

Geomap 学习教程

（学生用书）

碧海青天

GeoStar 之二次开发工具(GeoMap)

一、概述

GeoMap 作为 GeoStar NT 版的二次开发工具，以 COM (Component Object Model)为基础，以“控件 + 对象”的形式，向二次开发用户提供 GeoStar NT 版的强大功能。

二次开发用户可以利用 GeoMap 以及其它软件供应商提供的大量构件，诸如绘图、多媒体和数据库对象等，来根据终端用户的需要规划设计满足特定需求的应用程序。

GeoMap 由一个 OLE 控件 —GeoMap 和一组近 20 个 OLE 自动化对象构成，应用于标准 Windows 开发环境，用户可以根据需要选择合适的开发工具。GeoMap 是基于 Windows NT 4.0 开发的，因而其开发平台也立足于 Windows NT 4.0，Windows NT 4.0 下的 Visual Basic，Dephi，PowerBuilder，Visual Foxpro 等环境均适合利用 GeoMap 进行的软件开发。

用户可以利用 GeoMap 开发出具有如下功能的应用程序（未列出所有功能）：

- 按照工程、工作区并且分层、地物类组织地图数据。
- 分层、地物类显示地图，如道路、河流、边界。
- 地图的漫游与缩放。
- 绘制几何地物，如点、圆、线、多边形等。
- 显示地图注记。
- 符号化显示地物。
- 点查询方式选中地物。
- 线穿越查询方式选中地物。
- 范围包含查询方式选中地物。
- 计算点、线或面边界的缓冲区。
- 缓冲方式查询地物，可以选中距离点、线或面边界一定距离内的所有地物。
- 利用 SQL 表达式选中地物。
- 对选中的地物进行统计。
- 专题制图
- 在矢量地图上叠加影象，进行影象与矢量图的匹配。
- 添加、修改、删除选中地物的属性数据。
- 添加、修改、删除各种几何类型的地物。

- 添加、修改、删除地图注记。
- 地图数据的打印与打印预览。
- GeoStar 数据与其它 GIS 软件间数据的转换。

对 GeoMap 提供的功能进行分类如下：

功能名称	描述
基本操作	包括所有用户通过鼠标操作获得一个几何对象(圆、矩形框、折线、多边形等)的方法。
数据的组织与维护	包括系统中工作区、工程的创建、打开、关闭，工作区的提交，层的维护，地物类的维护等。
可视化操作	包括地图数据的缩放与漫游，地图数据的分层、地物类显示，显示比例尺的控制，显示范围的控制，层与地物类显示顺序的控制，显示窗口的风格、属性的控制，地物的符号化显示、随图放大显示、注记显示等。
缓冲分析计算	包括点、折线、多边形缓冲区的计算。
地物查询	包括按照点（Pick）、线（Cross）、面（Contain）的查询操作，利用缓冲分析计算功能实现的缓冲查询操作，对查询结果的维护，对查询结果的突出显示，空间对象与相应对象的属性数据数据之间的联系维护，并利用这种联系实现空间对象与属性数据的双向查询等。
编辑操作	包括增加、修改、删除点、线、面、注记等地物对象；线、面边界的内点增加、移动、删除；线、面对象的分割、合并。
专题制图	根据来自各种数据源的属性数据的特征和制图目的，选用适当的制图方法和图型以图形的方式再现属性数据的特征，包括分级统计制图、分区统计制图、质底法制图。
影象叠加	以影象作为矢量图的背景，影象图与矢量坐标匹配，同步缩放。
打印输出	包括打印输出到一页和按照给定的比例尺分页打印两种方式，提供打印输出前的预览功能。
数据交换	实现其它 GIS 软件数据向 GeoStar 数据格式的转换，支持的外部数据格式包括 Arc/Info E00，MapInfo 交换格式，AutoCAD DXF 格式，等。

二、 GeoMap 的设计基础

传统的软件开发模式开发出的应用系统，往往缺乏结构性，其资源使用的效率低下，并且难以与其它的应用程序实现真正的互用，系统的可靠性和可维护性在很大程度上取决于开发人员的经验和能力。基于部件的新型的软件开发技术，为应用系统的开发提供了新的思路，开发人员首先实现可靠的、小的对象模块（部件），或是直接从其他软件开发商获得需要的功能部件；然后利用这些功能部件装配成更复杂的符合应用要求的系统。通过控制各个小部件的可靠性和可维护性，实现对整个应用系统的可靠性及可维护性的控制。

