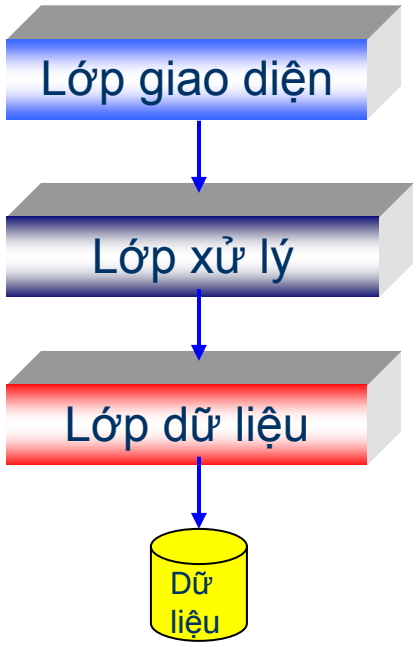
LẬP TRÌNH THEO KIẾN TRÚC 3 LỚP
(3-tiers architechture)

1. Xây dựng phần mềm theo kiến trúc 3 lớp:

Trước đây, đối với các phần mềm có sử dụng liên quan đến dữ liệu, thường khi làm người lập trình thường tích hợp việc giao tiếp với người sử dụng , xử lý rồi ghi xuống dữ liệu trên cùng một form (đây là mô hình một lớp). Nhưng trong kiến trúc 3 lớp (mô hình 3 lớp), phải có việc phân biệt rạch ròi giữa các lớp này. Mô hình 3 lớp có thể được mô tả như sau :



- ***Lớp thứ nhất :*** **Lớp giao diện** (giao tiếp với người sử dụng) : chỉ thuần xử lý việc giao tiếp với người sử dụng, nhập xuất, … mà không thực hiện việc tính toán, kiểm tra, xử lý, hay các thao tác liên quan đến cơ sở dữ liệu.

- ***Lớp thứ hai :*** **Lớp xử lý** : Lớp này chuyên thực hiện các xử lý , kiểm tra các ràng buộc, các qui tắc ứng xử của phần mềm , các chức năng cốt yếu, … Việc thực hiện này độc lập với cách thiết kế cũng như cài đặt giao diện. Thông tin cho lớp này thực hiện các xử lý của mình được lấy từ lớp giao diện.

- ***Lớp thứ ba :*** **Lớp dữ liệu** : Lớp này chuyên thực hiện các công việc liên quan đến dữ liệu. Dữ liệu có thể lấy từ cơ sở dữ liệu (Access, SQL Server …) hoặc tập tin (text, binary, XML …). Đối với cơ sở dữ liệu, lớp này thực hiện kết nối trực tiếp với cơ sở dữ liệu và thực hiện tất cả các thao tác liên

quan đến cơ sở dữ liệu mà phần mềm cần thiết. Đối với tập tin, lớp này thực hiện việc đọc, ghi tập tin theo yêu cầu của phần mềm. Việc thực hiện này do lớp xử lý gọi.

⇨ Rõ ràng, với mô hình này, các công việc của từng lớp là độc lập với nhau. Việc thay đổi ở một lớp không làm thay đổi các lớp còn lại, thuận tiện hơn cho quá trình phát triển và bảo trì phần mềm.

Lưu ý: lớp ở đây là tier chứ không phải là class

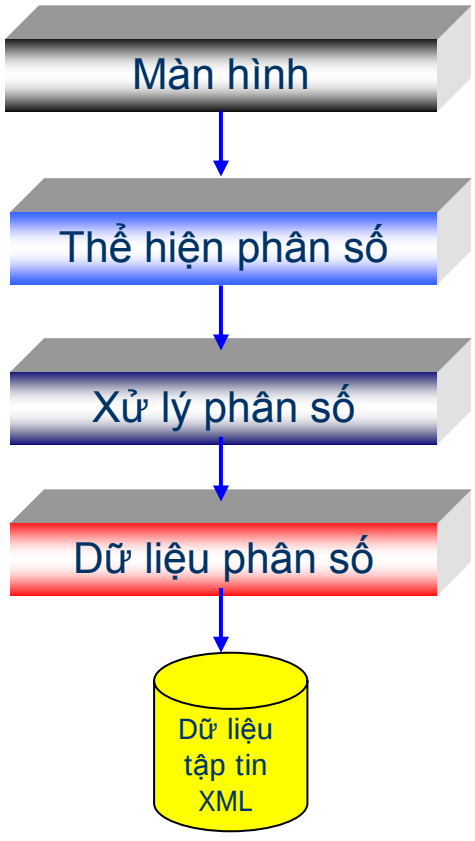
2. Ứng dụng minh họa:

Bài toán

Xây dựng chương trình tính tổng 2 phân số theo kiến trúc 3 lớp. Theo đó dữ liệu của phân số được đọc lên từ tập tin XML, kết quả sau khi được tính sẽ được ghi xuống tập tin XML

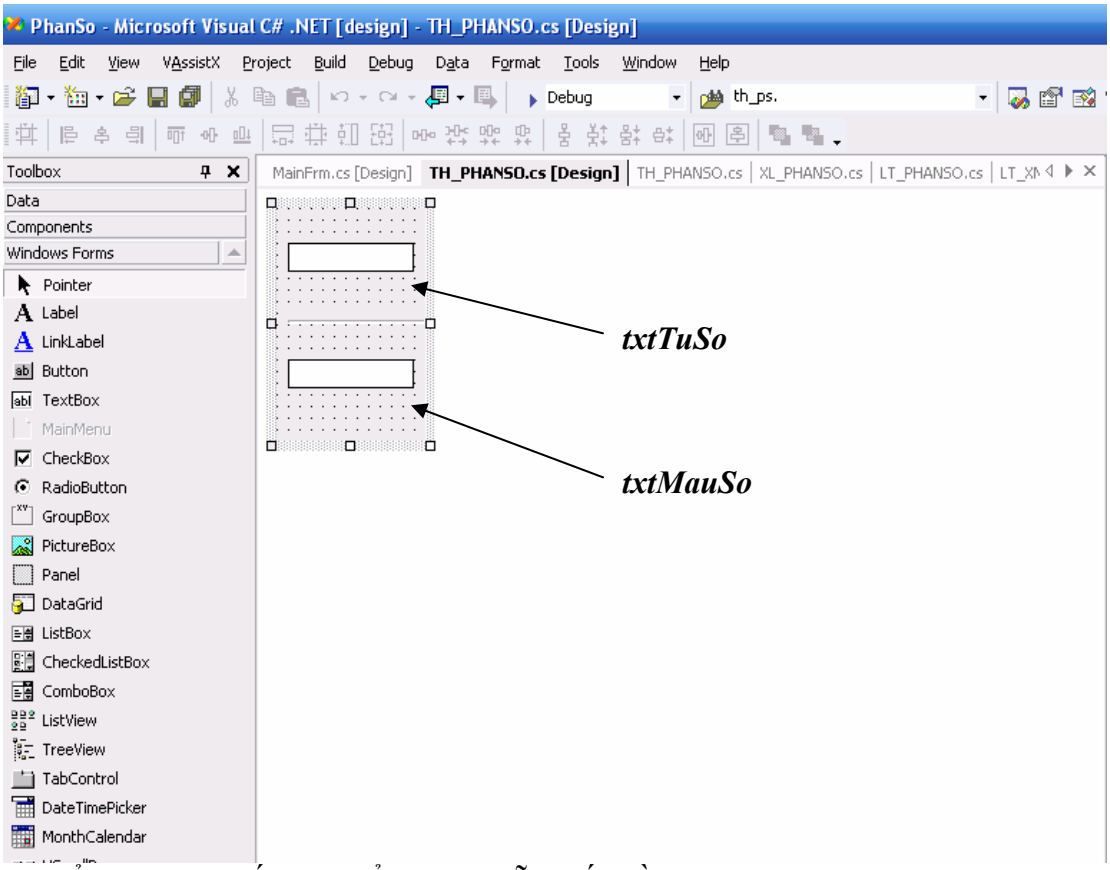
Cách làm thông thường là mọi việc đều được đẩy vào trong 1 form và xử lý trực tiếp trong form đó. Tuy nhiên, khi có sự thay đổi xảy ra về giao diện, xử lý, hay dữ liệu thì việc chỉnh sửa khá khó khăn. Do vậy, việc xây dựng theo kiến trúc 3 lớp sẽ khắc phục nhược điểm này.

