

# LẬP TRÌNH THEO KIẾN TRÚC 3-LAYER

## MỤC LỤC

1	Yêu cầu.....	2
2	Kiến trúc triển khai ứng dụng mức vật lý .....	2
3	Kiến trúc thiết kế ứng dụng mức logic.....	2
3.1	Mức nhìn vật lý và logic của ứng dụng.....	2
3.2	Sự phụ thuộc giữa các Layer.....	3
3.3	Cụ thể cài đặt.....	3
3.3.1	Xây dựng DTO (đối tượng truyền tải dữ liệu) .....	4
3.3.2	Xây dựng DAO (đối tượng truy cập dữ liệu) cho Data Access Layer .....	5
3.3.3	Xây dựng Business Object (đối tượng xử lý nghiệp vụ) cho Business Layer	6
3.3.4	Xây dựng các thể hiện trên GUI Layer .....	6
4	Tổng kết.....	8
4.1	Vai trò của các Layer .....	8
4.2	Lưu ý .....	8

## 1 Yêu cầu

Cho một phân số được lưu trữ trong tập tin *CSDLPhanSo.xml*. Tập tin này chỉ chứa những *phân số có giá trị không lớn hơn 5* (đây chính là yêu cầu nghiệp vụ). Xây dựng chương trình cho phép *đọc phân số* từ file đã cho, *sửa giá trị phân số* và *cập nhật* lại vào file này.

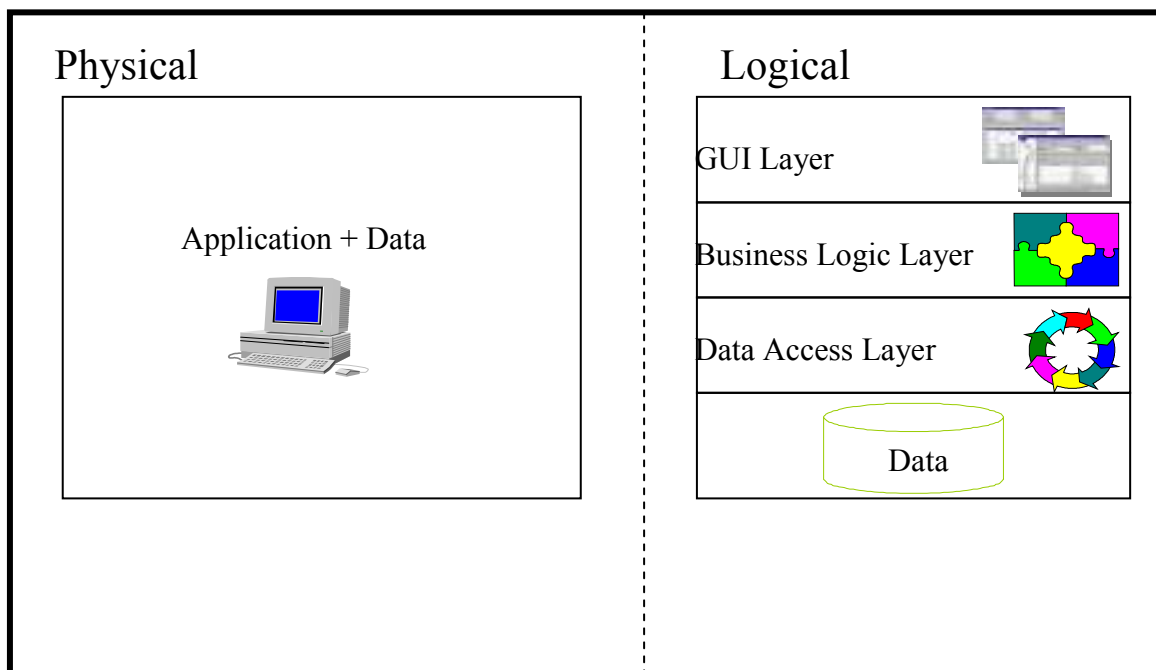
## 2 Kiến trúc triển khai ứng dụng mức vật lý

Ta chọn kiến trúc 1-tier, ứng dụng và dữ liệu sẽ được đặt trên máy đơn.