目前，在如何编制部件的技术规范方面，存在着两个阵营：微软公司的部件对象模型（COM）和多厂商的公共对象请求中介结构（CORBA），这两个标准的目标都是让开发人员通过装配部件来生成应用程序，这些部件带有可预测的标准接口，开发人员不必关心部件位于何处、是由谁设计的。

武汉吉奥信息工程技术有限公司研制的 GeoStar 的部件开发平台 GeoMap 是基于微软公司的 Windows NT4.0，因而微软公司的 COM 及其分布式部件模型（DCOM）成为 GeoMap 开发的技术基础。COM 技术是一种面向对象的技术，与 C++ 这样的技术相比，它是一种二进制的标准，所以当部件升级时，应用不需要编译，而且 COM 不局限于某一种开发语言，利用不同语言开发的、支持 COM 的部件可以混合使用。新的 COM 标准有可能实现网络透明，也就是说开发人员使用部件开发的应用，可以直接利用网络上的 COM 实现分布式应用而不需网络编程。

OLE（ActiveX）技术可以看作是微软公司对其所提出的 COM 模型的具体实现，COM 能被广泛接受为部件软件开发模型的一个主要原因就是由于 OLE，OLE 使 COM 不再仅仅是一个模型或规范，它使开发者能够真正体会到基于 COM 的软件设计方式的优越性。OLE / ActiveX 具有相当复杂的技术内容，它包括若干种适应不同需要然而有密切相关的技术，主要包括：结构化存储（Structured Storage），单一数据传输（Uniform Data Transfer），拖放（Drag and Drop），OLE 复合文档（又分成嵌入（Embedding）、链接（Linking）和在线编辑（Visual Editing）），假名（或称智能化名（Name and Binding）），自动化（OLE Automation）和控件（OLE Control），如图 2-1 所示。

