|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY**  **SCHOOL OF ELECTRONICS AND TELECOMMUNICATIONS**  logo_128  **BÁO CÁO ĐỒ ÁN I**  **Sử dụng mô hình MVC cho chương trình Shape**   |  |  |  | | --- | --- | --- | | **Nhóm sinh viên thực hiện**: | Nhóm 3 |  | | **Tên sinh viên** | MSSV | Mã lớp | | Nguyễn Văn Đạt | 20172458 | 696061 | | Đào Đức Dũng | 20172492 | 696061 | | Vũ Đức Thái | 20172804 | 696061 | | **Giảng viên hướng dẫn:** | PGS-TS. Nguyễn Đức Minh | |   Hanoi, 5-2020 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY**  **SCHOOL OF ELECTRONICS AND TELECOMMUNICATIONS**  logo_128  **BÁO CÁO ĐỒ ÁN I**  **Sử dụng mô hình MVC cho chương trình Shape**   |  |  |  | | --- | --- | --- | | **Nhóm sinh viên thực hiện**: | Nhóm 3 |  | | **Tên sinh viên** | MSSV | Mã lớp | | Nguyễn Văn Đạt | 20172458 | 696061 | | Đào Đức Dũng | 20172492 | 696061 | | Vũ Đức Thái | 20172804 | 696061 | | **Giảng viên hướng dẫn:** | PGS-TS. Nguyễn Đức Minh | |   Hanoi, 5-2020 |

# LỜI MỞ ĐẦU

Ngôn ngữ lập trình C++ là một trong những ngôn ngữ lập trình hướng đối tượng mạnh và phổ biến hiện nay do tính mềm dẻo và đa năng của nó. Không chỉ các ứng dụng được viết trên C++ mà cả những chương trình hệ thống lớn đều được viết hầu hết trên C++. C++ là ngôn ngữ lập trình được viết trên nền tảng của C, không những khắc phục được một số đặc điểm của ngôn ngữ C mà quan trọng hơn, C++ cung cấp cho người sử dụng một phương tiện lập trình theo kỹ thuật mới: lập trình hướng đối tượng. Đây là kỹ thuật lập trình được sử dụng hầu hết trong các ngôn ngữ mạnh hiện nay, đặc biệt là các ngôn ngữ hoạt động trong môi trường Windows như Microsoft Access, Visual Basic, Visual Foxpro… . Bên cạnh đó chúng em thiết kế giao diện đồ họa trong C++, sử dụng framework hỗ trợ Qt theo mô hình MVC (Model – View - Controller). Trong quá trình thực hiện cũng như nội dung bài báo cáo có không tránh khỏi những thiếu sót, chúng em rất mong được sự đóng góp của thầy và các để bài báo cáo được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn thầy Nguyễn Đức Minh đã tạo điều kiện hướng dẫn và cung cấp những tài liệu cần thiết trong quá trình hoàn thành bài báo cáo này.

# DANH MỤC HÌNH ẢNH

[Hình 1. 1 Mô hình M-V-C 1](file:///C:\Users\Admin\Downloads\Báo%20cáo%20Đồ%20án%201.docx#_Toc42849850)

[Hình 1. 2 Sơ đồ UML của mô hình MVC 3](#_Toc42849851)

[Hình 2. 1 Model cho Shape và Edge 5](#_Toc42849863)

[Hình 2. 2 Class Controller 6](#_Toc42849864)

[Hình 2. 3 View (Giao diện với người dùng) 7](#_Toc42849865)

**MỤC LỤC**

[LỜI MỞ ĐẦU i](#_Toc42849807)

[DANH MỤC HÌNH ẢNH ii](#_Toc42849808)

[CHƯƠNG 1. TỔNG QUAN VỀ MÔ HÌNH M-V-C 1](#_Toc42849809)

[1.1 Mục đích của MVC pattern 1](#_Toc42849810)

[1.2 Khái niệm Model (M). 1](#_Toc42849811)

[1.3 Khái niệm View (V). 2](#_Toc42849812)

[1.4 Khái niêm Controller (C). 2](#_Toc42849813)

