**《地理信息系统设计与开发》上机报告**

**（地信2017级）**

**姓 名 张清昱**

**学 号 07172336**

**班 级 地信17-1班**

**环境与测绘学院**

**2020.5**

**上机实验1 基于PostGIS的地理空间数据库的建立**

**一.上机目的**

了解并掌握地理空间数据库、地理空间数据引擎的概念。掌握将地理空间数据导入PostgreSQL中的方法，并使用某一款桌面GIS软件查看导入到PostGIS数据库中的空间数据，分析PostGIS中空间数据表的结构。

**二.上机软件环境**

OS：MacOS X；

应用软件：QGIS、PostgreSQL、PostGIS；

**三.上机内容**

（1）将.shp文件导入PostgreSQL数据库中；

（2）使用QGIS软件查看导入到PostGIS数据库中空间数据；

（3）分析PostGIS中空间数据表的结构。

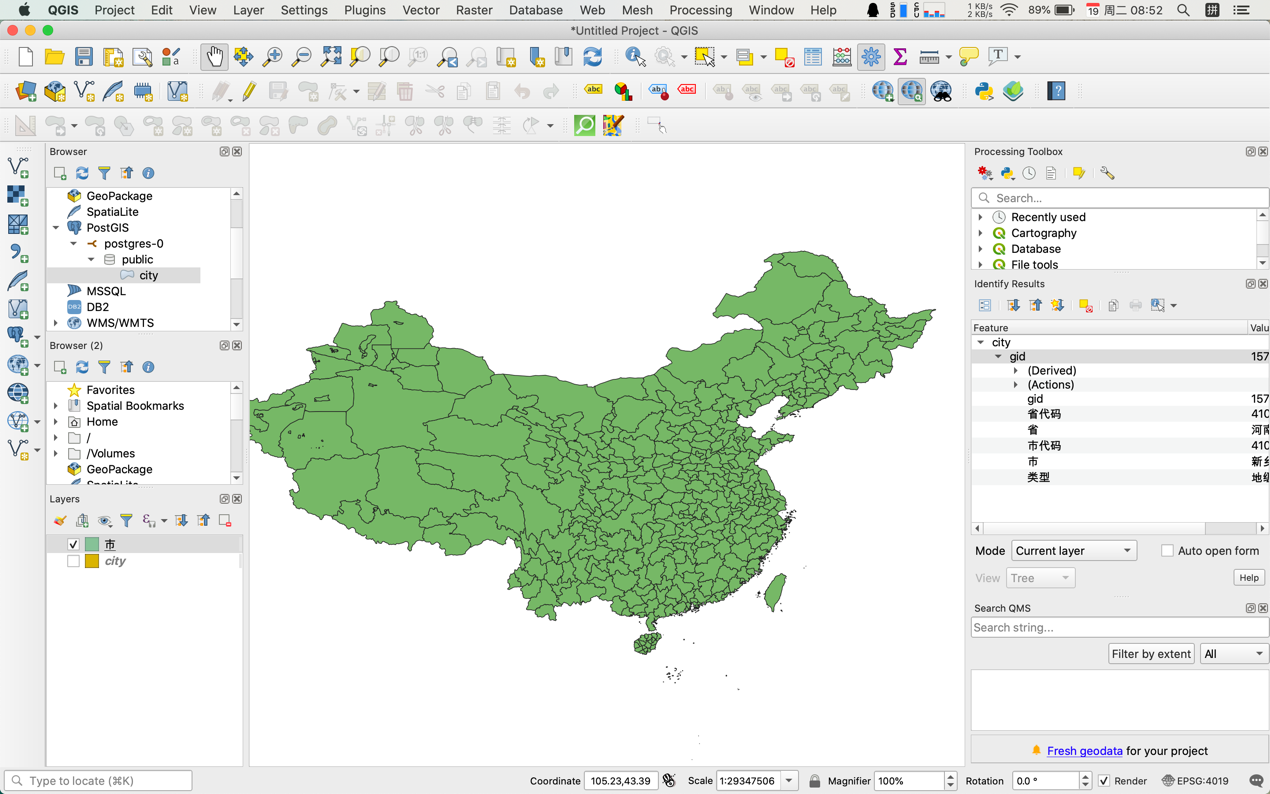
**四.上机要求**

要求上机报告格式规范，包括上机实验目的、软件环境、上机内容、上机要求、关键步骤、上机实验结果与体会等内容。独立完成上机实验。

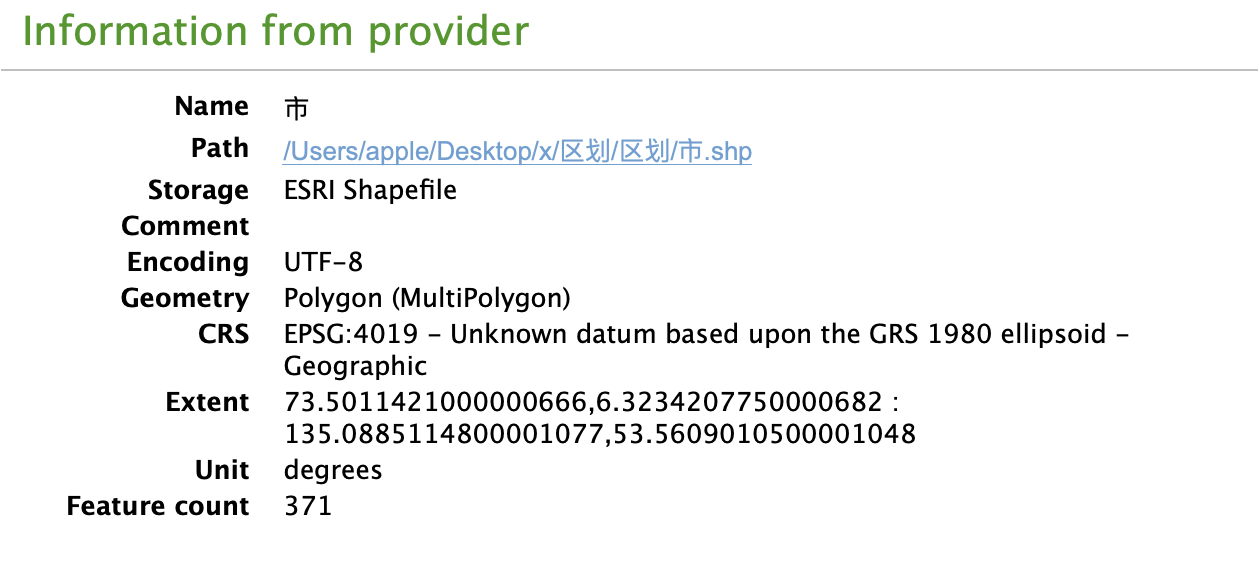
**五.关键步骤**

**1.使用QGIS打开.shp文件，查看.shp文件属性**

（1）打开QGIS，将.shp文件拖入其中，即可完成打开操作

****

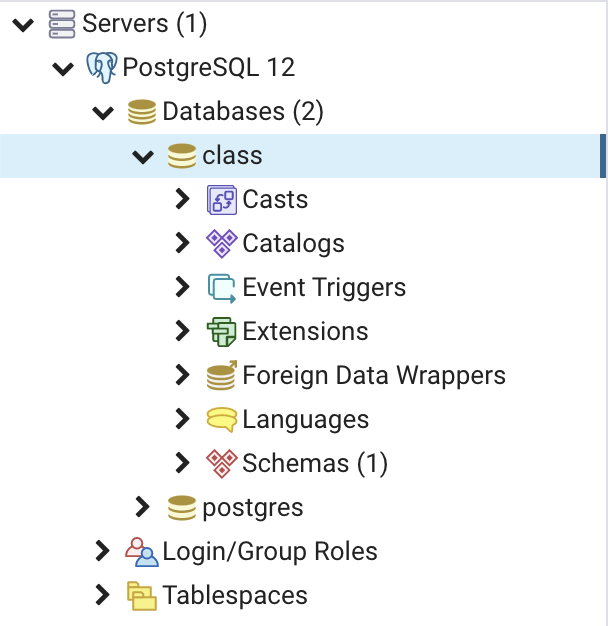
（2）在图层上右键选择Properties打开属性界面，查看.shp文件的图层属性，此步作用为查看SRID号，便于后续将.shp文件导入数据库，该文件的属性信息如图所示，可得到该图层的坐标参考系为EPGS:4019，编码方式为UTF-8



