. Điểm khác:

Trong mô hình 3 lớp, quá trình đi theo chiều dọc, bắt đầu từ PL, sang BL, rồi tới DL, và từ DL, chạy ngược lại BL rồi quay ra lại PL.



Hình 3: Mô hình 3 lớp

Còn trong mô hình MCD, dữ liệu được nhận bởi View, View sẽ chuyển cho Controller cập nhật vào Model, rồi sau đó dữ liệu trong Model sẽ được đưa lại cho View mà không thông qua Controller, do vậy luồng xử lý này có hình tam giác.



Hình 4: Mô hình MCD

MVP

- **Tổng quan**

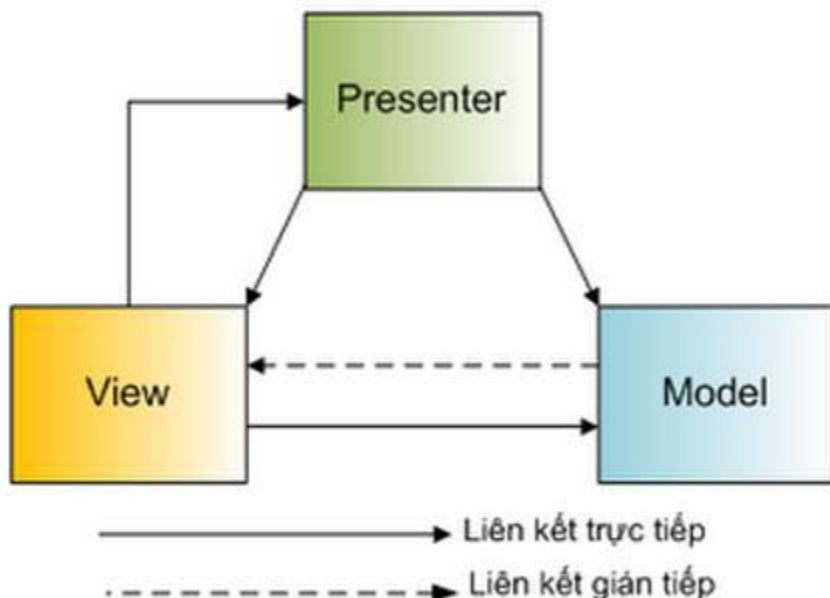
Mẫu kiến trúc Dolphin Smalltalk Model-View-Presenter chia ứng dụng thành các phần dữ liệu (data), trình bày (presentation) và các xử lý logic thuộc phần trình bày (presentation logic) thành những thành phần riêng biệt.

- **Lịch sử**

Phiên bản Dolphin Smalltalk Model-View-Presenter (gọi tắt là Dolphin MVP) là phiên bản của MVP được xây dựng dựa trên phiên bản Taligent MVP xuất hiện trước đó.

Dolphin MVP được xây dựng về cơ bản bên ngoài tương tự như MVC cổ điển nhưng khác nhau ở vai trò của Controller và Presenter.

- **Cấu trúc**



- **Các thành phần**

Model chứa dữ liệu và các tính toán xử lý logic để giải quyết vấn đề mà phần mềm hướng tới (business logic).

View là thành phần đảm nhận trình bày từ những dữ liệu của Model. View bao gồm những gì thể hiện trên màn hình như các control, form, widget,...

Presenter là thành phần đảm nhận các xử lý về trình bày mà nó cần đến sự tương tác trên dữ liệu.

- **Phối hợp các thành phần**

Trong khi vai trò của Model không thay đổi so với MVC cổ điển. Thành phần View của Dolphin MVP cũng thực hiện việc trình bày từ nội dung của Model và gắn kết với Model thông qua Observer Pattern. Tuy nhiên, nó thực hiện bắt các sự kiện xảy ra ngay bên trong nó. Một vài trường hợp đơn giản như TextView, dữ liệu nhập được xử lý trực tiếp và cập nhật thay đổi trực tiếp vào Model còn hầu hết các trường hợp khác, việc xử lý được chuyển giao cho Presenter đảm trách khi cần thao tác đến Model.

Trong khi View đảm nhận trình bày thì Presenter đảm trách cách Model được thao tác và thay đổi như thế nào bởi giao diện người dùng. Presenter là nơi chứa các xử lý đặc trưng của ứng dụng (application logic so với business logic của Model). Một điểm đáng chú ý khác là Presenter có khả năng thao tác trực tiếp lên View mà nó gắn kết, điều này khác biệt với phiên bản Taligent MVP xuất hiện trước đó.

- **So sánh Dolphin MVP và MVC cổ điển**

Điểm khác biệt cơ bản của Dolphin MVP và MVC cổ điển là sự khác nhau về vai trò của

Presenter và Controller, nó cũng dẫn đến sự khác nhau về vai trò giữa hai thành phần View. Trong Dolphin MVP, sự hiện diện của Controller bị loại bỏ, thay vào đó, việc xử lý các dữ liệu input được View đảm nhận và được chuyển cho Presenter khi có yêu cầu tương tác đến Model.

MVVM

- **View:** Tương tự như trong mô hình MVC, View là phần giao diện của ứng dụng để hiển thị dữ liệu và nhận tương tác của người dùng. Một điểm khác biệt so với các ứng dụng truyền thống là View trong mô hình này tích cực hơn. Nó có khả năng thực hiện các hành vi và phản hồi lại người dùng thông qua tính năng binding, command.
- **Model:** Cũng tương tự như trong mô hình MVC. Model là các đối tượng giúp truy xuất và thao tác trên dữ liệu thực sự.
- **ViewModel:** Lớp trung gian giữa View và Model. ViewModel có thể được xem là thành phần thay thế cho Controller trong mô hình MVC. Nó chứa các mã lệnh cần thiết để thực hiện data binding, command.

Một điểm cần lưu ý là trong mô hình MVVM, các tầng bên dưới sẽ không biết được các thông tin gì về tầng bên trên nó. Như hình minh họa dưới đây:

