Nhập môn công nghệ phần mềm

Cài đặt - Implementation

GV: ThS. Ngô Tiến Đức

Nội dung chính

- Luồng cài đặt
- Chuẩn bị kiểm thử
- Viết test case
- Cài đặt
- Tích hợp
- Tiến hành kiểm thử

Luồng cài đặt

Mục đích:

- Cài đặt hệ thống theo kết quả từ luồng thiết kế
- Kiểm thử phần mềm
  - Kiểm thử đơn vị ngay sau khi cài đặt một modul
  - Kiểm thử tích hợp và kiểm thử toàn bộ sản phẩm

Chuẩn bị kiểm thử (1)

Thực hiện:

- Viết test case trước khi cài đặt
- Test case dưới dạng hộp đen (black-box test):
  - Chỉ rõ đầu vào và đầu ra mong muốn

Chuẩn bị kiểm thử (2)

- Về nguyên tắc: Phải kiểm thử tất cả các trường hợp có thể
- Thực tế: Quá nhiều trường hợp phải kiểm thử

→ Sử dụng các kỹ thuật kiểm thử

Chuẩn bị kiểm thử (3)

Kỹ thuật phân vùng tương đương: Kiểm thử giá trị đại diện của từng vùng dữ liệu đầu vào (nhóm dữ liệu có cùng cách xử lý)

- Xác định các vùng tương đương hợp lệ và không hợp lệ
- Chọn giá trị đại diện cho từng vùng để kiểm thử

VD: Giá vé cho các đối tượng tùy theo độ tuổi

Chuẩn bị kiểm thử (4)

Kỹ thuật phân tích giá trị biên: Nếu một tham số đầu vào có một giới hạn biên **x** thì phải test ít nhất 4 trường hợp:

- Giá trị đầu vào ngay trên x
- Giá trị đầu vào ngay dưới x
- Giá trị đầu vào đúng bằng x
- Giá trị đầu vào bất kì cách xa x

VD: Phép chia số nguyên với điều kiện số bị chia phải khác 0

Chuẩn bị kiểm thử (5)

Kỹ thuật phân tích giá trị biên : Nếu một tham số đầu vào có 2 giới hạn biên $\mathbf{x_1}$ và $\mathbf{x_2}$ thì phải test ít nhất 7 trường hợp:

- Giá trị đầu vào đúng bằng x_1, ngay trên x_1, nhỏ hơn x_1
- Giá trị đầu vào đúng bằng x_2, ngay dưới x_2, lớn hơn x_2
- Giá trị đầu vào đúng bằng $(x_1 + x_2) / 2$

Chuẩn bị kiểm thử (6)

Kỹ thuật test chức năng thao tác CSDL: Với chức năng thêm một đối tượng vào CSDL thì phải test ít nhất 3 trường hợp:

- Thêm một đối tượng chưa có trong CSDL
- Thêm một đối tượng đã có trong CSDL
- Thêm liên tục 2 lần một đối tượng chưa có trong CSDL

Chuẩn bị kiểm thử (7)

Kỹ thuật test chức năng thao tác CSDL: Với chức năng sửa một đối tượng vào CSDL thì phải test ít nhất 3 trường hợp:

- Sửa một đối tượng chưa có trong CSDL
- Sửa một đối tượng đã có trong CSDL
- Sửa liên tục 2 lần một thuộc tính của đối tượng đã có trong CSDL

Chuẩn bị kiểm thử (8)

Kỹ thuật test chức năng thao tác CSDL: Với chức năng xóa một đối tượng vào CSDL thì phải test ít nhất 3 trường hợp:

- Xóa một đối tượng đã có trong CSDL
- Xóa một đối tượng chưa có trong CSDL
- Xóa liên tục 2 lần một đối tượng đã có trong CSDL

Chuẩn bị kiểm thử (9)

Kỹ thuật test chức năng thao tác CSDL: Với chức năng tìm kiếm một đối tượng vào CSDL thì phải test ít nhất 2 trường hợp:

- Tìm kiếm một đối tượng đã có trong CSDL
- Tìm kiếm một đối tượng chưa có trong CSDL

Chuẩn bị kiểm thử (10)

Kỹ thuật test chức năng thao tác CSDL:

Nếu CSDL có 3 bảng: tblClient lưu thông tin khách hàng, bảng tblProduct lưu thông tin sản phẩm, bảng tblBill lưu thông tin một khách hàng mỗi lần mua một số sản phẩm. Khi đó, chức năng thêm một hóa đơn vào trong CSDL phải kiểm thử các trường hợp:

- Thêm một hóa đơn đã có trong CSDL
- Thêm một hóa đơn chưa có trong CSDL

Chuẩn bị kiểm thử (11)

Trường hợp thêm hóa đơn chưa có trong CSDL lại bao gồm:

- Thêm một hóa đơn mà khách hàng và sản phẩm đã có trong CSDL
- Thêm một hóa đơn mà khách hàng chưa có trong CSDL, sản phẩm đã có trong CSDL
- Thêm một hóa đơn mà sản phẩm chưa có trong CSDL, khách hàng đã có trong CSDL
- Thêm một hóa đơn mà cả khách hàng và sản phẩm chưa có trong CSDL

Viết test case (1)

Các bước thực hiện: Với mỗi trường hợp

- Bước 1: Mô tả dữ liệu hiện có trong CSDL
- Bước 2: Mô tả các thao tác và kết quả mong đợi
- Bước 3: Mô tả kết quả mong đợi trong CSDL sau khi chạy test case

Viết test case (2)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn:

Các trường hợp cần kiểm thử

- Thêm một phòng chưa có trong CSDL
- Thêm một phòng đã có trong CSDL
- Thêm liên tục 2 lần một phòng chưa có trong CSDL

Viết test case (3)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn -

Trường hợp thêm một phòng chưa có trong CSDL

- Bước 1: Dữ liệu hiện có trong CSDL

id	name	type	displayPrice	description
1	101	single	10	old room
2	102	double	15	new room

Viết test case (4)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn -

Trường hợp thêm một phòng chưa có trong CSDL:

- Bước 2:

STT	Thao tác	Kết quả mong đợi
1	Quản lý chọn chức năng thêm phòng	Giao diện thêm phòng hiện ra với các ô nhập: ID phòng, tên phòng, kiểu phòng, giá hiển thị, mô tả, và 2 nút: nút thêm và nút hủy
2	Quản lý nhập : ID: 3 Tên: 103 Kiểu: Twin Giá hiển thị: 15 Và click nút "thêm" 1 lần	Thông báo thêm phòng thành công và nút OK
3	Quản lý click nút OK.	Trở lại giao diện thêm phòng

Viết test case (5)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn -

Trường hợp thêm một phòng chưa có trong CSDL:

- Bước 3: Kết quả mong đợi trong CSDL

id	name	type	displayPrice	description
1	101	single	10	old room
2	102	double	15	new room
3	103	twin	15	NULL

Viết test case (6)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn - Trường hợp thêm một phòng đã có trong CSDL

- Bước 1: Dữ liệu hiện có trong CSDL

id	name	type	displayPrice	description
1	101	single	10	old room
2	102	double	15	new room
3	103	twin	15	NULL

Viết test case (7)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn -

Trường hợp thêm một phòng đã có trong CSDL:

- Bước 2:

STT	Thao tác	Kết quả mong đợi
1	Quản lý chọn chức năng thêm phòng	Giao diện thêm phòng hiện ra với các ô nhập: ID phòng, tên phòng, kiểu phòng, giá hiển thị, mô tả, và 2 nút: nút thêm và nút hủy
2	Quản lý nhập : ID: 2 Tên: 104 Kiểu: Twin Giá hiển thị: 15 Và click nút "thêm" 1 lần	Thông báo ID phòng đã tồn tại, thêm phòng không thành công và nút OK.
3	Quản lý click nút OK.	Trở lại giao diện thêm phòng

Viết test case (8)

Ví dụ viết test case cho chức năng thêm một phòng cho khách sạn -

Trường hợp thêm một phòng đã có trong CSDL:

- Bước 3: Kết quả mong đợi trong CSDL

id	name	type	displayPrice	description
1	101	single	10	old room
2	102	double	15	new room
3	103	twin	15	NULL

Cài đặt (1)

Thứ tự:

- Cài đặt các lớp thực thể
- Cài đặt các lớp giao diện
- Cài đặt các lớp điều khiển

Cài đặt (2)

Đặt tên biến:

- Bắt đầu bằng chữ thường
- Có ý nghĩa và thống nhất để bảo trì được dễ hơn

Ví dụ: `frequency, freq, frqncy`; không nên đặt là `fr`

- Có thể sử dụng: `frequencyAverage, frequencyTotal` hoặc `averageFrequency, totalFrequency`
- Không nên đặt `FrequencyAverage` hoặc sử dụng đồng thời cả `frequencyAverage` và `totalFrequency`

Cài đặt (3)

Khi sử dụng các câu lệnh **if** lồng nhau:

- Thay vì viết:

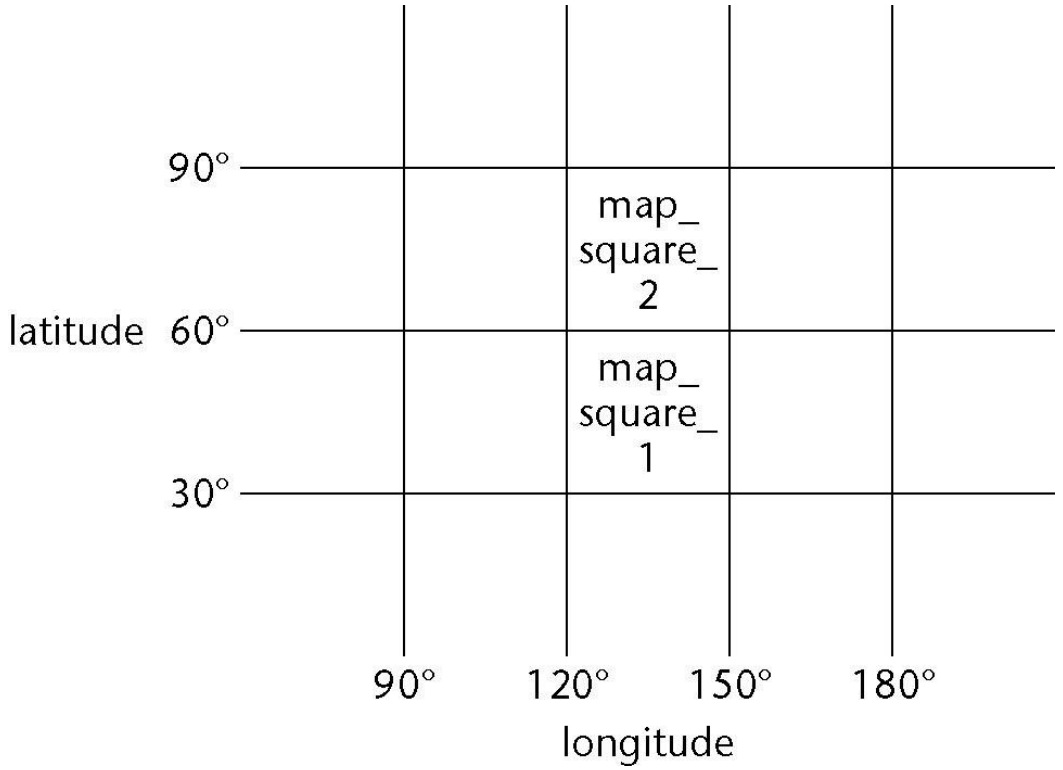
```
if <condition1>
    if <condition2>
    …
```

- Thì nên viết là:

```
if <condition1> && <condition2> …
```

Cài đặt (4)

Ví dụ: Xác định xem một điểm có tọa độ nằm trong vùng *map_square1* hoặc *map_square2* hoặc không



Cài đặt (4)

- Cách viết không chấp nhận được:

```
if (latitude > 30 && longitude > 120) {if (latitude <= 60 && longitude <= 150)
mapSquareNo = 1; else if (latitude <= 90 && longitude <= 150) mapSquareNo = 2
else print "Not on the map";} else print "Not on the map";
```

Cài đặt (5)

- Cách viết với format ổn nhưng cấu trúc chưa ổn:

```
if (latitude > 30 && longitude > 120)
{
  if (latitude <= 60 && longitude <= 150)
    mapSquareNo = 1;
  else
    if (latitude <= 90 && longitude <= 150)
      mapSquareNo = 2;
    else
      print "Not on the map";
}
else
  print "Not on the map";
```

Cài đặt (6)

- Cách viết chấp nhận được:

```
if (longitude > 120 && longitude <= 150 && latitude > 30 && latitude <= 60)
  mapSquareNo = 1;
else
  if (longitude > 120 && longitude <= 150 && latitude > 60 && latitude <= 90)
    mapSquareNo = 2;
  else
    print "Not on the map";
```

Cài đặt (7)

Chú thích code:

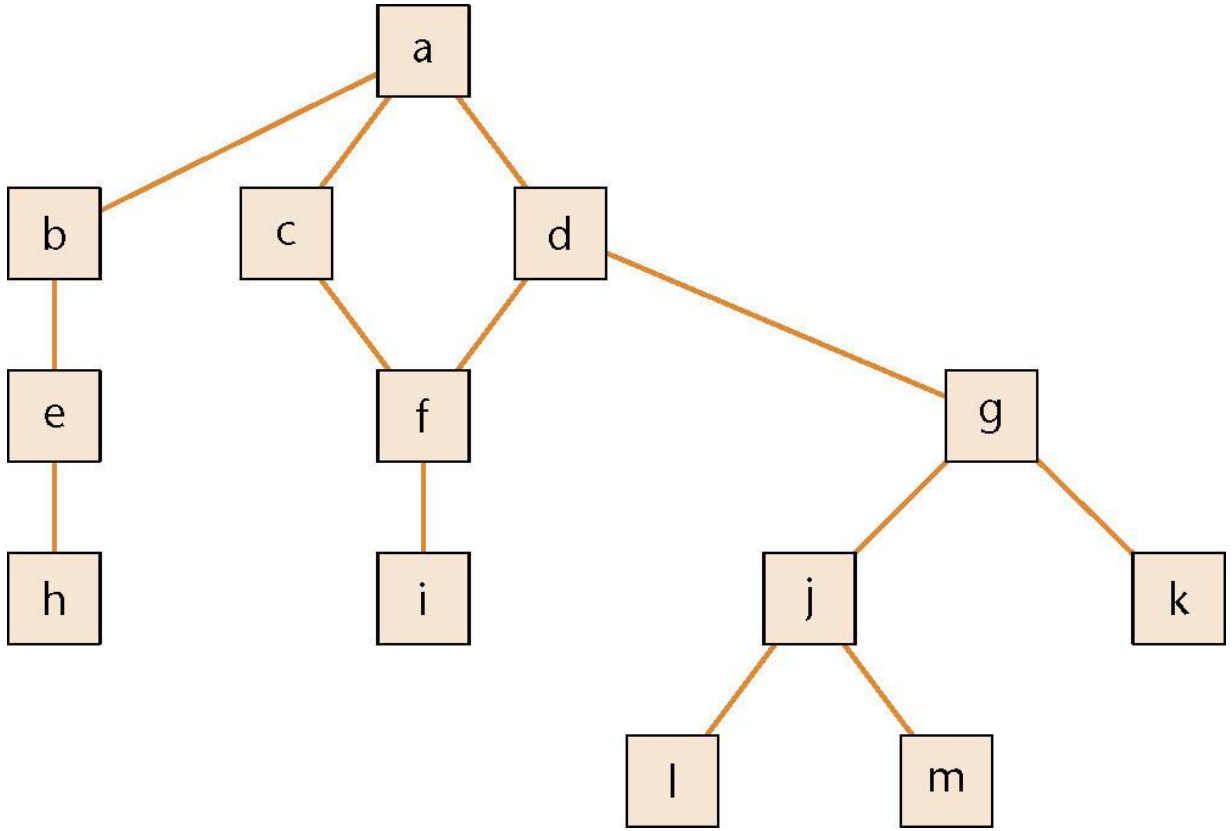
- Cần thiết cho việc bảo trì
- Nên chú thích ở đầu file, đầu mỗi lớp và đầu mỗi phương thức: Mô tả ngắn gọn chức năng và hoạt động, tên người code, những thay đổi qua các lần cập nhật, những việc cần làm,...
- Chú thích những đoạn code phức tạp và khó hiểu
- Không lạm dụng

Tích hợp (1)

Cách tiếp cận thông thường nhất:

Cài đặt sau đó tích hợp

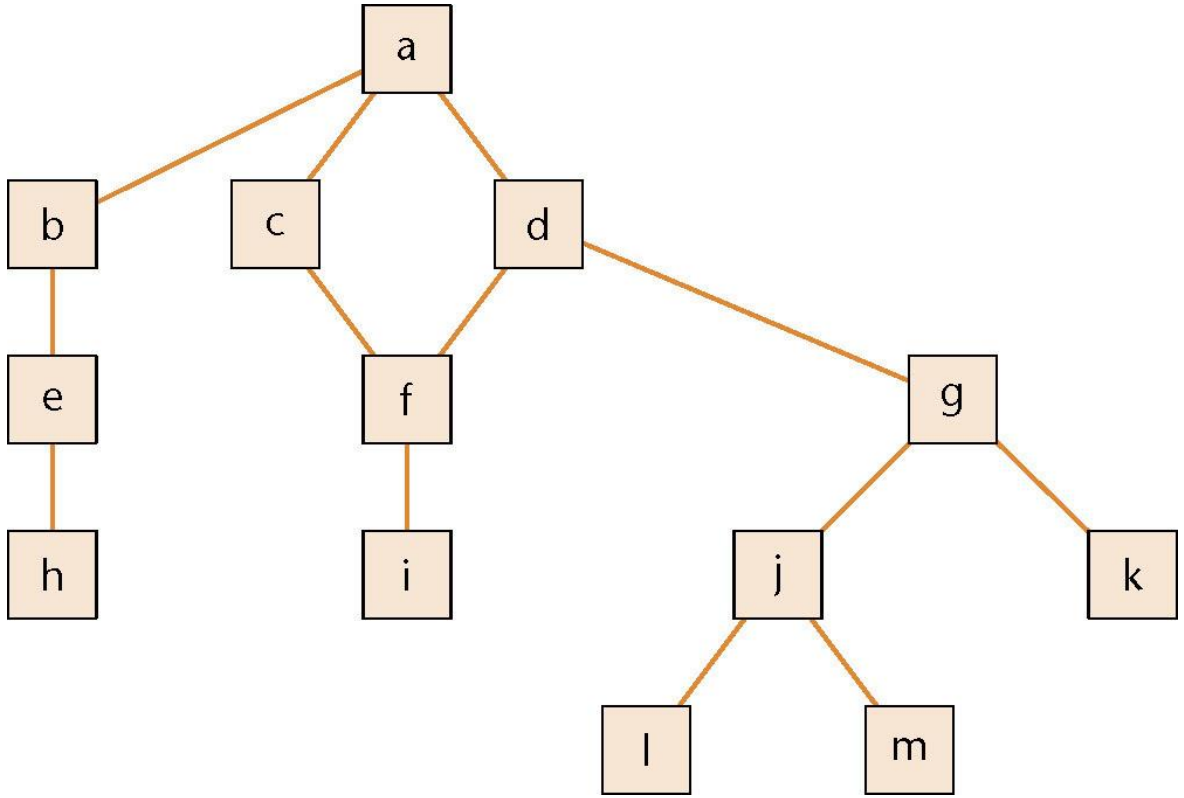
- Cài đặt và kiểm thử từng modul riêng biệt
- Tích hợp các modul rồi test toàn bộ sản phẩm



Tích hợp (2)

Để kiểm thử **a** thì phải coi **b, c, d** là các *stubs*, có thể là:

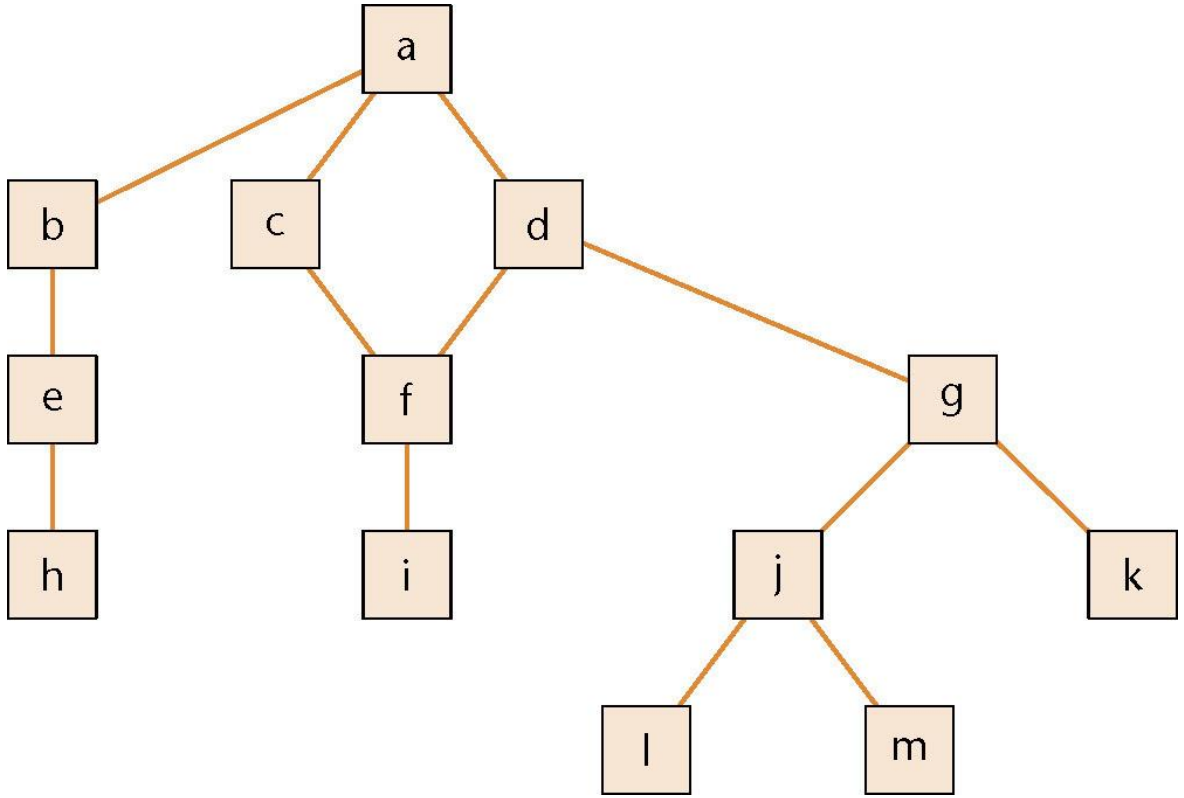
- Modul trống
- In ra một thông báo
- Trả về các giá trị đã được chuẩn bị trước từ test case



Tích hợp (3)

Để kiểm thử **h** thì cần phải thông qua một *driver* để gọi đến h:

- Một lần hoặc nhiều lần
- Nhiều lần với các kết quả trả về khác nhau



Tích hợp (4)

Các vấn đề:

- Phải viết các *stub* và *driver*
  - Chỉ để test
  - Bỏ đi sau khi đã hoàn thành unit test
- Khó khoanh vùng lỗi
  - Lỗi có thể nằm ở bất cứ đâu trong các modul

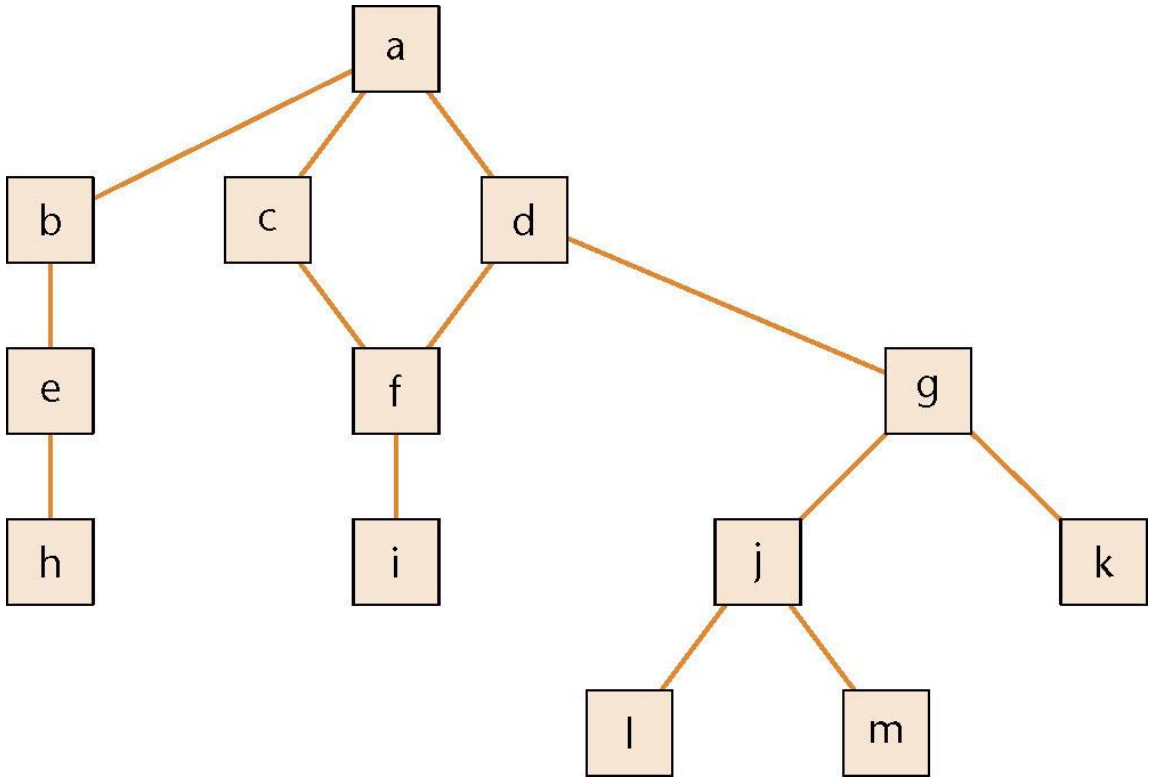
→ Cần kết hợp kiểm thử đơn vị và tích hợp

Tích hợp (5)

Tích hợp từ trên xuống (Top-down Integration):

Thứ tự: a, b, c, d, e, f, g, h, i, j, k, l, m

- **a** được cài đặt và kiểm thử với các stub **b, c, d**
- **b** được mở rộng thành modul và kiểm thử với stub **e**,...



Tích hợp (6)

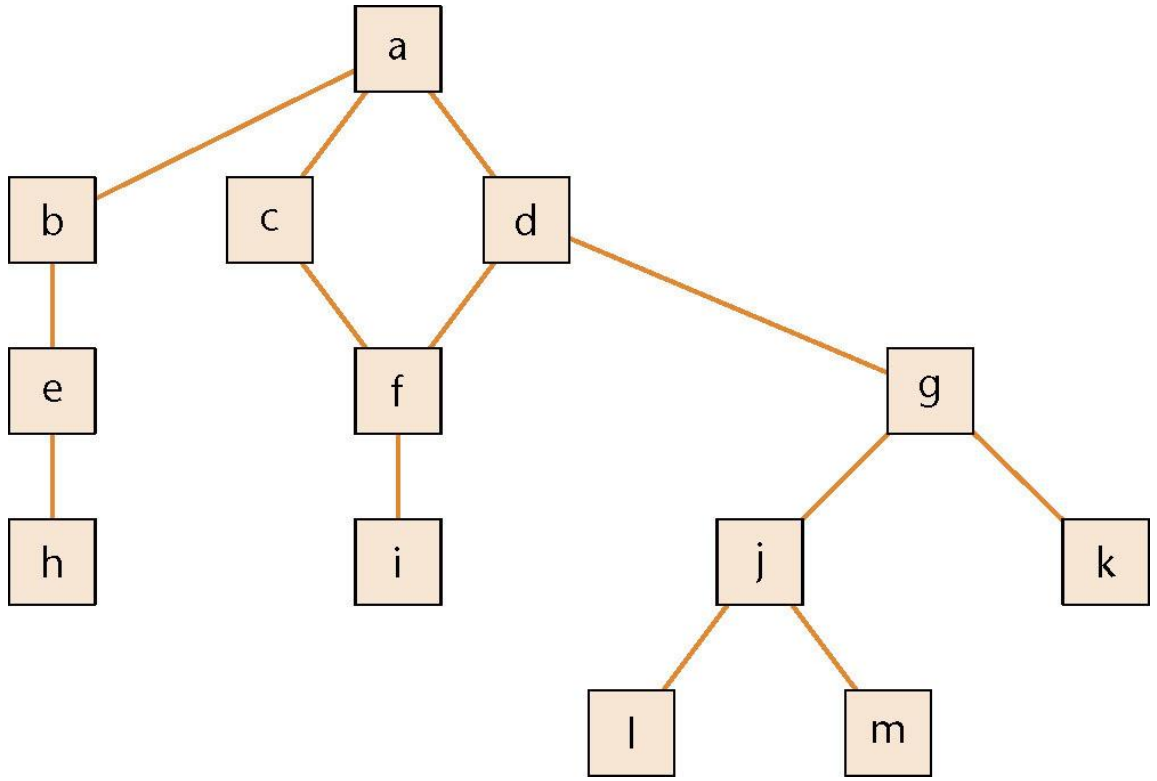
Tích hợp từ dưới lên (Bottom-up Integration):

Thứ tự: l, m, h, i, j, k, e, f, g, b, c, d, a

- Người thứ 1 code **h, e, b**
- Người thứ 2 code **i, f, c**
- Người thứ 3 code **l, m, j, k, g** sau đó

code **d** và tích hợp với người thứ 2

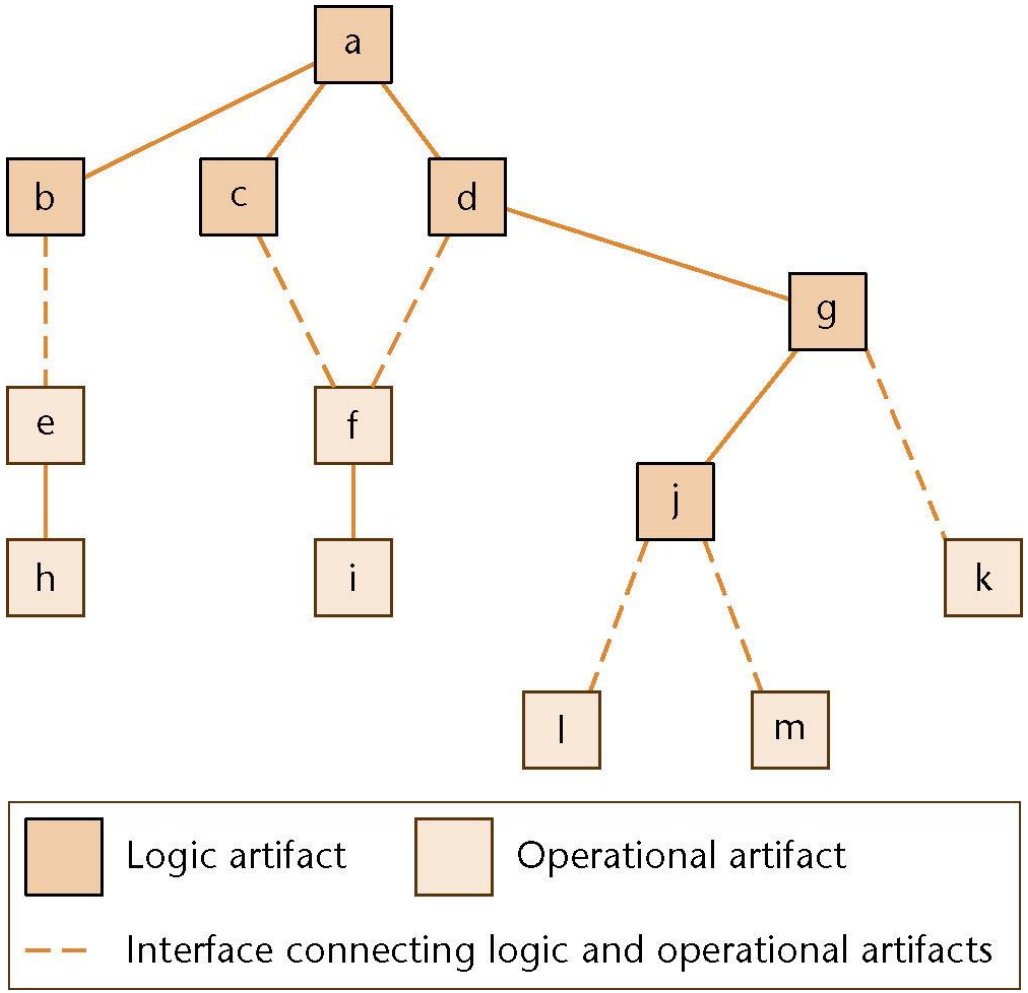
- Tích hợp **b, c, d** và code **a**



Tích hợp (7)

Tích hợp kiểu sandwich:

- a, c, c, d, g, j tích hợp từ trên xuống
- e, f, h, i, k, l , m tích hợp từ dưới lên



Tiến hành kiểm thử

- Với mỗi modul: Chạy các test case đã viết và lưu kết quả chạy test thành nhật ký
- Ví dụ:

Test case	Kết quả
1. Thêm phòng chưa có trong CSDL	Passed
2. Thêm phòng đã có trong CSDL	Passed
3. Thêm liên tục 2 lần 1 phòng đã có trong CSDL	Failed
…	…

Bài tập về nhà

- Bài tập cá nhân: Từ modul đã chọn, thực hiện các công việc:
  - Vẽ lại sơ đồ lớp và sơ đồ CSDL
  - Viết các test case
  - In bài làm và nộp trong buổi demo
  - Cài đặt theo kiến trúc đã thiết kế (bằng ngôn ngữ lập trình đã chọn) và demo chương trình

Tài liệu tham khảo

- Stephen R. Schach. *Object-Oriented and Classical Software Engineering*. 8th Edition, WCB/McGraw-Hill, 2010
- T. Đ. Quế, N. M. Hùng. *Bài giảng Nhập môn công nghệ phần mềm*. HVCNBCVT, 2020

Tham khảo

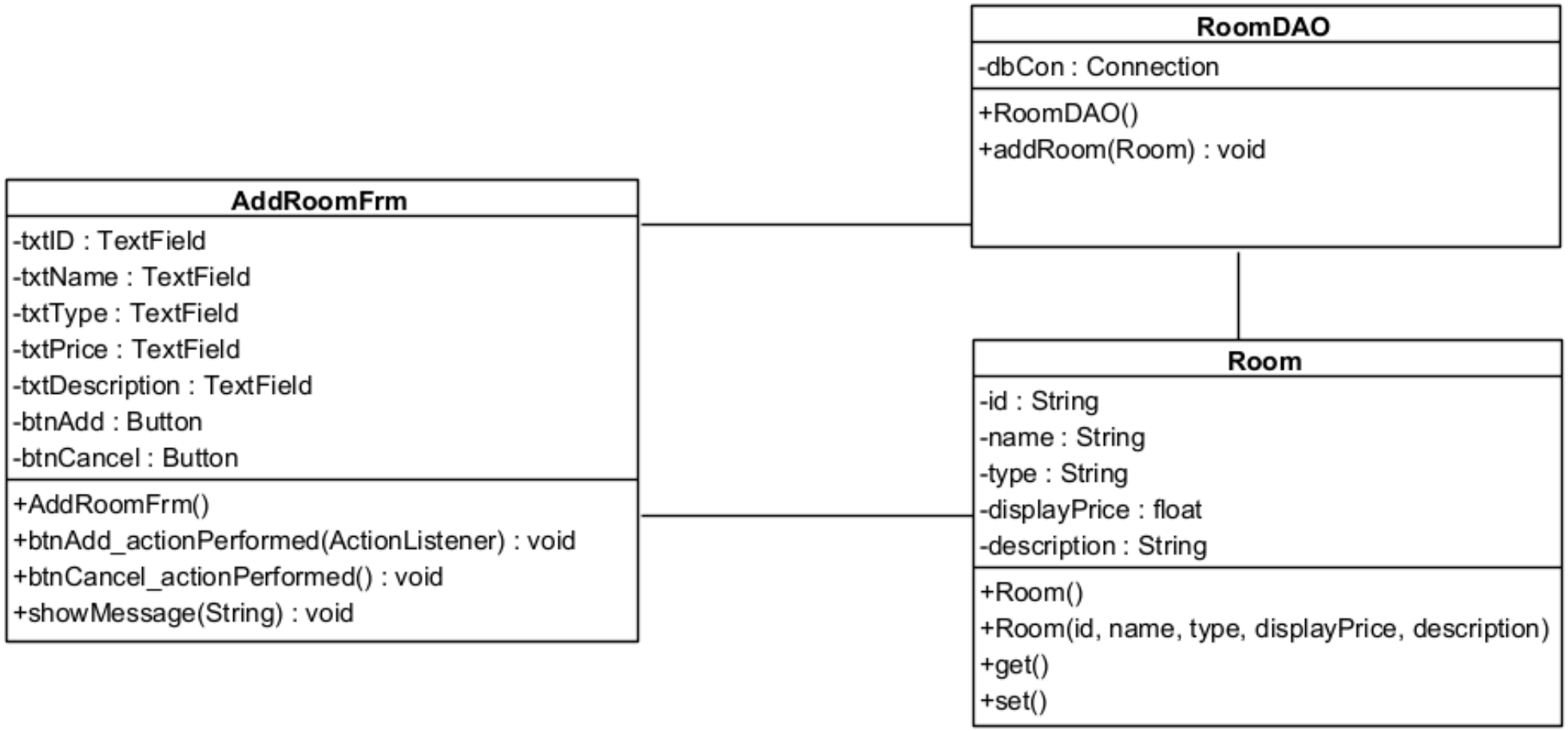
Cài đặt theo mô hình MVC

Modul thêm phòng khách sạn của người quản lý

Thông tin cơ bản

- Ngôn ngữ lập trình: Java
- CSDL: MySQL
- Trước khi chạy code:
  - Bật MySQL Server
  - Thêm driver JDBC của MySQL vào library của project
- Chương trình chưa bao gồm phần xử lý ngoại lệ

MVC thuần – Biểu đồ lớp



MVC thuần – Lớp Room

```
public class Room {
    private String id;
    private String name;
    private String type;
    private float displayPrice;
    private String description;

    public Room() {
        super();
    }

    public Room(String id, String name,…) {
        super();
        this.id = id;
        this.name = name;
        …
    }

    // getter(s) and setter(s) go here

} // eof
```

MVC thuần – Lớp AddRoomFrm

```
// import libraries
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
…

public class AddRoomFrm extends JFrame implements ActionListener {
    private JTextField txtID;
    private JTextField txtName;
    …

    public AddRoomFrm() {
           super("Room management - pure MVC");
           txtID = new JTextField(15);
           txtName = new JTextField(15);
           …
           JPanel content = new JPanel();
           content.setLayout(new GridLayout(6, 2));
           content.add(new JLabel("ID:"));
           content.add(txtID);
           content.add(new JLabel("Name:"));
           content.add(txtName);
           …
```

MVC thuần – Lớp AddRoomFrm

```
    content.add(txtDescription);
    content.add(btnSubmit);
    content.add(btnCancel);
    btnSubmit.addActionListener(this);
    btnCancel.addActionListener(this);
    this.setContentPane(content);
    this.pack();
    this.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            System.exit(0);
        }
    });
}

public void actionPerformed(ActionEvent e) {
    JButton btn = (JButton) e.getSource();
    if (btn.equals(btnSubmit)) {
        btnSubmit_actionperformed();
    } else if (btn.equals(btnCancel)) {
        btnCancel_actionperformed();
    }
}
```

MVC thuần – Lớp AddRoomFrm

```
 public void showMessage(String message) {
        JOptionPane.showMessageDialog(this, message);
    }

 public void btnSubmit_actionperformed() {
        Room room = new Room();
        room.setId(txtID.getText());
        room.setName(txtName.getText());
        room.setType(txtType.getText());
        room.setDisplayPrice(Float.parseFloat(txtDisplayPrice.getText()));
        room.setDescription(txtDescription.getText());
        RoomDAO rd = new RoomDAO();
        rd.addRoom(room);
        showMessage("Add room successfullly!");
    }
    public void btnCancel_actionperformed() {
        System.exit(0);
    }
}
// eof
```

MVC thuần – Lớp RoomDAO

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class RoomDAO {
    private Connection con;

    public RoomDAO() {
        String dbUrl = "jdbc:mysql://localhost:3306/hotel";
        // replace by your DB
        String dbClass = "com.mysql.cj.jdbc.Driver";
        try {
            Class.forName(dbClass);
            con = DriverManager.getConnection(dbUrl, "root", "1234");
             // replace by your MySQL account
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

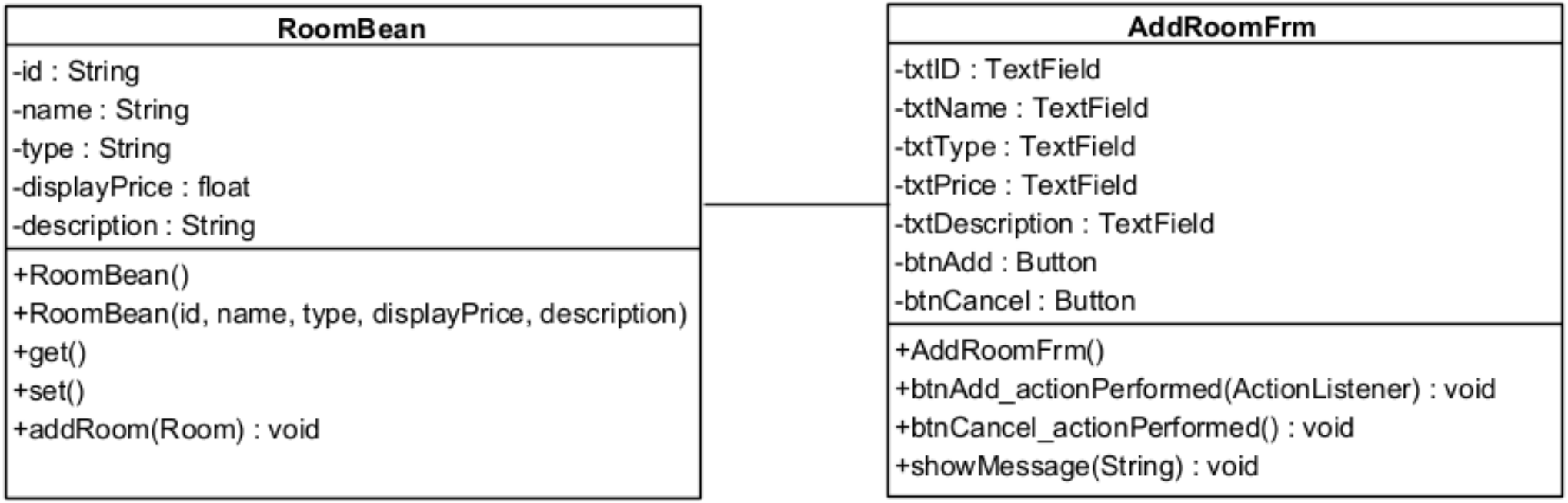
MVC thuần – Lớp RoomDAO

```
 public void addRoom(Room room) {
        String sql = "INSERT INTO tblRoom(id, name, type, displayPrice, description) VALUES(?,?,?,?,?)";
        try {
            PreparedStatement ps = con.prepareStatement(sql);
            ps.setString(1, room.getId());
            ps.setString(2, room.getName());
            ps.setString(3, room.getType());
            ps.setFloat(4, room.getDisplayPrice());
            ps.setString(5, room.getDescription());
            ps.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
// eof
```

MVC thuần – Lớp Main

```
public class Main {
    public static void main(String[] args) {
        AddRoomFrm arf = new AddRoomFrm();
        arf.setVisible(true);
    }
}
// eof
```

MVC Bean – Biểu đồ lớp



MVC Bean – Lớp RoomBean

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class RoomBean {
    private String id;
    private String name;
    …