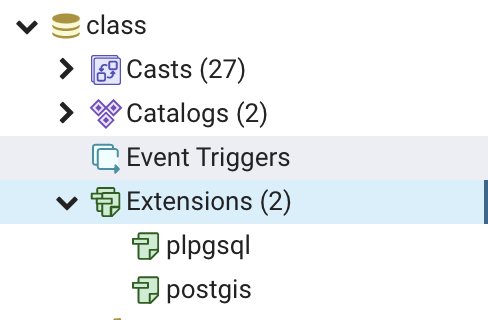
**2.将.shp文件导入postgresql数据库中**

（1）首先安装postgresql数据库与扩展postgis

（2）新建数据库class，作为存放本次实验的数据库



（3）创建完成数据库后，打开该数据库的postgis空间拓展，若已经启动成功，则会在extension中显示postgis，如图所示：



（4）由于Mac版postgis未附带Postgis Shapefile Import/Export Manager工具，使用命令行进行空间数据导入。

（5）首先创建sql文件，使用以下命令：

shp2pgsql -W UTF-8 -s 4019 /Users/apple/Desktop/x/区划/区划/市.shp city>city.sql

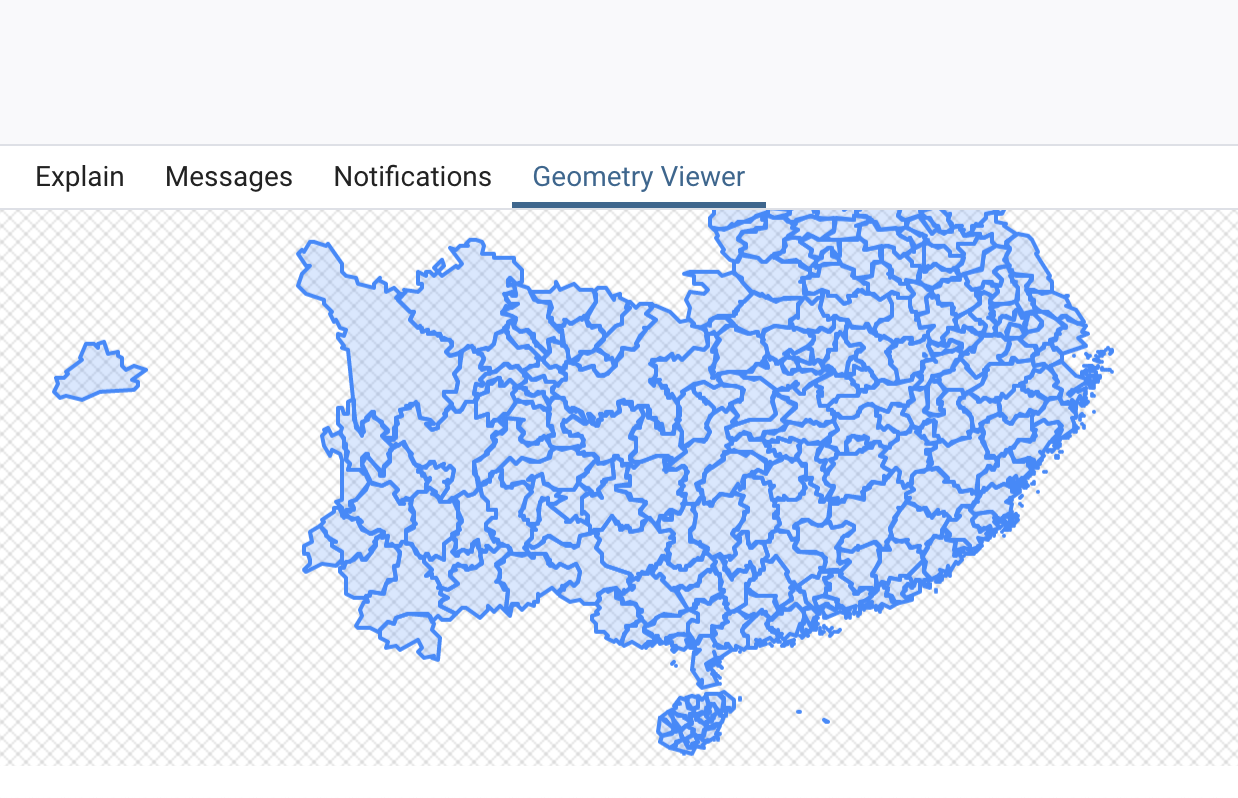
其中，shp2pgsql是安装后自带的终端命令，-W UTF-8指以UTF-8编码方式写入，此时参数指定的编码方式需要与之前查看的相同；-s 4019即为指定空间数据的投影方式；后续参数为需要导入.shp文件的路径，大于号左边是导入数据库后表的名称，右边是建立的sql文件的名称，执行后发现目录下已经创立了sql文件：



（6）此时已经创立了sql文件，将sql语句导入数据库即可，此时同样使用命令行语句：psql -U postgres -d class -f city.sql，格式为psql -U 用户名 -d 数据库名 -f sql脚本.sql，运行命令，将数据插入数据库，运行命令如图所示：