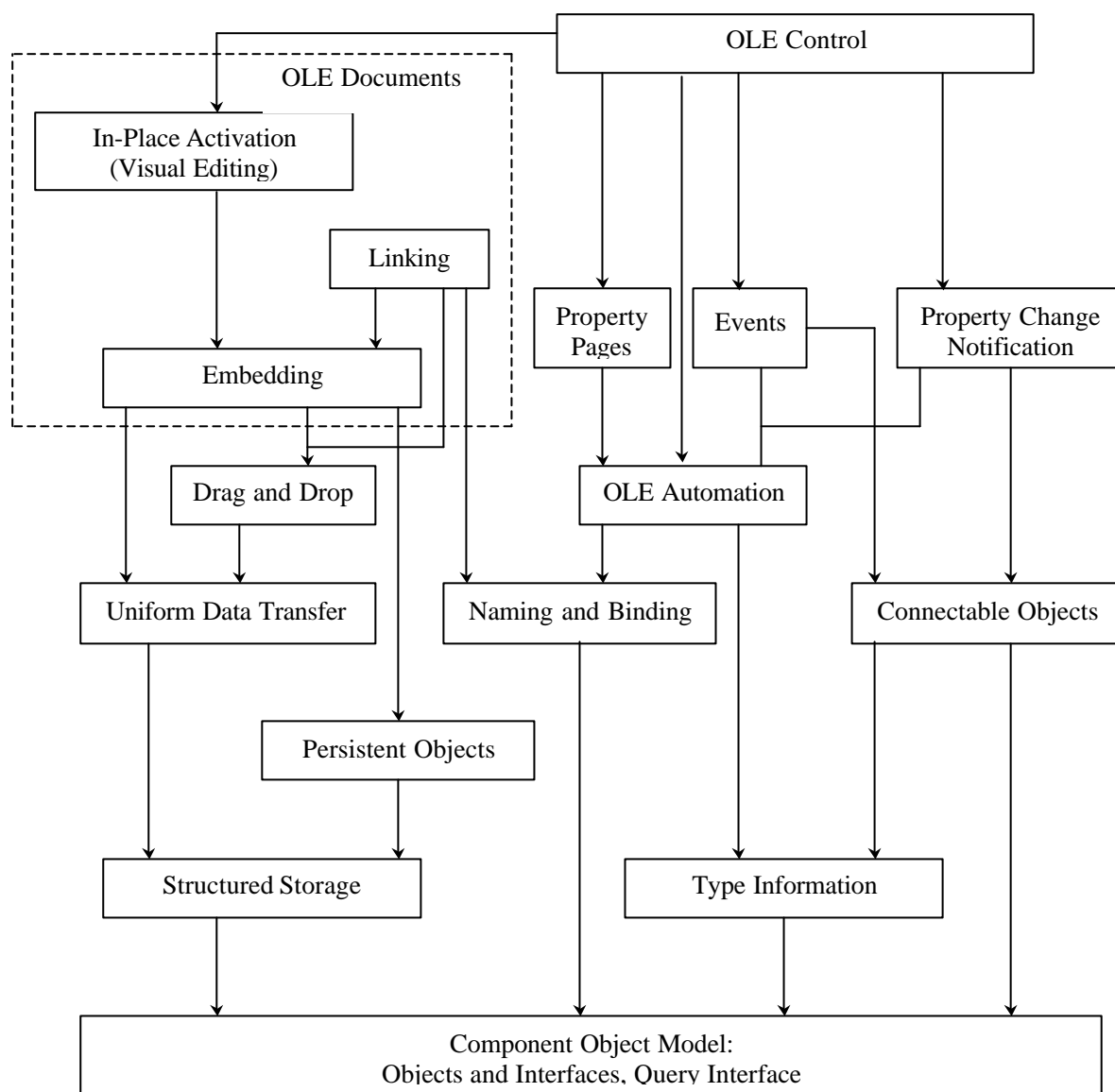


图 2-1 OLE 技术构成

GeoMap 的设计利用了 OLE / ActiveX 技术中的自动化技术及控件技术。GeoMap 提供一个可视化的控件和一组自动化对象，作为自动化服务器（Automation Servers），开发者利用自动化控制器（Automation Controller），如 Visual Basic, Delphi, Visual C⁺⁺, PowerBuilder 和 Visual FoxPro 等，编写代码操纵自动化服务器上提供的属性、方法，为自动化服务器的事件添加事件处理例程，完成部件到应用软件的装配过程，生成适应用户要求的系统，这些系统即是使用自动化服务器的客户（Automation clients）。

GeoMap 为 GIS 应用提供了理想的开发工具，开发者可以用任何一种支持 COM 技术的开发工具，如 Visual Basic, Delphi, Visual C⁺⁺, PowerBuilder 和 Visual FoxPro 等，来操纵 GeoMap 中提供的控件和自动化对象，既可以实现满足自己应用要求的功能，又可以选择设计出体现自

己风格的应用程序界面，将功能化与个性化完美结合。

三、GeoMap 的结构

GeoMap 采用 GeoStar for NT 3.0 (以下简称 GeoStar) 的数据, 可以组织、浏览、查询、打印输出 GeoStar 生成的 GIS 数据, 并能进行空间分析, 如缓冲分析; 提供了与其它 GIS 软件间的数据交换接口, GeoStar 数据与 Arc/Info, MapInfo 等数据的转换。

GeoMap 由一个 OLE 控件 GeoMap 和一组近 20 个 OLE 自动化对象构成, GeoMap 控件提供对 GIS 数据进行操纵的方法和属性, 并且提供了 GIS 数据的可视化界面, 自动化对象则帮助用户实现对 GIS 数据的有效组织和访问。

GeoStar 利用工程、工作区、层、地物类等概念组织数据, GeoMap 则利用自动化对象将这些概念封装起来, 相应地提供了 Project 对象、WorkspaceSet 对象、Workspace 对象、Layer 对象和 Feature 对象, 并且考虑到数据组织中大量出现集合类型的引用的情况, 对这些对象的集合也进行了封装。

(一) GeoMap 对象

1. 数据组织、维护对象

如图 3-1 中所示的用于组织、管理和描述数据的对象, 这些对象包括 Project 对象、WorkspaceSet 对象、Workspace 对象、Layer 对象、Feature 对象以及这些对象的集合对象, 此外还有 CreateInfo 对象用于描述工作区、工程的信息。