[1.5 Ưu nhược điểm của mô hình MVC 2](#_Toc42849814)

[1.6 Sơ đồ UML của mô hình MVC 3](#_Toc42849815)

[CHƯƠNG 2. MÔ TẢ HOẠT ĐỘNG CỦA MVC. 5](#_Toc42849816)

[2.1 Tạo Model cho Shape và Edge . 5](#_Toc42849817)

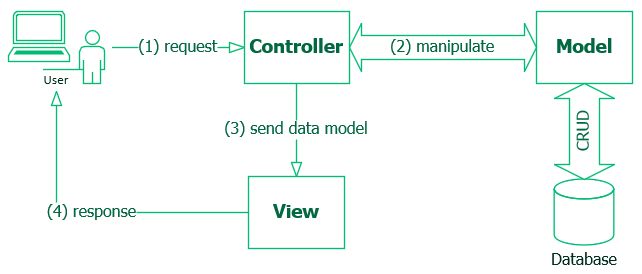
[2.2 Tạo Controller cho chương trình. 6](#_Toc42849818)

[2.3 Tạo View cho chương trình. 7](#_Toc42849819)

[KẾT LUẬN 8](#_Toc42849820)

[TÀI LIỆU THAM KHẢO 9](#_Toc42849821)

CHƯƠNG 1. TỔNG QUAN VỀ MÔ HÌNH M-V-C

Model – View – Controller (MVC) là một mẫu thiết kế nhằm mục tiêu chia tách phần Giao diện và Code để dễ quản lý, phát triển và bảo trì. MVC chia ứng dụng phần mềm ra làm 3 phần có tương tác với nhau là Model (Dữ liệu), View (Giao diện), Controller ( Code điều khiển tương tác giữa Model và View).

Hình 1. Mô hình M-V-C

Hình 1. 1 thể hiện mô hình của MVC bao gồm ba khối chính là Model, Controller và View, các khối này tác động với nhau theo các chiều nhất định.

Với việc sử dụng mô hình MVC giúp cho mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với các thành phần còn lại.

## 1.1 Mục đích của MVC pattern

MVC Pattern là mô hình thiết kế cho phép thay đổi thành phần dữ liệu (Model) sẽ không ảnh hưởng nhiều đến giao diện của người dùng vì mô hình đưa ra Model để không cho người dùng thao tác trực tiếp vào dữ liệu mà phải thông qua Model, do vậy cho dù dữ liệu vật lý thay đổi cấu trúc nhưng cấu trúc của Model cho việc truy cập, xử lý, lưu trữ dữ liệu sẽ không bị ảnh hưởng.

## 1.2 Khái niệm Model (M).

Model (M) là một bộ phận trong mô hình MVC có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng. Bộ phận này là một cầu nối giữa hai thành phần đó là View và Controller. Model thể hiện dưới hình thức là một cơ sở dữ liệu, nó thể hiện rõ các thao tác với cơ sở dữ liệu như cho phép xem, truy xuất, xử lý dữ liệu,….

## 1.3 Khái niệm View (V).

View là thành phần giao diện (theme) dành cho người sử dụng. Nơi mà người dùng có thể lấy được thông tin dữ liệu của MVC thông qua các thao tác truy vấn như tìm kiếm. View cũng có chức năng ghi nhận hoạt động của người dùng để tương tác với Controller. Tuy nhiên, View không có mỗi quan hệ trực tiếp nào với Controller, cũng không lấy được dữ liệu từ Controller mà chỉ hiển thị yêu cầu chuyền cho Controller.

Ví dụ: Nút “Delete” được tạo bởi View khi người dùng nhấn vào đó sẽ có một hành động được thực hiện trong Controller.

## 1.4 Khái niêm Controller (C).

Controller là một bộ phận có nhiệm vụ xử lý các yêu cầu từ phía người dùng đưa đến thông qua View. Từ đó, Controller đưa ra dữ liệu phù hợp với người dùng. Bên cạnh đó, Controller còn có chức năng kết nối với Model.