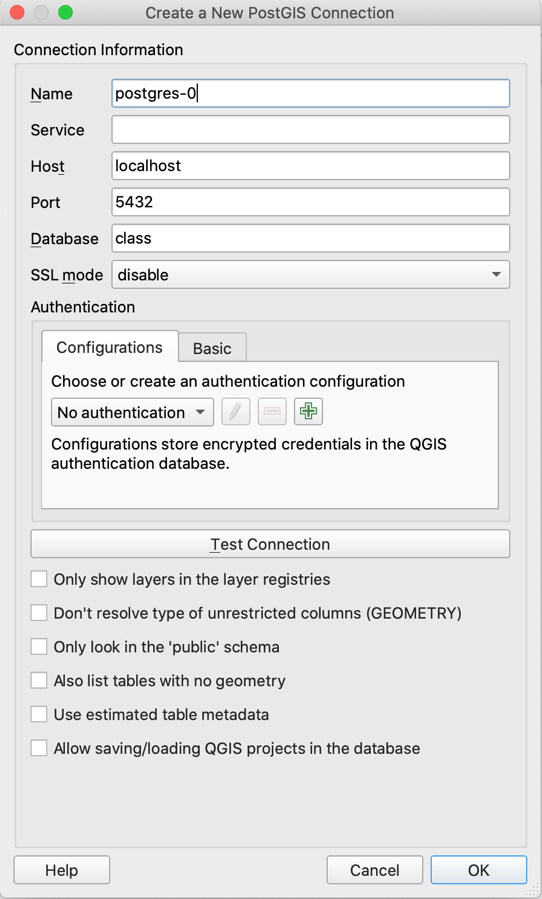


（7）此时可以打开pgadmin查看数据是否成功导入，在数据库calss中找到表city，右键选择查看所有数据，可以发现能够进行地理几何预览：

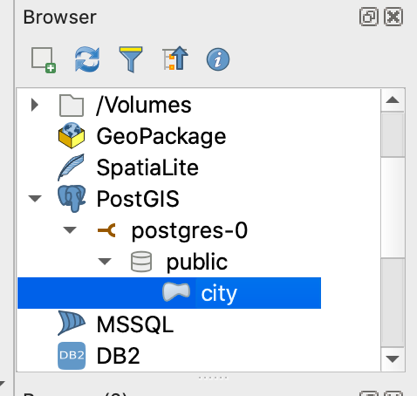


**3.在QGIS中链接postgresql数据库**

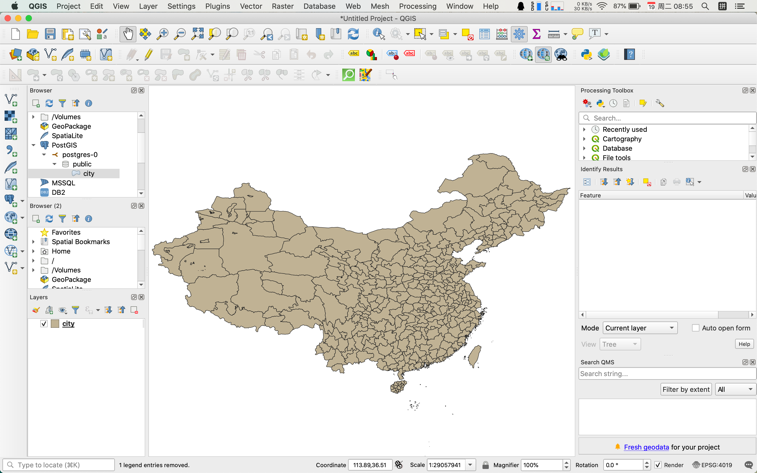
（1）左侧浏览器中选择PostGIS，右键新建链接



（2）连接成功后，会出现之前创立的数据表city



（3）浏览器中双击该表，即可添加该空间数据到该桌面GIS软件中：



**六.上机实验结果与体会**

图1为在QGIS中打开的属性表，图2为在postgresql中的数据表，不难看出数据库中多了一个geom列，该字段数据类型为WKT，猜测该列与数据在关系数据库的组织方式有很大的关系，通过该字段可以进行操作来进行简单的几何判断

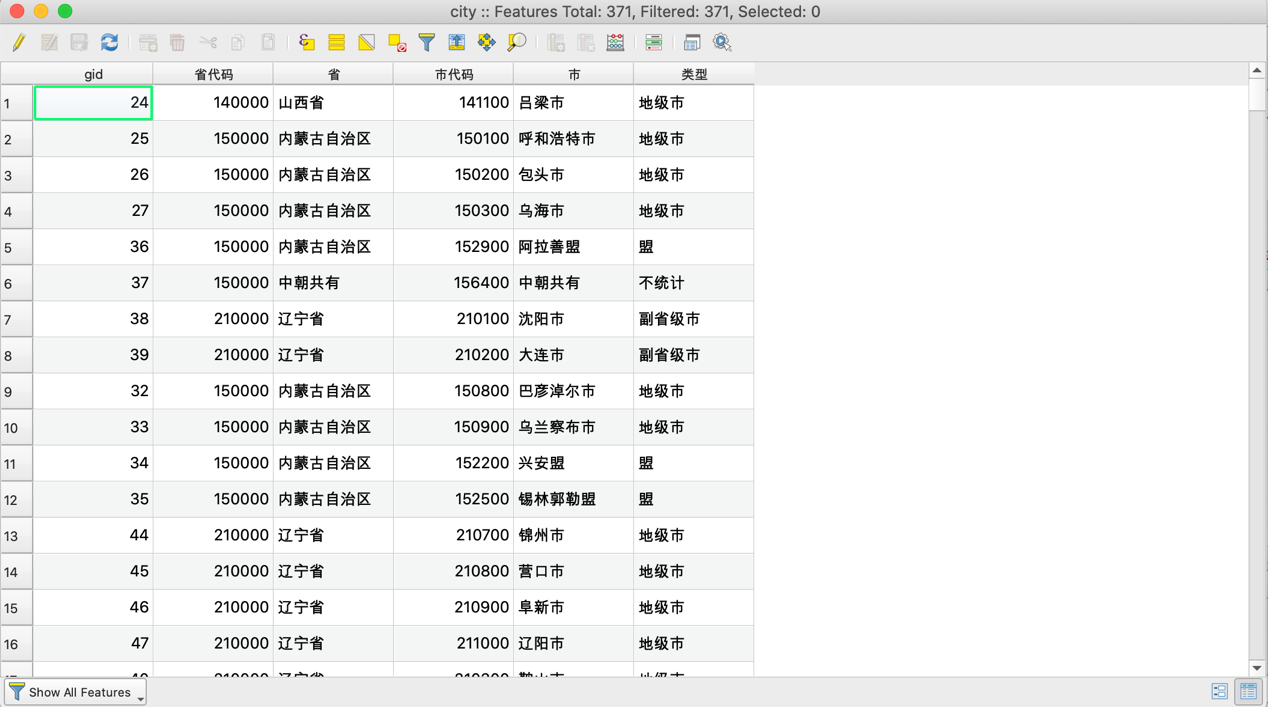


图 1



图 2

**上机实验2 基于组件的常用GIS功能实现**

1. **实验目的**

掌握基于ArcGIS Engine的GIS基本功能的开发方法。

1. **软件环境**

在入门实验的基础上用C#，结合ArcGIS Engine开发一个包含GIS基本操作的应用程序，实现GIS基本操作（放大、缩小、漫游、全图显示等）、打开/保存Mxd地图文档、距离量算／面积量算等。