2. 几何对象

包括 Envelope 对象、Circle 对象、Polygon 对象、Polycycle 对象、Point 对象等。几何对象用于反映 GIS 数据中具体地物对象的几何坐标数据, 以及返回用户利用鼠标操作获得的对象, 例如, 利用控件上的 TrackEnvelope 方法, 进行鼠标的拉框操作时, 用 Envelope 对象返回获得矩形框。几何对象还可以作为传递给控件上的属性或方法的参数, 例如, 将 TrackEnvelope 获得的 Envelope 对象传递给查询方法 QueryObjectsByEnvelope, 可以得到拉框查询的结果。

3. 其它对象

QueryResults 对象, 该对象用于组织调用空间查询方法后返回的查询结果。

(二) GeoMap 控件

GeoMap 通过控件上的属性、方法以及事件提供了 GeoStar 中的大部分功能, 下面将各个属性、方法、事件等按照具体功能进行分类。

1. 基本操作

用户通过鼠标操作获得一个几何对象(圆、矩形框、折线、多边形等)的方法。涉及 TrackCircle, TrackEnvelope, TrackPolyline, TrackPolygon 等方法, CircleTracking, CircleTracked,

EnvelopeTracking, EnvelopeTracked, PolylineTracking, PolylineTracked, PolygonTracking, PolygonTracked 等事件。

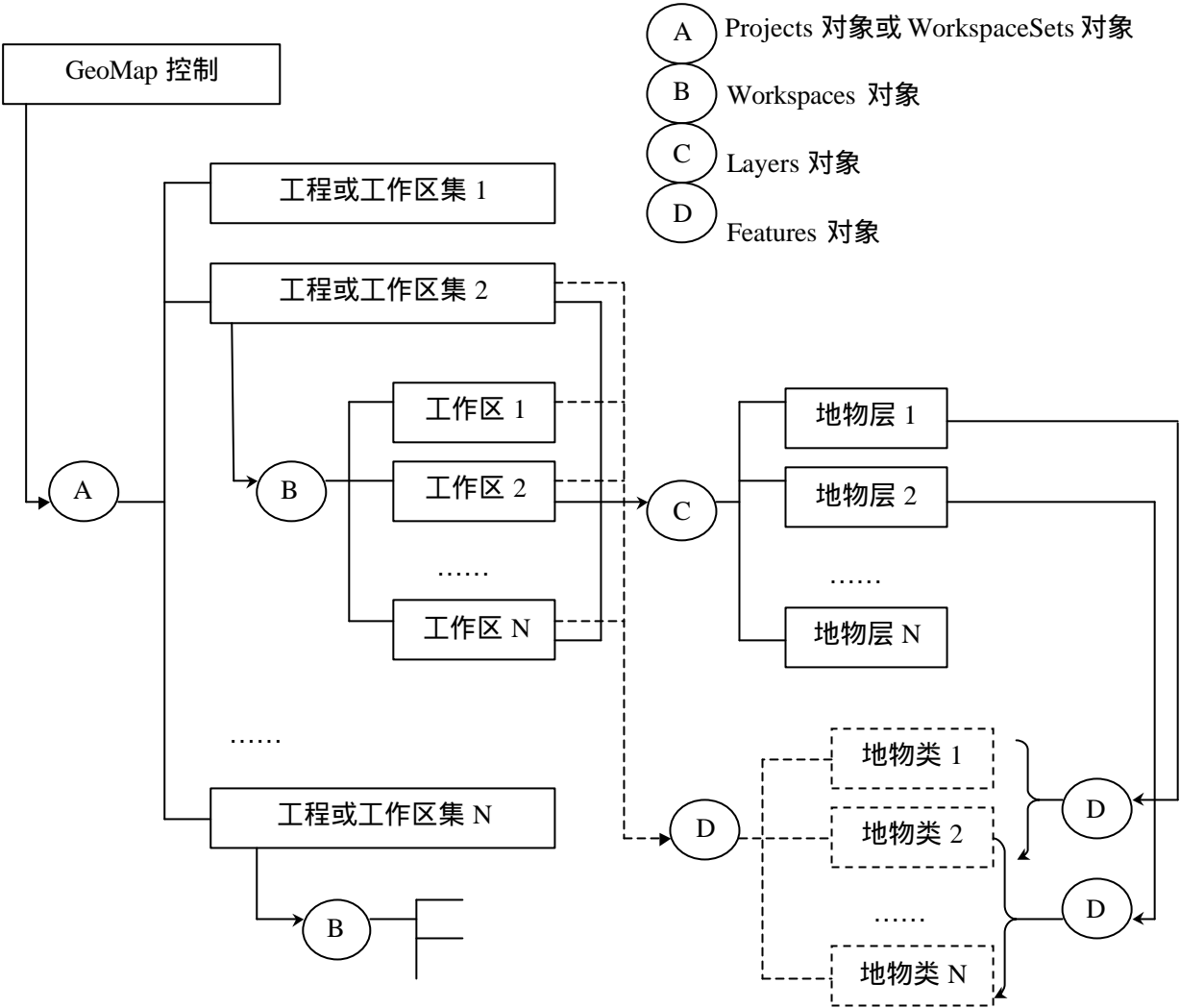


图 3-1 GeoMap 数据组织结构

2 . 数据的组织与维护

系统中工作区、工程的创建、打开、关闭，工作区的提交，层的维护，地物类的维护等。涉及 WorkspaceSets, Projects 等属性 ,OpenWorkspace, CloseWorkspace, OpenProject, CloseProject, CreateProject, CreateWorkspace , WorkspaceCheckIn 等方法 , 以及 AfterWorkspaceOpen, AfterWorkspaceClose 等事件。

3 . 可视化操作

地图数据的缩放与漫游，地图数据的分层、地物类显示，显示比例尺的控制，显示范围的控制，层与地物类显示顺序的控制，显示窗口的风格、属性的控制，地物的符号化显示、随图

放大显示、注记显示等。涉及 Appearance, BackColor, MaskEnvelope, MaskEnvelopeColor, FullEnvelope, ViewEnvelope, ViewScale, AnnotationVisible, Symbolization, Graphics Zooming 等属性, ViewCenterAt 等方法, 以及 ViewEnvelopeChange, FullEnvelopeChange 等事件。

4. 缓冲分析计算

点、折线、多边形缓冲区的计算。涉及 CalcBufferOfPoints, CalcBufferOfPolylines, CalcBufferOfPolygons 等方法。

5. 地物查询

按照点 (Pick)、线 (Cross)、面 (Contain) 的查询操作, 利用缓冲分析计算功能实现的缓冲查询操作, 对查询结果的维护, 对查询结果的突出显示, 空间对象与相应对象的属性数据之间的联系, 并利用这种联系实现空间对象与属性数据的双向查询等, 涉及 PickTolerance 属性, GetPoint, GetPolylines, GetPolygons, QueryObjectsByPoint, QueryObjectsByPolyline, QueryObjectsByPolygon, QueryObjectsByEnvelope, QueryObjectsByCircle 等方法, 以及前述基本操作中有关的属性、方法、事件等。

6. 打印输出

打印输出到一页和按照给定的比例尺分页打印两种方式, 提供打印输出前的预览功能。涉及 PrintStyle, PrintScale, PrintMarginXXX, MarginColor 等属性, DoPrint, DoPrintPreview 方法。

使用 GeoMap

(一) 举个小例子：您好，中国！

本节我们以 VB 为开发平台（WindowsNT 4.0 下的 Visual Basic 6.0），开发一个小例子，具体功能是用来打开一幅地图，说明利用 GeoMap 开发软件的简单和灵活性。

请在使用 GeoMap 之前先安装 GeoMap，并注册许可证。阅读以后的章节，需要 VB 的基础知识，限于篇幅就不多介绍 VB 的内容。下面一步一步介绍如何完成此项功能。

第一步：将 GeoMap 控件加入 V B 的工具箱。

1、选择【工程部件】菜单项(如图 4-1-1 所示)：

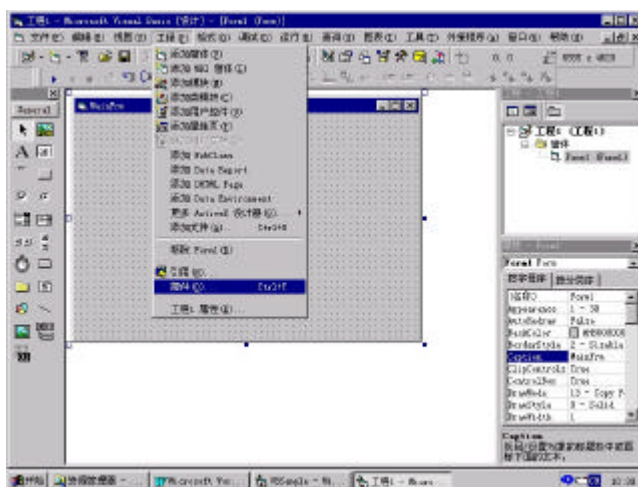


图 4-1-1

2、在弹出的“部件”对话框中选中 Geomap 控件 (如图 4-1-2 所示)：

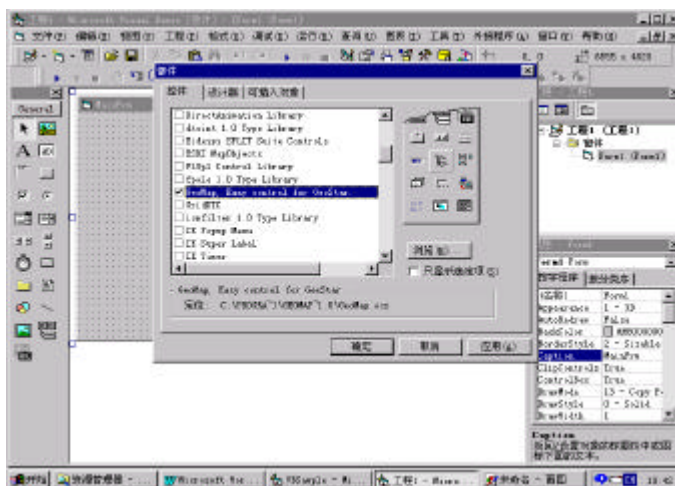


图 4-1-2

3、点击“确定”，Geomap 控件的就加入了 VB 工具箱。

第二步：将 Geomap 控件加入示例的 Form 中。(如图 4-1-3 所示)

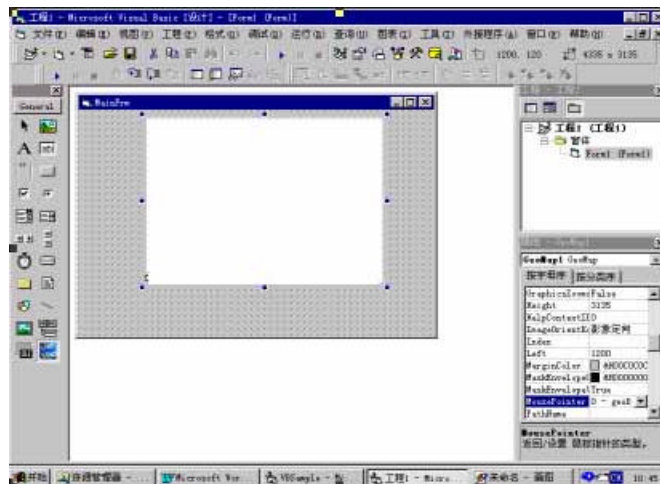


图 4-1-3

第三步：在 Form 上添加 “打开”、“关闭” 按钮，在按钮的 Click 事件代码中写入如下代码。

```
Option Explicit
Dim PathName As String    '保存文件的路径信息

Private Sub BtnOpen_Click()    '打开工作区

    On Error GoTo endofsub
    Me.OpenDlg.ShowOpen
    PathName = OpenDlg.FileName
    Map.OpenWorkspace PathName

endofsub:
    Exit sub
End Sub

Private Sub BtnClose_Click()    ' 关闭

    If PathName = "" Then Exit Sub
    Map.CloseWorkspace PathName
    PathName = ""
End Sub
```