## 1.5 Ưu nhược điểm của mô hình MVC

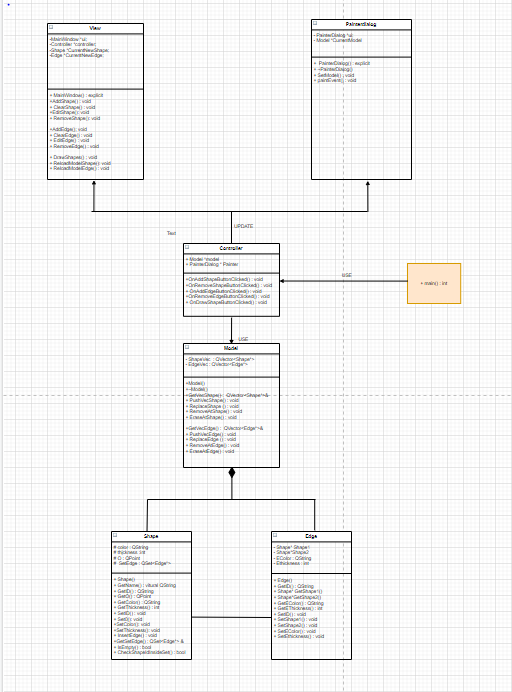
\**Ưu điểm:*

* Kiểm tra dễ dàng: Với MVC, lập trình viên có thể dễ dàng kiểm tra, rà soát lỗi phần mềm trước khi tới tay người tiêu dùng, đảm bảo chất lượng và độ uy tín cao hơn.
* Chức năng Soc (Separation of Concern): Chức năng này cho phép lập trình viên phân tách rõ ràng các phần như Model, giao diện, data, nghiệp vụ.
* Tính kết hợp: Việc tích hợp ở mô hình MVC cho phép người lập trình viên thoải mái viết code trên nền tảng ngôn ngữ lập trình.
* Đơn giản: Đây là một mô hình với kết cấu tương đối đơn giản. Dù người lập trình viên không có quá nhiều chuyên môn cũng có thể sử dụng.

\**Nhược điểm:*

MVC thường được sử dụng vào những dự án lớn. Do đó, với các dự án nhỏ, mô hình MVC có thể gây cồng kềnh, tốn thời gian trong quá trình phát triển cũng như thời gian trung chuyển dữ liệu.

## 1.6 Sơ đồ UML của mô hình MVC



Hình 1. Sơ đồ UML của mô hình MVC

Hình 1. 2 mô tả mô hình MVC của chương trình Shape bao gồm 3 khối Model chứa các dữ liệu của Shape và Edge dưới dạng vector, khối Controller chứa các hàm thao tác với dữ liệu và khối View chứa giao diện với người dùng.

Do sơ đồ UML có kích thước lớn dẫn đến hình ảnh chưa được rõ nên nhóm sẽ đặt đường link đến sơ đồ UML:

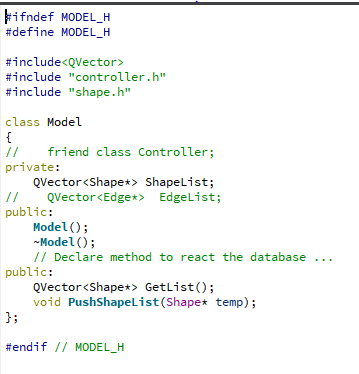
<https://drive.google.com/file/d/1lroABb79g604S7jzkUDns9SUbA_Q60IB/view?fbclid=IwAR3G25QRHumLGczlySaZBNPiIgCa0y2CLXDZkhR4EBxn2xE9cC2u1mMIoXI>

# CHƯƠNG 2. MÔ TẢ HOẠT ĐỘNG CỦA MVC

Cách thức hoạt động của mô hình MVC: Khi có một yêu cầu từ phía User lên View bằng cách click, edit,…. Bộ phận controller có nhiệm vụ nhận yêu cầu, xử lý yêu cầu đó. Và nếu cần truy xuất dữ liệu, nó sẽ gọi đến phần model – là bộ phận làm việc với Database….. Model nhận được thông tin từ Controller và thực thi các yêu cầu. Sau khi Model hoàn tất việc xử lý, toàn bộ kết quả được đẩy về phần View và hiện thị lại kết quả cho User.