1. **实验内容**

1．完成代码的编写工作，基于ArcGIS Engine实现GIS基础功能。

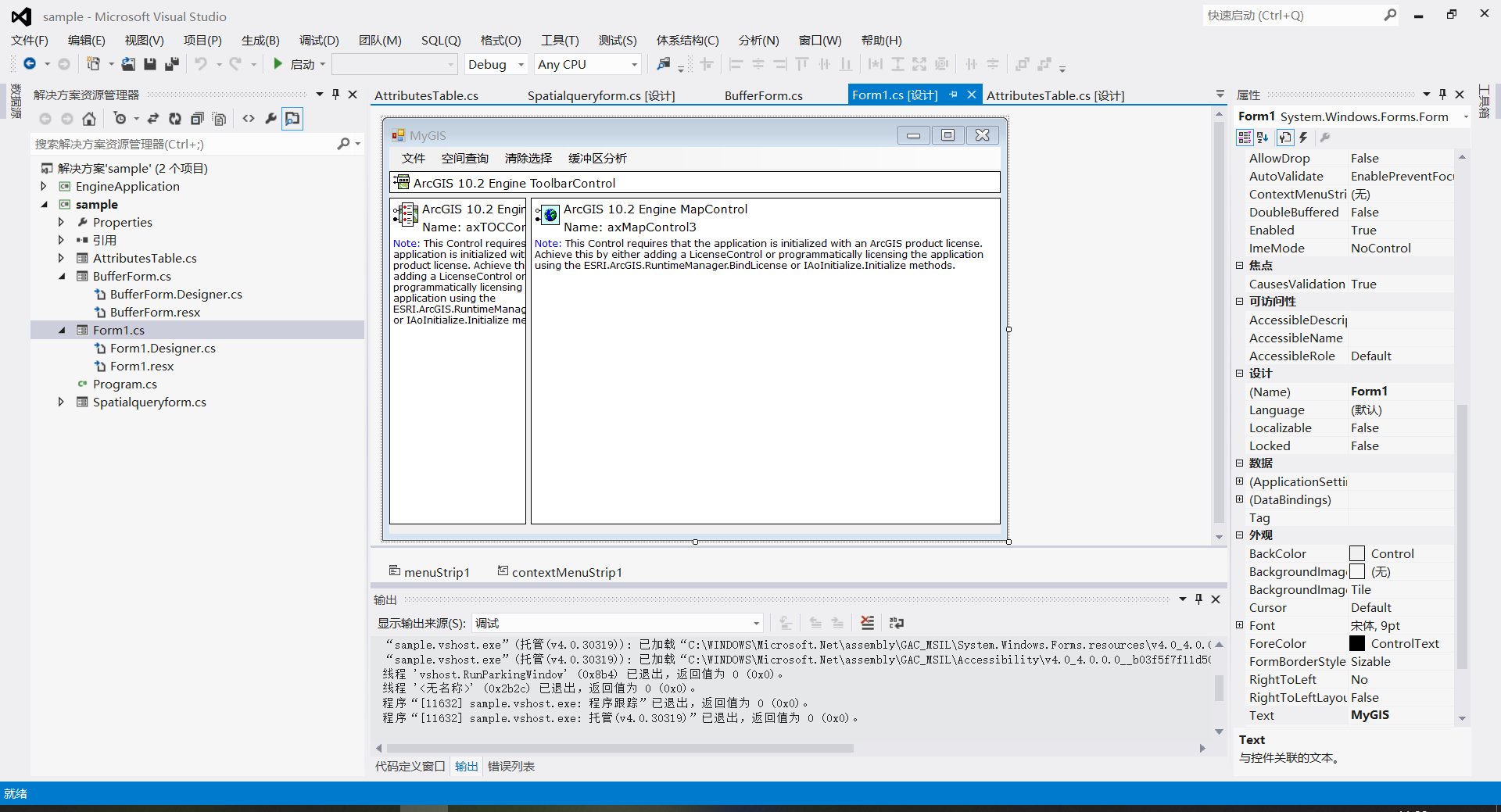
2．了解利用ComGIS进行应用型地理信息系统建设工作的基本思路。

3．独立实验，认真、按时完成上机实验报告。

**四、关键步骤**

**1.窗口设计**

导入控件，设计主界面窗口，功能加入如菜单栏、工具条（放大、缩小、漫游、全图显示等）、显示界面、属性界面等。如图所示。

 **2.加入全局变量**

IMapDocument mapDocument;

ESRI.ArcGIS.Carto.ILayer pGlobalFeaturelayer = null;

**3.导入AE库**

using System;  
 using System.Collections.Generic;  
 using System.ComponentModel;  
 using System.Data;  
 using System.Drawing;  
 using System.Linq;  
 using System.Text;  
 using System.Threading.Tasks;  
 using System.Windows.Forms;  
 using ESRI.ArcGIS.Carto;  
 using ESRI.ArcGIS.Geometry;  
 using ESRI.ArcGIS.Geodatabase;  
 using ESRI.ArcGIS.Controls;  
 using ESRI.ArcGIS.Display;  
 using ESRI.ArcGIS.esriSystem;  
 using ESRI.ArcGIS;  
 using ESRI.ArcGIS.Geoprocessor;  
 using ESRI.ArcGIS.Geoprocessing;

**4.功能具体实现**

打开Mxd文档：

private void 打开Mxd文档ToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog OpenMXD = new OpenFileDialog();

OpenMXD.Title = "打开地图";

OpenMXD.InitialDirectory = "E:";

OpenMXD.Filter = "Map Documents (\*.mxd)|\*.mxd";

if (OpenMXD.ShowDialog() == DialogResult.OK)

{

string MxdPath = OpenMXD.FileName;

axMapControl1.LoadMxFile(MxdPath);

}

}

保存Mxd文档：

private void 保存Mxd文档ToolStripMenuItem\_Click(object sender, EventArgs e)

{

try

{

string sMxdFileName = axMapControl1.DocumentFilename;

IMapDocument pMapDocument = new MapDocumentClass();

if (sMxdFileName != null && axMapControl1.CheckMxFile(sMxdFileName))