这样就完成了设计工作。让我们来看一下程序运行的效果。
运行示例 Form，单击“打开”按钮，出现如图 4-1-4 对话框：

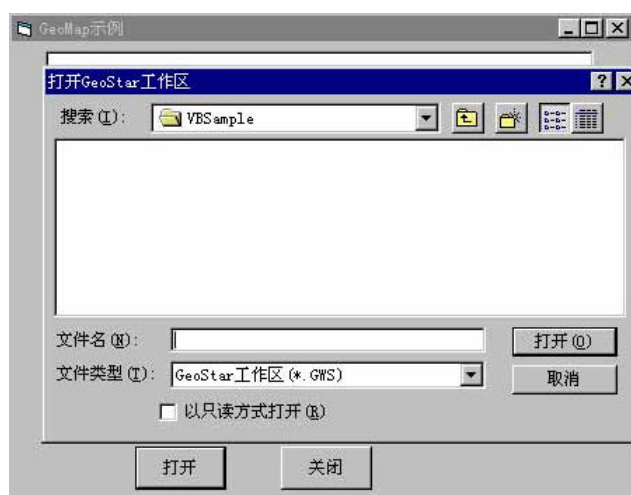


图 4-1-4

选中您要打开的中国地图，单击“打开”按钮，一幅中国地图就显示出来了（如图 4-1-5 所示）。



图 4-1-5

从上面的小例子可以看到，几行代码就可以打开一个地图窗口。有了 Geomap 这个易学、便利的 GIS 开发工具，可以实现对地图的显示、查询、分析等，满足您 GIS 工程开发多样、复杂的需求。

（二）Geomap 的文件管理

在 Geomap 中，数据组织是以工作区、工程进行分层管理的。一般地，一幅地图我们习惯地将它作为一个工作区来管理，多幅地图可以分作多个工作区来管理，也可以建立工程来管理。具体内容请参阅 GeoStar 手册。

在 Geomap 中，我们还引入一种新的文件——Geomap 文档，它是保存上次打开工作区、工程的状态，如：比例尺、地图窗口的中心等，用来快速显示需要多次使用的地图窗口。

下面介绍 GeoMap 的文件管理设计：

第一步：在示例 Form 中建立如下菜单，如图 4-2-1 所示：

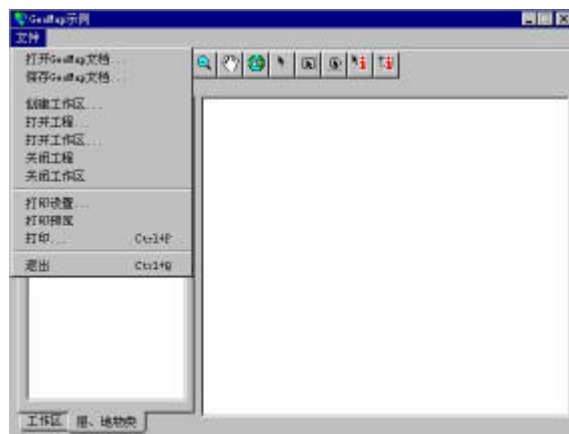


图 4-2-1

在示例 Form 的代码窗口写下初始化代码：

```
Public pathname As String 打开文件的路径
Dim createWorkspaceFrm As basicBaseCreateForm ' basicBaseCreateForm 为创建
工程或工作区的窗体
Private WithEvents printsetFrm As PrintPageSetupFrm 'PrintPageSetupFrm 为设
置打印参数的窗体
```

第二步：在【文件|打开 GeoMap 文档】菜单中写入如下代码；

```
Private Sub mFileOpen_Click()
```

```
'打开 GeoMap 文档
```

```
Dialog1.DialogTitle = "打开 GeoMap"
```

```
Dialog1.Flags = cdlOFNHideReadOnly
```

```
Dialog1.Filter = "GepMap(*.GMP)|*.GMP|(*.*)|*.*"
```

```
Dialog1.ShowOpen
```

```
PathName = Dialog1.FileName
```

```
Map.Open PathName
```

```
End Sub
```