Kiến trúc của chương trình như sau



Xây dựng lớp thể hiện phân số (TH_PHANSO)

Sử dụng User Control để cài đặt cho TH_PHANSO. Thêm User Control vào project bằng cách chọn **Project > Add User Control.** Đặt tên User Control đó. Ta có **TH_PHANSO.cs**



Do thể hiện tử số và thể hiện mẫu số đều là TextBox do đó trong lớp TH_PHANSO cần thiết lập các properties là tuso và mauso có kiểu int.

```
public int tuso{
    set{
        this.txtTuSo.Text = value.ToString();
    }
    get{
        return int.Parse(this.txtTuSo.Text);
    }
}
public int mauso
{
    set
    {
        this.txtMauSo.Text = value.ToString();
    }
    get
    {
        return int.Parse(this.txtMauSo.Text);
    }
}
```

Lớp lưu trữ phân số (LT_PHANSO)

Tập tin XML lưu trữ có định dạng như sau

```
<?xml version ="1.0" encoding = "utf-8"?>
<PHANSO>
    <Tu_so>5</Tu_so>
    <Mau_so>3</Mau_so>
</PHANSO>
```

Để thực hiện việc đọc và ghi dữ liệu XML ta sử dụng DOM.

Khai báo tuso và mauso để thực hiện việc lưu trữ

```
public int tuso;
public int mauso;
```

Thực hiện cài đặt *hàm khởi tạo mặc định* với tham số truyền vào là đường dẫn file XML

```
public LT_PHANSO(string strFilename)
{
    //
    // TODO: Add constructor logic here
    //
    XmlDocument doc = LT_XML.DocTaiLieu(strFilename);
    if(doc == null)
    {
        tuso = 0;
        mauso = 0;
        return;
    }

    XmlElement ele = doc.DocumentElement;
    tuso = int.Parse(ele.SelectSingleNode("Tu_so").InnerText);
    mauso = int.Parse(ele.SelectSingleNode("Mau_so").InnerText);
}
```

Thực hiện cài đặt *hàm ghi phân số* với tham số truyền vào là đường dẫn file XML

```
public void GhiPhanSo(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 = this.tuso.ToString();
     root.AppendChild(ele_Tuso);
     XmlElement ele_Mauso =
root.OwnerDocument.CreateElement("Mau_so");
     ele_Mauso.InnerText = this.mauso.ToString();
     root.AppendChild(ele_Mauso);

     LT_XML.GhiTaiLieu(strFilename,doc);
}
```

Lớp lưu trữ XML (LT_XML)

Việc load và save XmlDocument được tách ra thành một lớp riêng là lớp LT_XML

```
public static XmlDocument DocTaiLieu(string strFilename)
{
     XmlDocument kq = new XmlDocument();
     try
     {
          kq.Load(strFilename);
     }
     catch{
          return null;
     }
     return kq;
}
```

```
public static void GhiTaiLieu(string strFilename,
                                        XmlDocument doc)
{
     try{
          doc.Save(strFilename);
     }
     catch{
     }
}
```

Lớp xử lý phân số (XL_PHANSO)

Lớp này sẽ thực hiện cài đặt các hàm liên quan đến xử lý và tính toán trên phân số như định nghĩa phép cộng 2 phân số, rút gọn phân số hay cập nhật giá trị từ đối tượng thể hiện.
Khai báo 2 đối tượng lần lượt thuộc về lớp LT_PHANSO và TH_PHANSO để giúp tạo liên kết với tầng xử lý với 2 tầng còn lại là tầng dữ liệu và tầng giao diện.

```
private LT_PHANSO lt_ps = null;
private TH_PHANSO th_ps = null;
```

Cài đặt hàm khởi tạo mặc định để tạo liên kết với đối tượng thể hiện và đối tượng xử lý

```
public XL_PHANSO(LT_PHANSO lt_ps, TH_PHANSO th_ps)
{
    this.lt_ps = lt_ps;
     this.th_ps = th_ps;
    this.th_ps.tuso = this.lt_ps.tuso;
    this.th_ps.mauso = this.lt_ps.mauso;
}
```

Cài đặt phương thức ghi

```
public void Ghi(string strFilename)
{
    this.lt_ps.tuso = this.th_ps.tuso;
    this.lt_ps.mauso = this.th_ps.mauso;
    this.lt_ps.GhiPhanSo(strFilename);
}
```

Cài đặt toán tử +

```
public static XL_PHANSO operator +(XL_PHANSO ps1,XL_PHANSO ps2)
{
    XL_PHANSO kq = new XL_PHANSO(new LT_PHANSO(), new
TH_PHANSO());
    kq.th_ps.tuso = ps1.th_ps.tuso * ps2.th_ps.mauso +
ps2.th_ps.tuso * ps1.th_ps.mauso;
    kq.th_ps.mauso = ps1.th_ps.mauso * ps2.th_ps.mauso;
    return kq;
}
```

Cài đặt hàm cập nhật từ đối tượng xử lý phân số khác

```
public void CapNhat(XL_PHANSO ps)
{
    this.th_ps.tuso = ps.th_ps.tuso;
    this.th_ps.mauso = ps.th_ps.mauso;
}
```

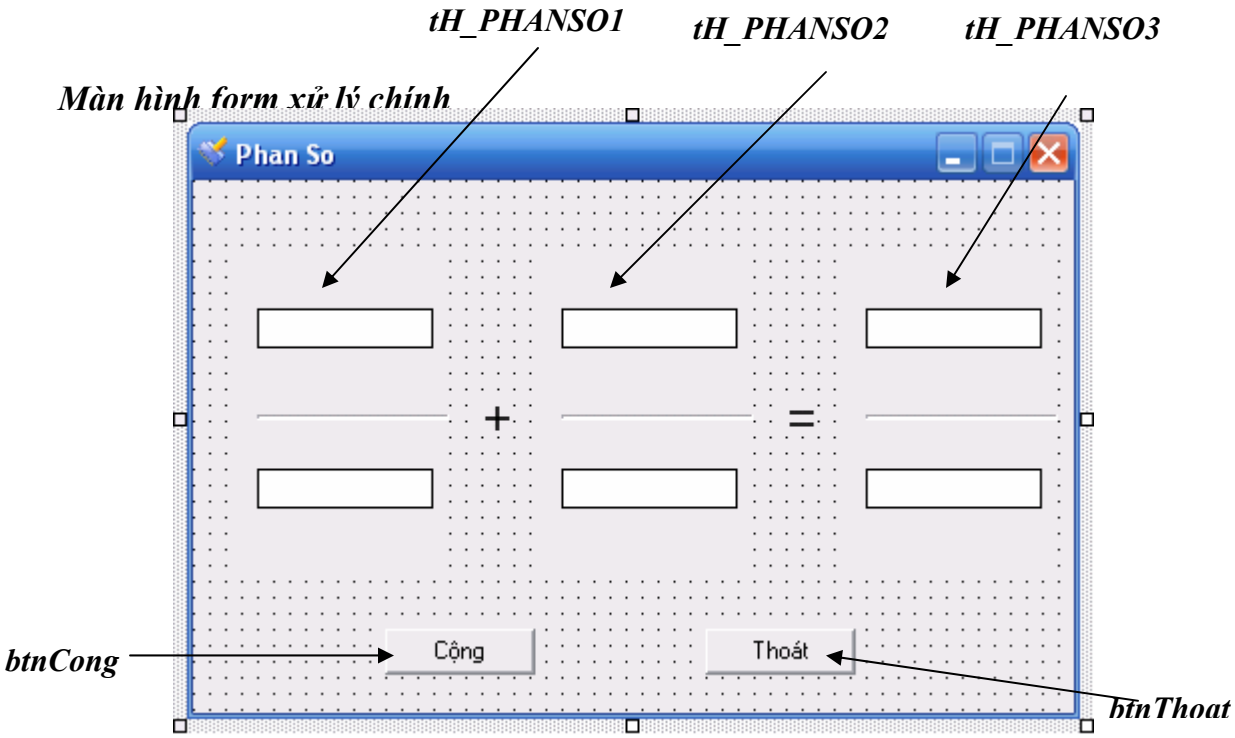
Cài đặt hàm rút gọn phân số

```
public void RutGon()
{
    int tuso = this.th_ps.tuso;
    int mauso = this.th_ps.mauso;
    int maxUC = TimMaxUocChung(tuso,mauso);
    tuso = tuso/maxUC;
    mauso = mauso/maxUC;
    this.th_ps.tuso = tuso;
    this.th_ps.mauso = mauso;
}
```