{

if (pMapDocument.get\_IsReadOnly(sMxdFileName))

{

MessageBox.Show("本地图文档是只读的,不能保存!");

pMapDocument.Close();

return;

}

}

else

{

SaveFileDialog pSaveFileDialog = new SaveFileDialog();

pSaveFileDialog.Title = "请选择保存路径";

pSaveFileDialog.OverwritePrompt = true;

pSaveFileDialog.Filter = "ArcMap文档 (\*.mxd) |\* .mxd1ArcMap模板(\* .mxt)|\*.mxt";

pSaveFileDialog.RestoreDirectory = true;

if (pSaveFileDialog.ShowDialog() == DialogResult.OK)

{

sMxdFileName = pSaveFileDialog.FileName;

}

else

{ return; }

}

pMapDocument.New(sMxdFileName);

pMapDocument.ReplaceContents(axMapControl1.Map as IMxdContents);

pMapDocument.Save(pMapDocument.UsesRelativePaths, true);

pMapDocument.Close();

MessageBox.Show("保存地图文档成功!");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

距离量算/面积量算：

private void 面积量算ToolStripMenuItem\_Click(object sender, EventArgs e)

{

flag = 2;

}

private void 距离量算ToolStripMenuItem\_Click(object sender, EventArgs e)

{

flag = 1;

}

private IPolyline DistanceMeasure()

{

ILineElement pLineElement;

IActiveView pActiveView;

IRgbColor pRgbColor;

IRubberBand pRubberBand;

ISimpleLineSymbol pSimpleLineSymbol;

IPolyline pPloyline;

pActiveView = axMapControl1.ActiveView;

pSimpleLineSymbol = new SimpleLineSymbolClass();

pSimpleLineSymbol.Style = esriSimpleLineStyle.esriSLSSolid;

pRgbColor = new RgbColorClass();

pRgbColor.Red = 125;

pSimpleLineSymbol.Color = pRgbColor;

pRubberBand = new RubberLineClass();

pLineElement = new LineElementClass();

pLineElement.Symbol = pSimpleLineSymbol;

pPloyline = pRubberBand.TrackNew(pActiveView.ScreenDisplay, pSimpleLineSymbol as ISymbol) as IPolyline;

pElement = new LineElement();

pElement = pLineElement as IElement;

pElement.Geometry = pPloyline;

double a = pPloyline.Length;

IGraphicsContainer pGraphicsContainer = axMapControl1.ActiveView.FocusMap as IGraphicsContainer;

pGraphicsContainer.AddElement(pElement, 0);

axMapControl1.ActiveView.Refresh();

return pPloyline;

}

private IPolygon AreaMeasure()

{

IRgbColor pRgbColor;

IActiveView pActiveView;

IRubberBand pRubberBand;

IElement pElement;

IGraphicsContainer pGraphicsContainer;

IPolygonElement pPolygonElement;

ISimpleFillSymbol pSimpleFillSymbol;

IPolygon pPolygon;

pActiveView = axMapControl1.ActiveView;

pSimpleFillSymbol = new SimpleFillSymbolClass();

pSimpleFillSymbol.Style = esriSimpleFillStyle.esriSFSBackwardDiagonal;

pRgbColor = new RgbColorClass();

pRgbColor.Red = 250;

pSimpleFillSymbol.Color = pRgbColor;

pRubberBand = new RubberPolygonClass();

pPolygonElement = new PolygonElementClass();

pPolygon = pRubberBand.TrackNew(pActiveView.ScreenDisplay, pSimpleFillSymbol as ISymbol) as IPolygon;

pElement = new PolygonElement();

pElement = pPolygonElement as IElement;

pElement.Geometry = pPolygon;

pGraphicsContainer = axMapControl1.ActiveView.FocusMap as IGraphicsContainer;

pGraphicsContainer.AddElement(pElement, 0);

axMapControl1.ActiveView.Refresh();

return pPolygon;

}

private void axMapControl1\_OnMouseDown(object sender, EventArgs e)

{

switch (flag)

{

case 0:

break;

case 1:

IPolyline polyline = DistanceMeasure();

MessageBox.Show("距离为" + Convert.ToInt64(polyline.Length).ToString());

break;

case 2:

IPolygon polygon = AreaMeasure();

IArea pArea = polygon as IArea;

MessageBox.Show("面积为" + Convert.ToInt64(Math.Abs(pArea.Area)).ToString());

break;

default:

break;

this.axMapControl1.Map.ClearSelection();

switch (mTool)

{

case "SpatialQuery":

ESRI.ArcGIS.Carto.IActiveView qActiveView = this.axMapControl1.ActiveView;

ESRI.ArcGIS.Geometry.IPoint pPoint = qActiveView.ScreenDisplay.DisplayTransformation.ToMapPoint(e.x, e.y);

ESRI.ArcGIS.Geometry.IGeometry pGeometry = null;

switch (this.mQueryModel)

{

case 0:

pGeometry = this.axMapControl1.TrackRectangle();

break;

case 1:

pGeometry = this.axMapControl1.TrackCircle();

break;

case 2:

pGeometry = AreaMeasure();

axMapControl1.ActiveView.GraphicsContainer.DeleteAllElements();

axMapControl1.Refresh();

break;

}

ESRI.ArcGIS.Carto.IFeatureLayer pFeatureLayer = this.axMapControl1.Map.get\_Layer(this.mLayerIndex) as ESRI.ArcGIS.Carto.IFeatureLayer;

DataTable pDataTable = this.LoadQueryResult(axMapControl1, pFeatureLayer, pGeometry);

panel1.Visible = true;

this.dataGridView1.DataSource = pDataTable.DefaultView;

this.dataGridView1.Refresh();

break;

default:

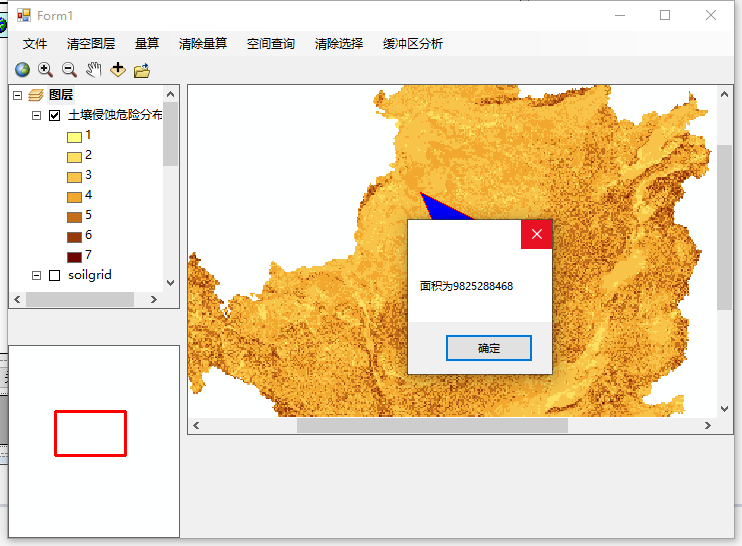
break;