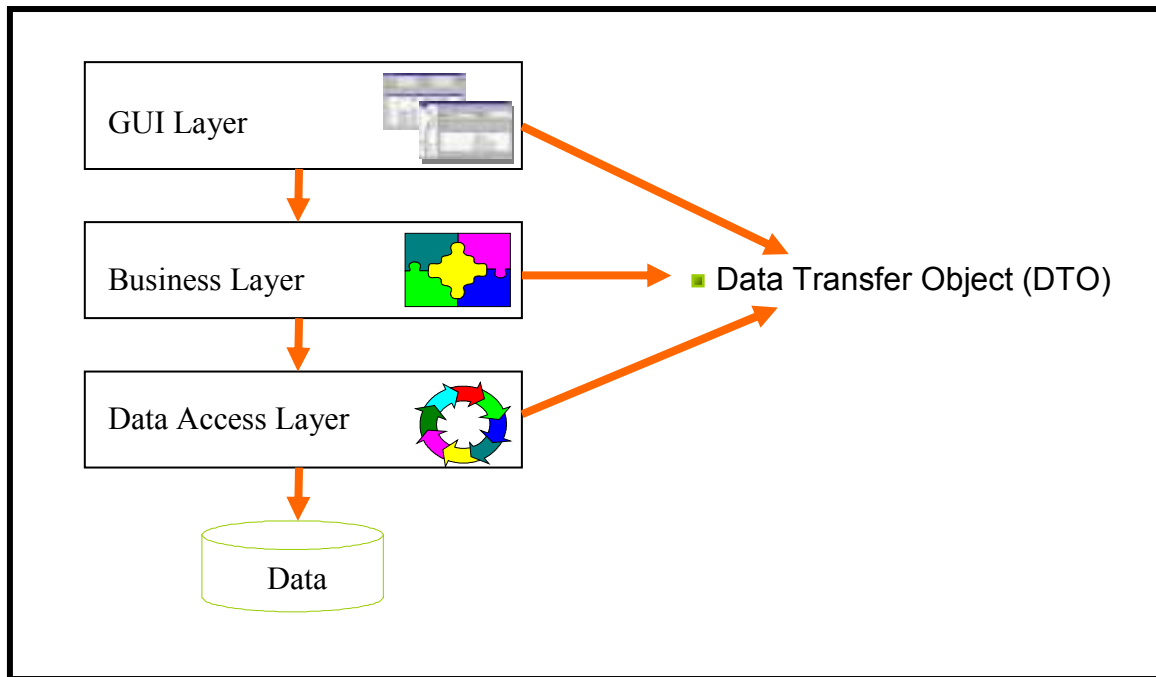
## 3 Kiến trúc thiết kế ứng dụng mức logic

Ta chọn kiến trúc 3-layer để xây dựng ứng dụng.

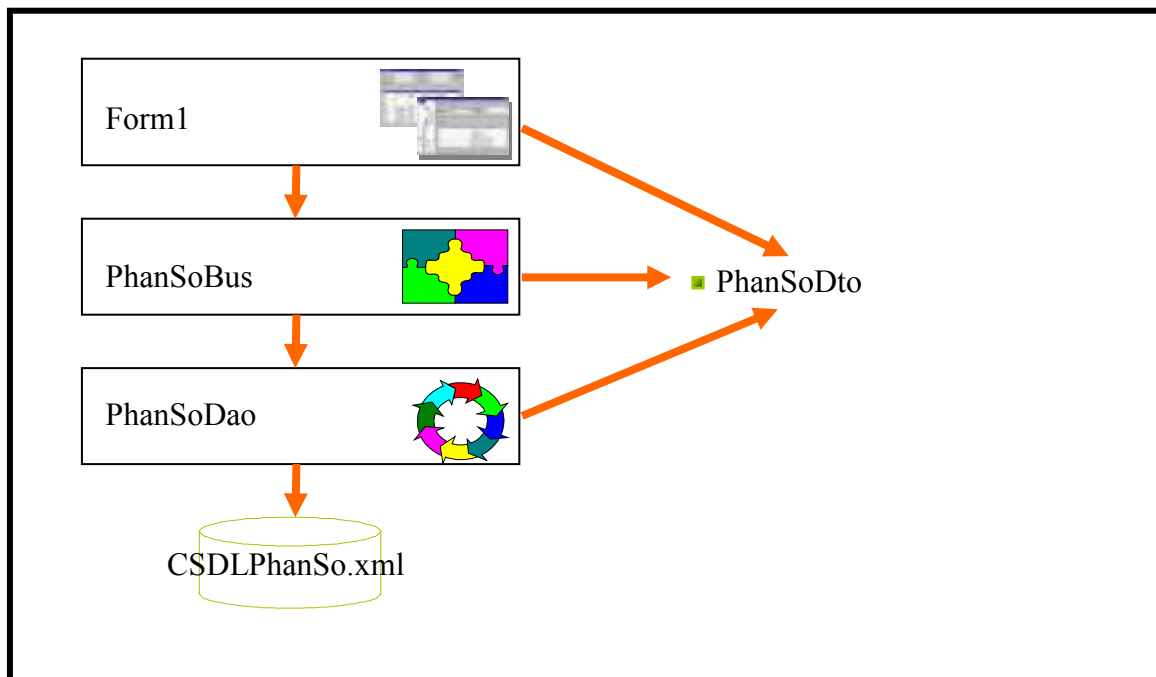
### 3.1 Mức nhìn vật lý và logic của ứng dụng



### 3.2 Sự phụ thuộc giữa các Layer



### 3.3 Cụ thể cài đặt



### 3.3.1 Xây dựng DTO (đối tượng truyền tải dữ liệu)

Ta sẽ xây dựng lớp PhanSoDto để truyền tải dữ liệu qua lại giữa các layer.