    public RoomBean() {
        super();
    }

    public RoomBean(String id, String name,…) {
        super();
        this.id = id;
        this.name = name;
        …
   // getter(s) and setter(s) go here
```

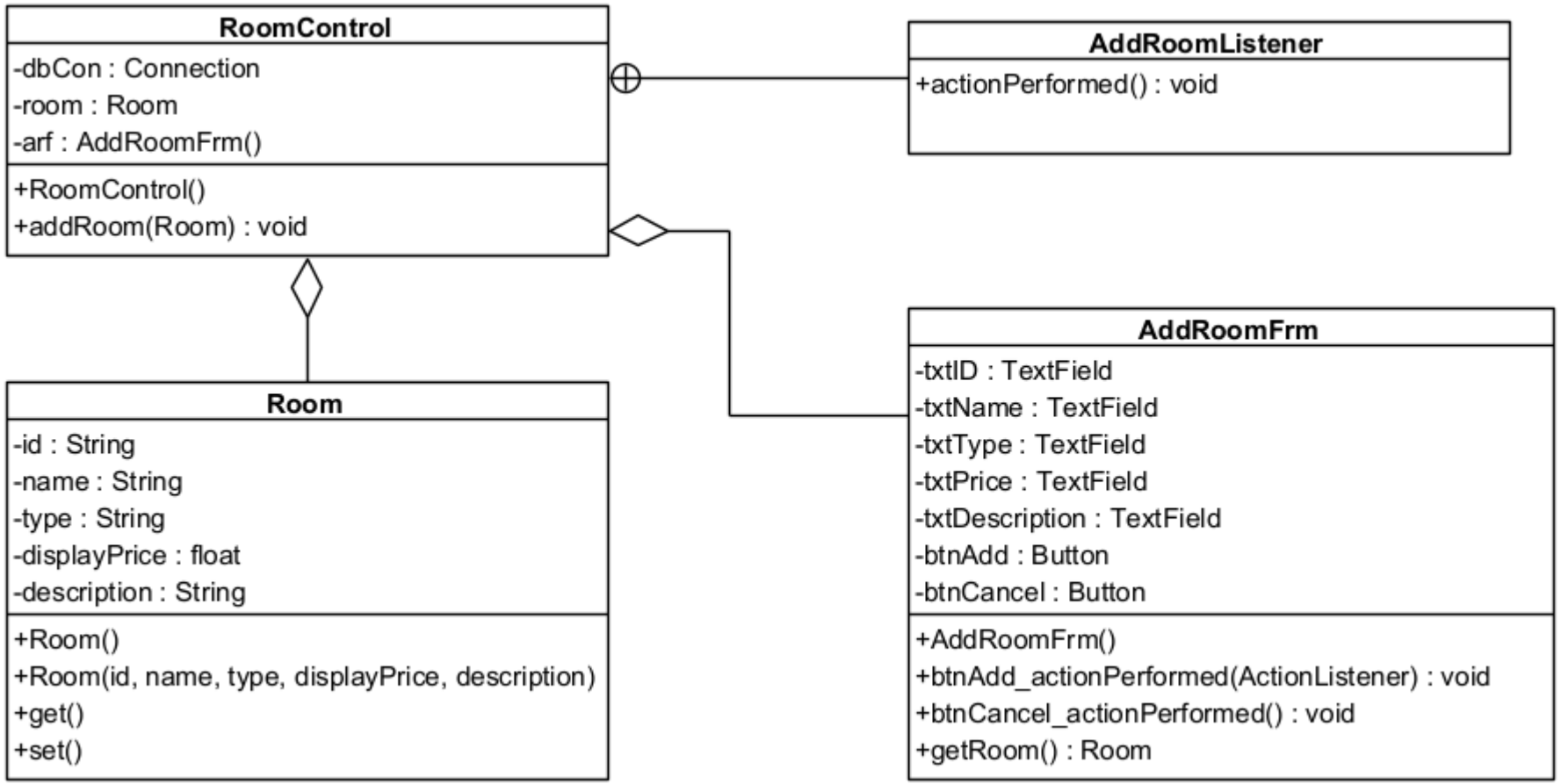
MVC Bean – Lớp RoomBean

```
public void addRoom() {
        String dbUrl = "jdbc:mysql://localhost:3306/hotel";
        String dbClass = "com.mysql.cj.jdbc.Driver";
        String sql = "INSERT INTO tblRoom(id, name, type, displayPrice, description) VALUES(?,?,?,?,?)";
        try {
            Class.forName(dbClass);
            Connection con = DriverManager.getConnection(dbUrl, "root", "1234");
            PreparedStatement ps = con.prepareStatement(sql);
            ps.setString(1, getId());
            ps.setString(2, getName());
            …
            ps.executeUpdate();
            con.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
// eof
```

MVC Bean – Lớp Main

```
public class Main {
    public static void main(String[] args) {
        AddRoomFrm arf = new AddRoomFrm();
        arf.setVisible(true);
    }
}
// eof
```

MVC cải tiến – Biểu đồ lớp



MVC cải tiến – Lớp Room

```
public class Room {
    private String id;
    private String name;
    private String type;
    private float displayPrice;
    private String description;

    public Room() {
        super();
    }

    public Room(String id, String name,…) {
        super();
        this.id = id;
        this.name = name;
        …
    }

    // getter(s) and setter(s) go here

} // eof
```

MVC cải tiến – Lớp AddRoomFrm

```
// import libraries
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
…

public class AddRoomFrm extends JFrame implements ActionListener {
    private JTextField txtID;
    private JTextField txtName;
    …

    public AddRoomFrm() {
           super("Room management - pure MVC");
           txtID = new JTextField(15);
           txtName = new JTextField(15);
           …
           JPanel content = new JPanel();
           content.setLayout(new GridLayout(6, 2));
           content.add(new JLabel("ID:"));
           content.add(txtID);
           content.add(new JLabel("Name:"));
           content.add(txtName);
           …
```

MVC cải tiến – Lớp AddRoomFrm

```
    content.add(txtDescription);
     content.add(btnSubmit);
     content.add(btnCancel);
     btnSubmit.addActionListener(this);
     btnCancel.addActionListener(this);
     this.setContentPane(content);
     this.pack();
     this.addWindowListener(new WindowAdapter() {
         public void windowClosing(WindowEvent e) {
             System.exit(0);
         }
     });
}

public void actionPerformed(ActionEvent e) {
     JButton btn = (JButton) e.getSource();
    if (btn.equals(btnCancel)) btnCancel_actionperformed();
}

public void btnCancel_actionperformed() {
     System.exit(0);
}
```

MVC cải tiến – Lớp AddRoomFrm

```
 public void showMessage(String message) {
        JOptionPane.showMessageDialog(this, message);
    }

    public Room getRoom() {
        Room room = new Room();
        room.setId(txtID.getText());
        room.setName(txtName.getText());
        room.setType(txtType.getText());
        room.setDisplayPrice(Float.parseFloat(txtDisplayPrice.getText()));
        room.setDescription(txtDescription.getText());
        return room;
    }

    public void addSubmitListener(ActionListener log) {
        btnSubmit.addActionListener(log);
    }
}
// eof
```

MVC cải tiến – Lớp RoomControl

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class RoomControl {
    private Connection con;
    private Room room;
    private AddRoomFrm arf;

 public RoomControl() {
        String dbUrl = "jdbc:mysql://localhost:3306/hotel";
        String dbClass = "com.mysql.cj.jdbc.Driver";
        try {
            Class.forName(dbClass);
            con = DriverManager.getConnection(dbUrl, "root", "1234");
        } catch (Exception e) {
            e.printStackTrace();
        }
        arf = new AddRoomFrm();
        arf.addSubmitListener(new AddRoomListener());
        arf.setVisible(true);
    }
```

MVC cải tiến – Lớp RoomControl

```
 public void addRoom(Room room) {
        String sql = "INSERT INTO tblRoom(id, name, type, displayPrice, description) VALUES(?,?,?,?,?)";
        try {
            PreparedStatement ps = con.prepareStatement(sql);
            ps.setString(1, room.getId());
            ps.setString(2, room.getName());
            …
            ps.executeUpdate();
        } catch (Exception e) { e.printStackTrace(); }
    }

    class AddRoomListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            try {
                room = arf.getRoom();
                addRoom(room);
                arf.showMessage("Add room successfullly!");
            } catch (Exception ex) { ex.printStackTrace(); }
        }
    }
}
// eof
```

MVC cải tiến – Lớp Main

```
public class Main {
    public static void main(String[] args) {
        RoomControl rc = new RoomControl();    }
}
// eof
```