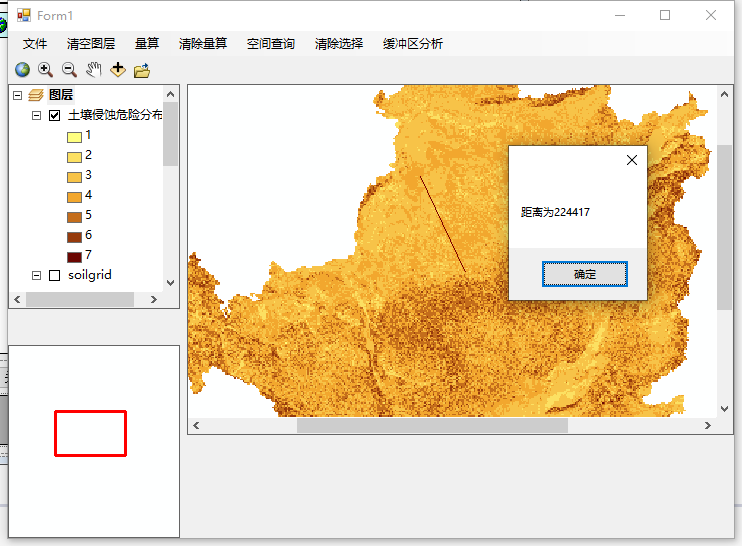
}

}

}

效果如下：





清除选择：

private void 清除选择ToolStripMenuItem1\_Click(object sender, EventArgs e)

{

this.panel1.Visible = false;

mTool = null;

}

空间查询：

public SpatialQueryForm()

{

{

InitializeComponent();

}

private ESRI.ArcGIS.Controls.AxMapControl m\_mapControl;

public int mQueryModel;

public int mLayerIndex;

public SpatialQueryForm(ESRI.ArcGIS.Controls.AxMapControl mapControl)

{

InitializeComponent();

this.m\_mapControl = mapControl;

}

private void SpatialQueryForm\_Load(object sender, EventArgs e)

{

if (m\_mapControl.LayerCount <= 0)

return;

for (int i = 0; i < m\_mapControl.LayerCount; ++i)

{

cobLayer.Items.Add(m\_mapControl.get\_Layer(i).Name);

}

this.cobSearchStyle.Items.Add("矩形查询");

this.cobSearchStyle.Items.Add("圆形查询");

this.cobSearchStyle.Items.Add("多边形查询");

cobLayer.SelectedIndex = 0;

cobSearchStyle.SelectedIndex = 0;

}

private void button1\_Click(object sender, EventArgs e)

{

this.DialogResult = DialogResult.OK;

if (this.cobLayer.Items.Count <= 0)

{

MessageBox.Show("还未添加图层");

return;

}

this.mLayerIndex = cobLayer.SelectedIndex;

this.mQueryModel = cobSearchStyle.SelectedIndex;

}

switch (flag)

{

case 0:

break;

case 1:

IPolyline polyline = DistanceMeasure();

MessageBox.Show("距离为" + Convert.ToInt64(polyline.Length).ToString());

break;

case 2:

IPolygon polygon = AreaMeasure();

IArea pArea = polygon as IArea;

MessageBox.Show("面积为" + Convert.ToInt64(Math.Abs(pArea.Area)).ToString());

break;

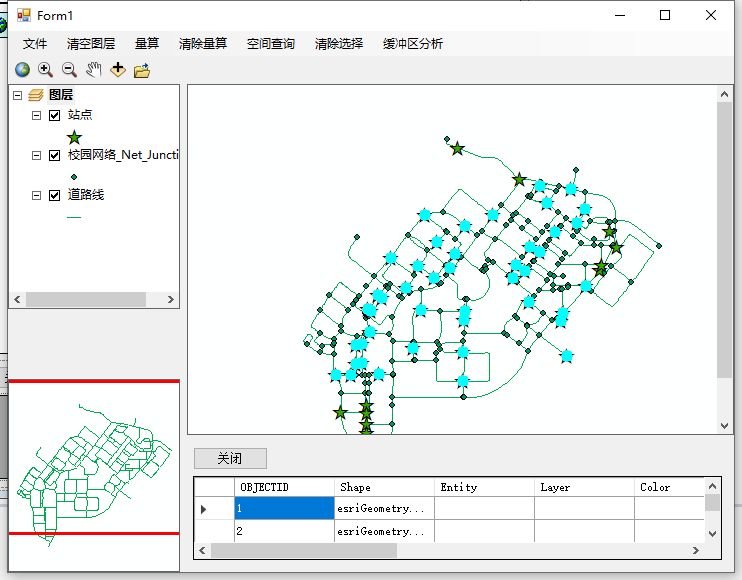
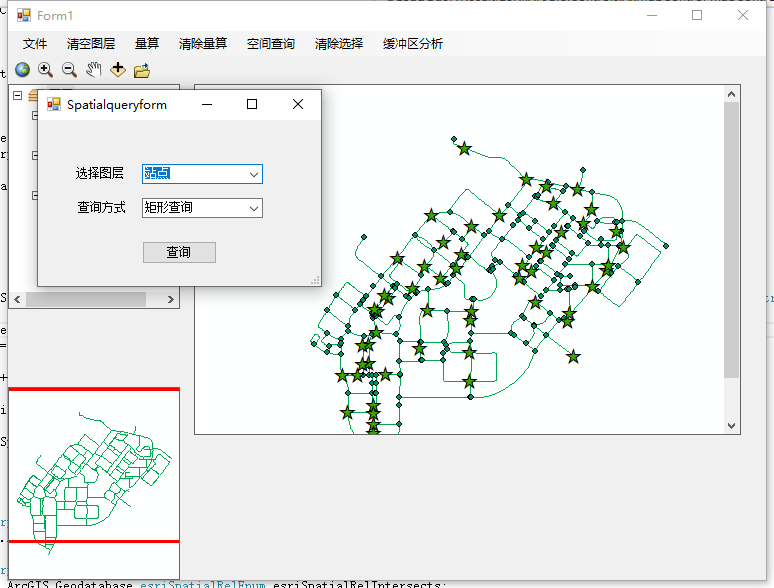
default:

break;

}

}

运行效果如下：



缓冲区分析：

private void 缓冲区分析ToolStripMenuItem\_Click(object sender, EventArgs e)

{

BufferForm bufferForm = new BufferForm(this.axMapControl1.Object);

if (bufferForm.ShowDialog() == DialogResult.OK)

{

//获取输出文件路径

string strBufferPath = bufferForm.strOutputPath;