\*Lưu ý: Các DTO vẫn có xử lý riêng của nó. Lớp PhanSoDto có xử lý riêng là hàm **TimMaxUocChung** và hàm **RutGon** để tìm ước số chung và rút gọn phân số trước khi ghi vào file.

```
//PhanSoDto.cs
public class PhanSoDto
{
    private int tu;
    private int mau;

    public int Tu
    {
        get{return tu;}
        set{tu = value;}
    }

    public int Mau
    {
        get{return mau;}
        set{mau = value;}
    }

    public PhanSoDto()
    {
        Tu = 0;
        Mau = 1;
    }

    public PhanSoDto(int aTu, int aMau)
    {
        Tu = aTu;
        Mau = aMau;
    }

    public PhanSoDto RutGon()
    {
        PhanSoDto psDto = new PhanSoDto();
        int maxUC = TimMaxUocChung(Tu, Mau);
        psDto.Tu = Tu/maxUC;
        psDto.Mau = Mau/maxUC;
        return psDto;
    }

    private int TimMaxUocChung(int a, int b)
    {
        while(a!=b)
        {
            if(a>b)
                a -= b;
            else
                b -= a;
        }
    }
}
```

```
        return a;
    }
}
```

### 3.3.2 Xây dựng DAO (đối tượng truy cập dữ liệu) cho Data Access Layer

Ta sẽ xây dựng lớp PhanSoDao để truy cập dữ liệu của phân số. Lớp này sẽ có hàm **DocPhanSo** và **GhiPhanSo** để đọc phân số từ file và ghi giá trị của phân số vào file

\*Lưu ý: Các DAO vẫn có thể có xử lý riêng của nó. Dữ liệu được lấy ra và ghi vào file đều thông qua đối tượng truyền tải dữ liệu PhanSoDto.

```
//PhanSoDao.cs
public class PhanSoDao
{
    public PhanSoDao()
    {

    }

    public PhanSoDto DocPhanSo(string strFileName)
    {
        PhanSoDto phanso = null;
        XmlDocument doc = new XmlDocument();
        doc.Load(strFileName);
        if (doc != null)
        {
            phanso = new PhanSoDto();
            XmlElement ele = doc.DocumentElement;
            phanso.Tu =
            int.Parse(ele.SelectSingleNode("Tu_so").InnerText);
            phanso.Mau =
            int.Parse(ele.SelectSingleNode("Mau_so").InnerText);
        }

        return phanso;
    }

    public void GhiPhanSo(PhanSoDto psDao, string strFileName)
    {
        XmlDocument doc = new XmlDocument();
        XmlElement root = doc.CreateElement("PHANSO");
        doc.AppendChild(root);
        XmlElement ele_Tuso =
            root.OwnerDocument.CreateElement("Tu_so");
        ele_Tuso.InnerText = psDao.Tu.ToString();
        root.AppendChild(ele_Tuso);
        XmlElement ele_Mauso =
            root.OwnerDocument.CreateElement("Mau_so");
        ele_Mauso.InnerText = psDao.Mau.ToString();
        root.AppendChild(ele_Mauso);

        doc.Save(strFileName);
    }
}
```

### 3.3.3 Xây dựng Business Object (đối tượng xử lý nghiệp vụ) cho Business Layer

Ta sẽ xây dựng lớp PhanSoBus để đáp ứng nhu cầu truy cập dữ liệu từ GUI Layer, kiểm tra các yêu cầu nghiệp vụ (trong bài toán này là phân số không lớn hơn 5) trước khi cập nhật dữ liệu. Cụ thể lớp này sẽ có 2 hàm:

- **DocPhanSo**: dùng để đọc phân số từ file.
- **GhiPhanSo**: dùng để ghi giá trị phân số vào file. Trong hàm này có xử lý nghiệp vụ yêu cầu phân số được ghi vào phải có giá trị > 5. Nếu yêu cầu nghiệp vụ không được đảm bảo lớp sẽ “quăng ra” một Exception và không thực hiện tiếp công việc.

```
//PhanSoBus.cs
public class PhanSoBus
{
    public PhanSoBus()
    {
        //
        // TODO: Add constructor logic here
        //
    }

    public PhanSoDto DocPhanSo(string strFileName)
    {
        PhanSoDao psDao = new PhanSoDao();
        PhanSoDto psDto = psDao.DocPhanSo(strFileName);
        return psDto;
    }

    public void GhiPhanSo(PhanSoDto psDto, string strFileName)
    {
        int temp = psDto.Tu/psDto.Mau;

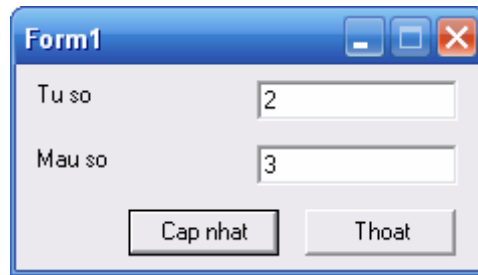
        if (temp > 5)
        {
            throw new Exception("Chương trình không cho phép nhập
phan so lon hon 5");
        }

        PhanSoDao psDao = new PhanSoDao();
        psDao.GhiPhanSo(psDto.RutGon(), strFileName);
    }
}
```

### 3.3.4 Xây dựng các thể hiện trên GUI Layer

Tạo Form1 có giao diện như sau:

## Lập trình theo kiến trúc 3-Layer



\*Chú ý: Form1 cũng sẽ có những xử lý riêng của nó. Ta thêm vào Form1 những xử lý sau:

- Hàm quản lý sự kiện KeyPress của txtTuSo, và txtMauSo để đảm bảo người dùng chỉ nhập vào được các ký số.
- Hàm quản lý sự kiện Validated của txtMauSo để đảm bảo người dùng không nhập được phân số có mẫu số là 0.

```
//Form1.cs
private string strFileName = @"C:\CSDLPhanSo.xml";

private void txtTuSo_KeyPress(object sender,
System.Windows.Forms.KeyPressEventArgs e)
{
    if (e.KeyChar < '0' || e.KeyChar > '9')
    {
        e.Handled = true;
    }
}

private void txtMauSo_KeyPress(object sender,
System.Windows.Forms.KeyPressEventArgs e)
{
    if (e.KeyChar < '0' || e.KeyChar > '9')
    {
        e.Handled = true;
    }
}

private void txtMauSo_Validated(object sender, System.EventArgs e)
{
    int temp = int.Parse(txtMauSo.Text);
    if (temp == 0)
    {
        MessageBox.Show("Mau khong duoc bang 0");
        txtMauSo.Focus();
    }
}

private void Form1_Load(object sender, System.EventArgs e)
{
    PhanSoBus psBus = new PhanSoBus();
    PhanSoDto psDto = psBus.DocPhanSo(strFileName);
    txtTuSo.Text = psDto.Tu.ToString();
    txtMauSo.Text = psDto.Mau.ToString();
}
```

```
private void btnCapNhat_Click(object sender, System.EventArgs e)
{
    try
    {
        PhanSoDto psDto = new PhanSoDto();
        psDto.Tu = int.Parse(txtTuSo.Text);
        psDto.Mau = int.Parse(txtMauSo.Text);
        PhanSoBus psBus = new PhanSoBus();
        psBus.GhiPhanSo(psDto, strFileName);
        MessageBox.Show("Cap nhat thanh cong", "Thong tin",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch(Exception ex)
    {
        MessageBox.Show(ex.Message, "Loi", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void btnThoat_Click(object sender, System.EventArgs e)
{
    this.Close();
}
```

## 4 Tổng kết

### 4.1 Vai trò của các Layer

- GUI (Presentation) Layer: Nhập liệu và trình bày dữ liệu, có thể bao gồm bước kiểm tra dữ liệu trước khi gọi Business Logic Layer.
- Business Logic Layer: Đáp ứng yêu cầu thao tác dữ liệu của GUI Layer, kiểm tra các yêu cầu nghiệp vụ trước khi cập nhật dữ liệu, quản lý các Transaction, quản lý các concurrent access.
- Data Access Layer: Kết nối CSDL, tìm kiếm, thêm, xóa, sửa, ... trên CSDL.

### 4.2 Lưu ý

- Các DTO được “quăng qua quăng lại” giữa các Layer
- Cần phân biệt vai trò Business Logic Layer và khái niệm “xử lý”.
- Mỗi layer vẫn có xử lý riêng, đặc trưng của layer đó.
- Đôi khi việc quyết định 1 xử lý nằm ở layer nào chỉ mang tính chất tương đối.
- Người ta thường xây dựng mỗi layer là một project. Các layer đều tham chiếu đến dll của DTO, và GUI Layer tham chiếu đến dll của BusinessLayer, BusinessLayer tham chiếu đến dll của Data Access Layer (xem sơ đồ sự phụ thuộc giữa các Layer).