## 2.1 Tạo Model cho Shape và Edge .

Model là nơi chứa các dữ liệu của lớp Shape và lớp Edge. Các thuộc tính của Shape và Edge được lưu trữ dưới dạng Container : Vector.



Hình 2. Model cho Shape và Edge

Hình 2. 1 mô tả khối Model cho chương trình Shape, tại đây Shape và Edge được lưu dưới dạng vector là ShapeList và EdgeList.

Các thuộc tính chung của Shape : Point O, Qstring color, Qstring thickness. Mỗi subclass ( Circle, Square, Rectangle, Over,…. ) có những thuộc tính riêng biệt của chúng được khai báo ở từng subclass và kế thừa những thuộc tính của lớp Shape. Các object này được lưu trữ ở QVector <Shape\*> Shapelist.

Các thuộc tính của Edge: Qstring EColor, Qstring EThickness, Shape\* shape1, Shape\* shape2. Các object của lớp Edge được lưu trữ ở Qvector <Edge\*> Edgelist.

## 2.2 Tạo Controller cho chương trình.

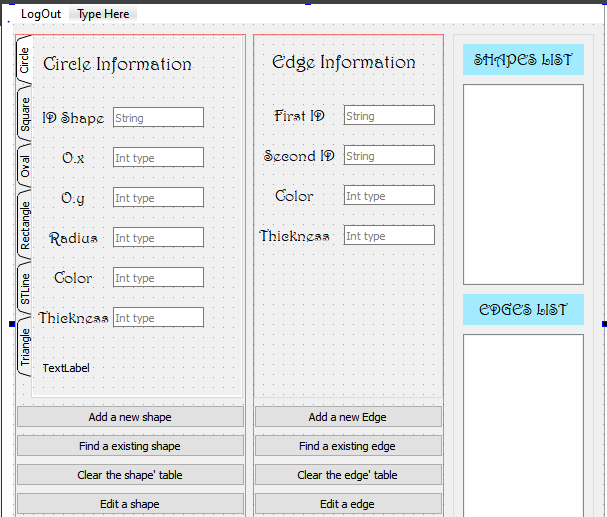


Hình 2. Class Controller

Hình 2. 2 mô tả khối Controller của chương trình Shape, Controller chứa Model và View. Hàm khởi tạo Controller( Model \* m) , void OnaddShapeButtonClicked( MainWindow\* view).

Khi nhận được một yêu cầu từ user lên View. Từ View sẽ gửi lại yêu cầu cho Controller để nó xử lý nếu cần truy xuất đến dữ liệu thì Controller sẽ gọi đến Model.

## 2.3 Tạo View cho chương trình.



Hình 2. View (Giao diện với người dùng)

Hình 2. 3 mô tả khối View trong mô hình MVC, và để hiển thị dữ liệu trong mô hình chúng em sử dụng QlistWidgetItem và sử dụng hộp thoại thông báo QmessageBox để phản hồi lại với user.

- Line Edit để nhập các thuộc tính của Shape và Edge đồng thời cũng sử dụng Line Edit để hiển thị dữ liệu.

- Textlabel.

- TabWidget để cho người dùng chọn các Shape cần thao tác.

- Các slots:

+ AddShape, FindShape, ClearShape, EditShape, RemoveShape.

+ AddEdge, FindEdge, ClearEdge, EditEdge.

-QListWidgetItem để hiển thị danh sách các Shape và Edge.

=> Đây là phần người dùng sẽ thao tác và nhận dữ liệu.

KẾT LUẬN

Trong bản báo cáo này nhóm 3 đã tìm hiểu về mô hình MVC Design Pattern, các khái niệm về M,V,C. Và cách tạo Model, Controller, View. Trong quá trình thực hiện không thể tránh được những thiếu sót rất mong thầy có thể góp ý để lần sau chúng em thực hiện tốt hơn. Qua đây chúng em cũng xin chân thành cảm ơn thầy Nguyễn Đức Minh đã hướng dẫn chúng em để hoàn thành sản phẩm này.

# TÀI LIỆU THAM KHẢO