Интегрированные компьютерные системы проектирования и анализа
Лабораторная №3
Создание COM сервера

COM-сервер представляет собой приложение или динамически подгружаемую библиотеку (dll), в рамках которых проходит жизненный цикл COM-объектов. Приложение отправляющее запросы к COM-объектам внутри COM-сервера называют COM-клиентом. Различают три типа COM-серверов:

1. Внутренний сервер (in-process server) реализуется динамическими библиотеками, которые подключаются к приложению-клиенту и работают с ним в одном адресном пространстве.

2. Локальный сервер (local server) реализуется отдельным процессом, который работает на том же компьютере, что и клиент.

3. Удаленный сервер (remote server) реализуется процессом, работающим, в общем случае, не на том компьютере, на котором выполняется клиент.

При работе с внутренним сервером, получаемый клиентом указатель интерфейса ссылается на реальный COM-объект функционирующий в адресном пространстве клиентского процесса. При работе с локальным либо удаленным сервером, получаемый клиентом указатель интерфейса ссылается на специальный объект-заместитель (proxy-object), который функционирует внутри клиентского процесса. Получив вызов от клиента, заместитель упаковывает его параметры – этот процесс называется маршалингом – и при помощи служб операционной системы передает вызов в процесс сервера. В процессе сервера запрос передается еще одному специализированному объекту - заглушке (stub), которая распаковывает параметры вызова – этот процесс называется демаршалингом – и предает его требуемому объекту.

Механизм операционной системы, который передает вызовы из клиентского процесса в процесс локального сервера, называется Local Procedure Call (LPC). Аналогичный механизм для удаленного сервера называется Remote Procedure Call (RPC), и реализуется средствами DCOM.

DCOM представляет собой расширение модели COM, ориентированное на поддержку распределенных объектных приложений, функционирующих в сети.

Для создания экземпляра класса COM-объекта используется специальный объект – фабрика класса. Для каждого COM-объекта должна существовать фабрика класса. Фабрика класса, которая также является COM-объектом, должна поддерживать интерфейс IClassFactory.

В этом интерфейсе ключевым является метод CoCreateInstance, который, собственно, и создает экземпляр класса COM-объекта.

Информация об интерфейсах объектов COM-сервера объединяется в библиотеки типов (type library). Библиотеки типов записываются при помощи специального языка IDL (Interface Definition Language), который имеет много общего с C++. Каждая библиотека типов имеет собственный GUID и поддерживает интерфейс ITypeLib, который дает возможность с ней работать как с единым объектом.

Для генерации GUID в Visual Studio необходимом воспользоваться Tools → Create Guid.

В качестве пример рассмотрим создание COM-сервера. Для этого создадим новый проект типа Class Library. Его код приведен в листинге *CSharpServer.cs*.

Простейший пример клиента показан в листинге *CSharpClinet.cs.* Тоже, но языке C++ приведено в листинге CPPClient.cpp.

Для регистрации сервера используйте команду

```
regasm CSharpServer.dll /tlb:CSharpServer.tlb
```

Файл: *CSharpServer.cs*

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Runtime.InteropServices;
using System.Windows.Forms;

// compile with: /target:library
// post-build command: regasm CSharpServer.dll /tlb:CSharpServer.tlb

namespace CSharpServer
{
    [Guid("DBE0E8C4-1C61-41f3-B6A4-4E2F353D3D05"),
    ComVisible(true)]
    public interface IManagedInterface
    {
        int PrintHi(string name);
    }

    [Guid("C6659361-1625-4746-931C-36014B146679"),
    ClassInterface(ClassInterfaceType.None),
    ComVisible(true)]
    public class InterfaceImplementation : IManagedInterface
    {
        public int PrintHi(string name)
        {
            MessageBox.Show("Hello world");
            Console.WriteLine("Hello, {0}!", name);
            return 33;
        }
    }

}
```

Файл: *CSharpClinet.cs*

```
using System;
using System.Collections.Generic;
using System.Text;
using CSharpServer;

namespace CSharpConsoleApplication
{
    class Program
    {
        static void Main(string[] args)
        {
            InterfaceImplementation serv = new InterfaceImplementation();
            serv.PrintHi("vasya");
        }
    }
}
```

Файл: *CPPClient.cpp*

```
#include "stdafx.h"

#include <windows.h>
#include <stdio.h>
#include <iostream>

// To use managed-code servers like the C# server,
// we have to import the common language runtime:
#import <mscorlib.tlb> raw_interfaces_only
#import "..\AddControl\bin\Debug\CSharpServer.tlb" no_namespace named_guids
```

```
int main(int argc, char* argv[])
{
	IManagedInterface *cpi = NULL;
	int retval = 1;

	// Initialize COM and create an instance of the InterfaceImplementation class:
	CoInitialize(NULL);