//缓冲区图层载入到MapControl

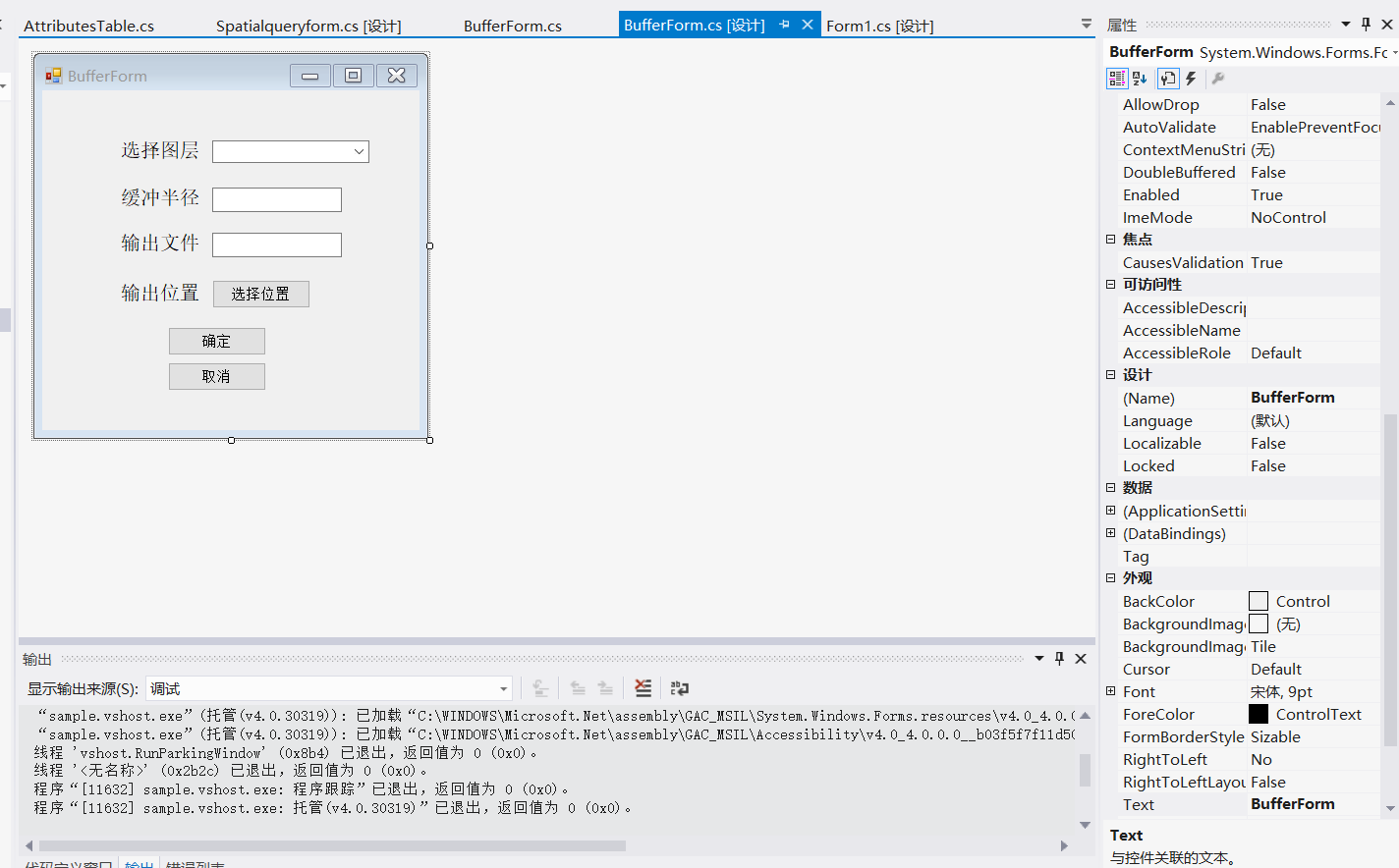
int index = strBufferPath.LastIndexOf("\\");

this.axMapControl1.AddShapeFile(strBufferPath.Substring(0, index), strBufferPath.Substring(index));

}

}

缓冲区分析设计如图所示：



部分代码如下：

namespace sample

{

public partial class BufferForm : Form

{

public BufferForm()

{

InitializeComponent();

}

public BufferForm(object hook)

{

ESRI.ArcGIS.RuntimeManager.Bind(ESRI.ArcGIS.ProductCode.EngineOrDesktop);

IAoInitialize aoInitialize = new AoInitialize();

esriLicenseStatus licenseStatus = esriLicenseStatus.esriLicenseUnavailable;

licenseStatus = aoInitialize.Initialize(esriLicenseProductCode.esriLicenseProductCodeAdvanced);

InitializeComponent();

this.mHookHelper.Hook = hook;

}

private IHookHelper mHookHelper = new HookHelperClass();

public string strOutputPath;

private IFeatureLayer GetFeatureLayer(string layerName)

{

IFeatureLayer pFeatureLayer = null;

for (int i = 0; i < this.mHookHelper.FocusMap.LayerCount; i++)

{

ILayer pLayer = this.mHookHelper.FocusMap.get\_Layer(i);

if (pLayer.Name == layerName)

{

pFeatureLayer = pLayer as IFeatureLayer;

}

}

if (pFeatureLayer != null)

return pFeatureLayer;

else

return null;

}

private void BufferForm\_Load(object sender, EventArgs e)

{

if (null == mHookHelper || null == mHookHelper.Hook || 0 == mHookHelper.FocusMap.LayerCount)

return;

for (int i = 0; i < this.mHookHelper.FocusMap.LayerCount; i++)

{

ILayer pLayer = this.mHookHelper.FocusMap.get\_Layer(i);

cboLayers.Items.Add(pLayer.Name);

}

if (cboLayers.Items.Count > 0)

cboLayers.SelectedIndex = 0;

//设置生成文件的默认输出路径和名称

string tempDir = @"F:\BF";

txtOutputPath.Text = System.IO.Path.Combine(tempDir, ((string)cboLayers.SelectedItem + "\_buffer .shp"));

}

private void btnOutputLayer\_Click(object sender, EventArgs e)

{

//定义输出文件路径

SaveFileDialog saveDlg = new SaveFileDialog();

//检查路径是否存在

saveDlg.CheckPathExists = true;

saveDlg.Filter = "Shapefile (\* .shp)|\*.shp";

//保存时覆盖同名文件

saveDlg.OverwritePrompt = true;

saveDlg.Title = "输出路径";

//对话框关闭前还原当前目录

saveDlg.RestoreDirectory = true;

saveDlg.FileName = (string)cboLayers.SelectedItem + "\_ buffer . shp";

//读取文件输出路径到txtOutputPath

DialogResult dr = saveDlg.ShowDialog();

if (dr == DialogResult.OK)

txtOutputPath.Text = saveDlg.FileName;