Để rút gọn ta cần tính ước chung lớn nhất, có thể cài đặt hàm này chung với lớp XL_PHANSO

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

Thực hiện cài đặt màn hình chính (MainFrm)



Trong form chính sẽ thực hiện khai báo 3 đối tượng xử lý phân số

```
private PhanSo.XL_PHANSO xl_PhanSo1;
private PhanSo.XL_PHANSO xl_PhanSo2;
private PhanSo.XL_PHANSO xl_PhanSo3;
```

Thực hiện khởi tạo 3 đối tượng xử lý phân số vừa khai báo

```
public MainFrm()
{
     //
     // Required for Windows Form Designer support
     InitializeComponent();

     xl_PhanSo1 = new XL_PHANSO(new LT_PHANSO("phanso1.xml"),
                                tH_PHANSO1);
     xl_PhanSo2 = new XL_PHANSO(new LT_PHANSO("phanso2.xml"),
                                tH_PHANSO2);
     xl_PhanSo3 = new XL_PHANSO(new LT_PHANSO(""),tH_PHANSO3);
}
```

Viết hàm xử lý cho các nút chức năng trên form:

Hàm xử lý cho nút Cộng

```
private void btnCong_Click(object sender, System.EventArgs e)
{
    XL_PHANSO kq = xl_PhanSo1 + xl_PhanSo2;
    xl_PhanSo3.CapNhat(kq);
    xl_PhanSo3.Ghi("ketqua.xml");
    xl_PhanSo3.RutGon();
}
```

Hàm xử lý cho nút Thoat

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

Tạo các tập tin phanso1.xml, phanso2.xml, có định dạng như ví dụ ở trên.

Thực hiện biên dịch và chạy thử chương trình.

3. Nhận xét :

✓ Thực hiện cài đặt với kiến trúc 3 lớp sẽ giúp chương trình dễ dàng thay đổi, tái sử dụng lại chương trình.

Ví dụ:

❖ TH_PHANSO không thể hiện tử số và mẫu số bằng TextBox nữa mà thay bằng control khác (ví dụ như ***MyControl*** thì cũng không ảnh hưởng, lúc đó chỉ cần thay đổi code trong phần property tử số và mẫu số mà thôi.

```
public int tuso{
    set{
        this.MyControl.Value = value.ToString();
    }
    get{
        return int.Parse(this.MyControl.Value);
    }
}

public int mauso
{
    set
    {
        this.MyControl.Value = value.ToString();
    }
    get
    {
        return int.Parse(this.MyControl.Value);
    }
}
```

❖ Khi không lưu trữ bằng XML mà chuyển sang dùng cơ sở dữ liệu thì ta chỉ cần thay code phần LT_PHANSO, mà không cần thay đổi code phần TH_PHANSO, cũng như XL_PHANSO.

✓***Chú ý:***

- Không phụ thuộc phương pháp lập trình.
- Mỗi nghiệp vụ không nhất thiết chỉ được giải quyết bởi 3 đối tượng.
- Không là một kiến trúc "siêu việt".