	HRESULT hr = CoCreateInstance(CLSID_InterfaceImplementation,
		NULL, CLSCTX_INPROC_SERVER,
		IID_IManagedInterface, reinterpret_cast<void**>(&cpi));

	if (FAILED(hr))
	{
		printf("Couldn't create the instance!... 0x%x\n", hr);
	}
	else
	{
		printf("Calling function.\n");
		fflush(stdout);
		// The variable cpi now holds an interface pointer
		// to the managed interface.
		// notice that the ASCII characters are
		// automatically marshaled to Unicode for the C# code.
		if (cpi->PrintHi("Vasya") == 33)
			retval = 0;
		printf("Returned from function.\n");

		cpi->Release();
		cpi = NULL;
	}

	// Be a good citizen and clean up COM:
	CoUninitialize();
	std::cin.ignore(1);
	return retval;
	}
```

Вариант 1.
Реализовать COM сервер, в котором:
1. Описаны классы объектов «Линия» и «Треугольник».
2. Все операции с внутренней структурой объектов производятся через интерфейсы ILine и ITriangle, а все операции связанные с выводом графики на экран производятся через интерфейс IGraphicObject.
3. На экран выводится желтый треугольник с синей, красной и зеленой сторонами.
4. Реализовать два клиента, которые демонстрирует все возможности COM сервера. Они клиент должен быть реализован на C#, другой на произвольном языке без поддержки платформы .NET.

Вариант 2.
Реализовать COM сервер, в котором:
1. Описаны классы объектов «Точка» и «Прямая».
2. Все операции с внутренней структурой объектов производятся через интерфейсы IPoint и IRightLine, а все операции связанные с записью и чтением из файла производятся через интерфейс IPersistent.
3. Из некоторого файла читается информация о 2 точках и 1 прямой. В результирующий файл записывается информация о прямой, которая проходит через две точки, и о точке пересечения этих прямых.
4. Реализовать два клиента, которые демонстрирует все возможности COM сервера. Они клиент должен быть реализован на C#, другой на произвольном языке без поддержки платформы .NET.

Вариант 3.
Реализовать COM сервер, в котором:
1. Описаны классы объектов «Скаляр» и «Вектор».
2. Все операции с внутренней структурой объектов производятся через интерфейсы IScalar и IVector, а все операции связанные с записью и чтением данных из строки производятся через интерфейс ISerializable.
3. С клавиатуры считывается **одна** строка, а из нее читается информация о скаляре и векторе. В результирующую строку записывается информация о векторе - произведении введенных скаляра и вектора и о произведении элементов вектора. Результирующая строка выводится на экран.
4. Реализовать два клиента, которые демонстрирует все возможности COM сервера. Они клиент должен быть реализован на C#, другой на произвольном языке без поддержки платформы .NET.

Вариант 4.
Реализовать COM сервер, в котором:
1. Описаны классы объектов «Линия» и «Прямоугольник».
2. Все операции с внутренней структурой объектов производятся через интерфейсы ILine и IRectangle, а все операции связанные с выводом графики на экран производятся через интерфейс IGraphicObject.
3. На экран выводится зеленый прямоугольник с серой, желтой, красной и фиолетовой сторонами.
4. Реализовать два клиента, которые демонстрирует все возможности COM сервера. Они клиент должен быть реализован на C#, другой на произвольном языке без поддержки платформы .NET.

Вариант 5.
Реализовать COM сервер, в котором:
1. Описаны классы объектов «Точка» и «Окружность».
2. Все операции с внутренней структурой объектов производятся через интерфейсы IPoint и ICircumference, а все операции связанные с записью и чтением из файла производятся через интерфейс IPersistent.
3. Из некоторого файла читается информация о 3 точках и 1 окружности. В результирующий файл выводится информация о точках, которые находятся внутри окружности. В этот же файл записывается информация об окружности, центр которой совпадает с первой введенной точкой, а сама окружность проходит через вторую введенную точку.
4. Реализовать два клиента, которые демонстрирует все возможности COM сервера. Они клиент должен быть реализован на C#, другой на произвольном языке без поддержки платформы .NET.

Вариант 6.
Реализовать COM сервер, в котором:
1. Описаны классы объектов «Скаляр» и «Матрица».
2. Все операции с внутренней структурой объектов производятся через интерфейсы IScalar и IMatrix, а все операции связанные с записью и чтением данных из строки производятся через интерфейс ISerializable.
3. С клавиатуры считывается **одна** строка, а из нее читается информация о скаляре и матрице. В результирующую строку записывается информация о матрице - произведении введенных скаляра и матрицы и о сумме элементов матрицы. Результирующая строка выводится на экран.
4. Реализовать два клиента, которые демонстрирует все возможности COM сервера. Они клиент должен быть реализован на C#, другой на произвольном языке без поддержки платформы .NET.