}

private void btnBuffer\_Click(object sender, EventArgs e)

{

//缓冲距离

double bufferDistance;

//输入的缓冲距离转换为double

double.TryParse(txtBufferDistance.Text.ToString(), out bufferDistance);

//判断输出路径是否合法

if (!System.IO.Directory.Exists(System.IO.Path.GetDirectoryName(txtOutputPath.Text)) || ".shp" != System.IO.Path.GetExtension(txtOutputPath.Text))

{

MessageBox.Show("输出路径错误!");

return;

}

//判断图层个数

if (mHookHelper.FocusMap.LayerCount == 0)

return;

IFeatureLayer pFeatureLayer = GetFeatureLayer((string)cboLayers.SelectedItem);

if (null == pFeatureLayer)

{

MessageBox.Show("图层" + (string)cboLayers.SelectedItem + "不存在！\r\n");

return;

}

//获取一-个geoprocessor的实例

Geoprocessor gp = new Geoprocessor();

//0verwriteOutput为真时,輸出圏居会覆盖当前文件夫下的同名

gp.OverwriteOutput = true;

//缓冲区保存路径

strOutputPath = txtOutputPath.Text;

//创建一-个Buffer- I具的实例

ESRI.ArcGIS.AnalysisTools.Buffer pbuffer = new ESRI.ArcGIS.AnalysisTools.Buffer(pFeatureLayer, strOutputPath, bufferDistance.ToString());//执行缓冲区分析

IGeoProcessorResult results = null;

results = (IGeoProcessorResult)gp.Execute(pbuffer, null);

//判断缓冲区是否成功生成

if (results.Status != esriJobStatus.esriJobSucceeded)

MessageBox.Show("图层" + pFeatureLayer.Name + ")缓冲区生成失败 ! ");

else

{

this.DialogResult = DialogResult.OK;

MessageBox.Show("缓冲区生成成功! ");

}

}

private void button3\_Click(object sender, EventArgs e)

{

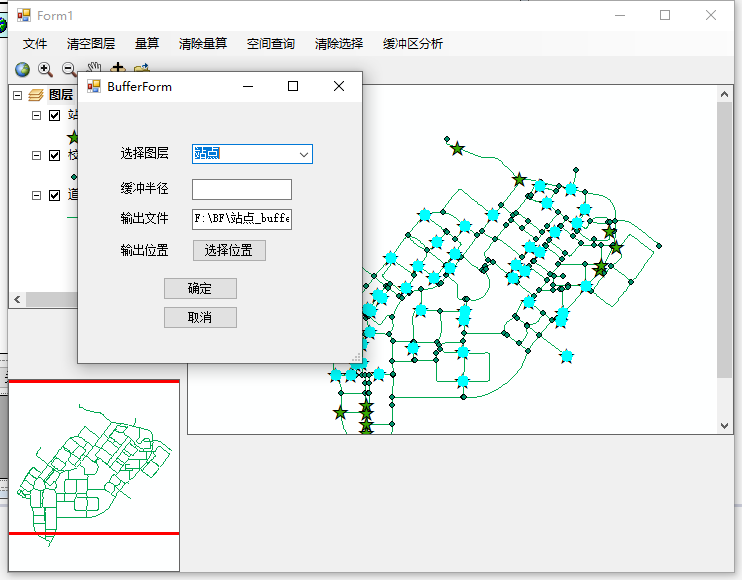
this.Dispose();

}

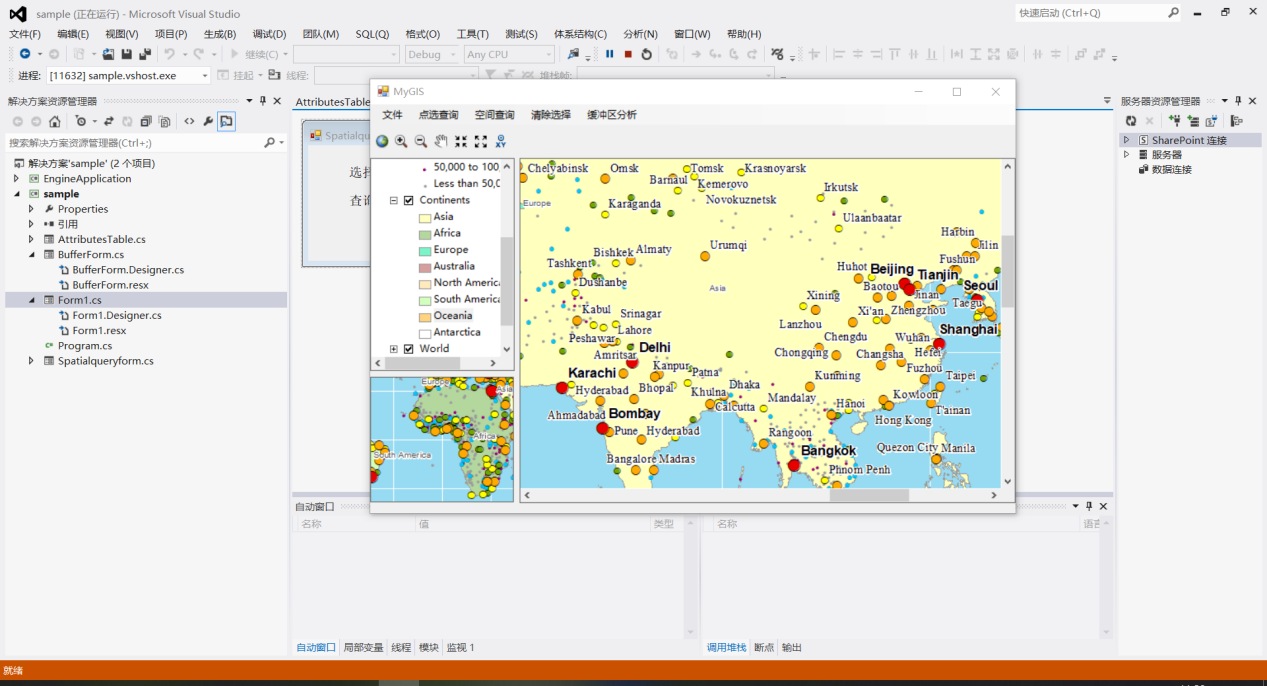
}

}

效果如图下：



运行大体界面结果如图所示。



**五、实验结果与体会**

本次实验是对AE开发技能的一次实验，针对不同的功能需要不同的代码来实现，但编写代码本身并不是全部的难点，代码与控件之间的组织方式也是需要仔细分析与处理的，在代码出现问题的时候也需要有一定的ｄｅｂｕｇ能力。

同时也间接体会到了GIS软件开发设计的工作流程，对GIS设计与开发这门课有了更加深入的认识。