运行示例 Form，单击【文件|打开 GeoMap 文档】菜单项，系统打开上次打开的地图文件，如图 4-2-2 所示：

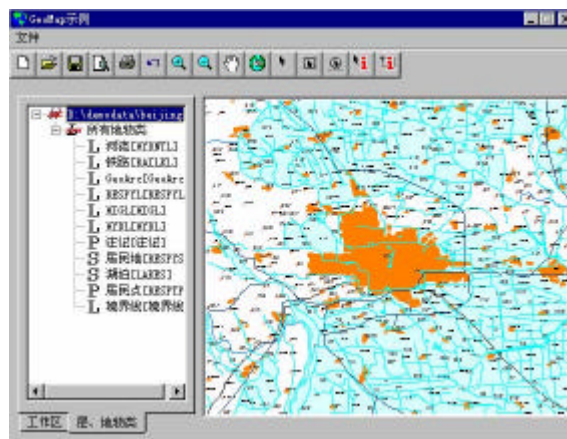


图 4-2-2

第三步：在【文件|保存 GeoMap 文档】菜单中写入如下代码；

```
Private Sub mFilesave_Click()  
    '保存 GeoMap  
    Map.Save  
End Sub
```

运行示例 Form，单击【文件|保存 GeoMap 文档】菜单项，系统保存当前打开的地图文件的状态信息。以*.gmp 文件保存。

第四步：在【文件|建立工作区】菜单中写入如下代码：

```
Dim createWorkspaceFrm As basicBaseCreateForm  
' basicBaseCreateForm 为用于创建工程或工作区的参数信息的窗体
```

```
Private Sub mFilereate_Click() '新建工作区  
    Set createWorkspaceFrm = New basicBaseCreateForm  
    createWorkspaceFrm.Caption = "新建工作区参数"  
    createWorkspaceFrm.createType = 0 '工作区  
    createWorkspaceFrm.Show vbModal  
    If createWorkspaceFrm.bOkPressed Then  
        Map.OpenWorkspace createWorkspaceFrm.ProjectPathName  
    End If
```

'创建工程调用 Map.CreateProject CreateInfo 方法，创建工作区需调用 Map.CreateWorkspace CreateInfo 方法。其中 CreateInfo 为创建工程或工作区参数信息的对象，包括比例尺、坐标单位类型、投影参数、XYZ 方向的坐标精度、坐标原点坐标等，可参考示例 Geosample 加以声明。

```
End Sub
```

运行示例 Form，单击【文件|建立工作区】菜单项，系统就可以建立工作区，如图 4-2-3 所示：



图 4-2-3

第五步：在【文件|打开工程】菜单中写入如下代码；

Private Sub mFileGeoPrjOpen_Click() '打开工程

```
Dialog1.DialogTitle = "打开工程"
Dialog1.Flags = cdIOFNHideReadOnly
Dialog1.Filter = "吉奥工程(*.gpj)|*.gpj"
Dialog1.ShowOpen
PathName = Dialog1.FileName
Map.OpenProject PathName
'此处可编写鼠标拉框选择打开工程中的工作区代码
End Sub
```

运行示例 Form，单击【文件|打开工程】菜单项，系统就可以打开工程文件(*.gpj)，如图 4-2-4 所示

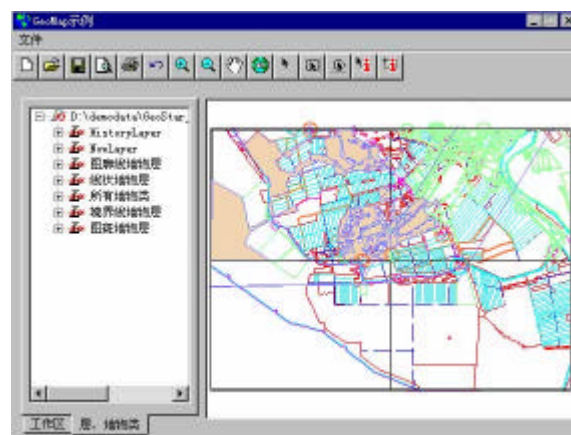


图 4-2-4

第六步：在【文件|关闭工程】菜单中写入如下代码：

Private Sub mFileGeoPrjClose_Click() '关闭工程

```
If PathName = "" Then Exit Sub
```

```

Map.CloseProject PathName
PathName = ""
End Sub

```

运行示例 Form，单击【文件|关闭工程】菜单项，系统就可以关闭工程文件(*.gpj)，如图 4-2-5 所示

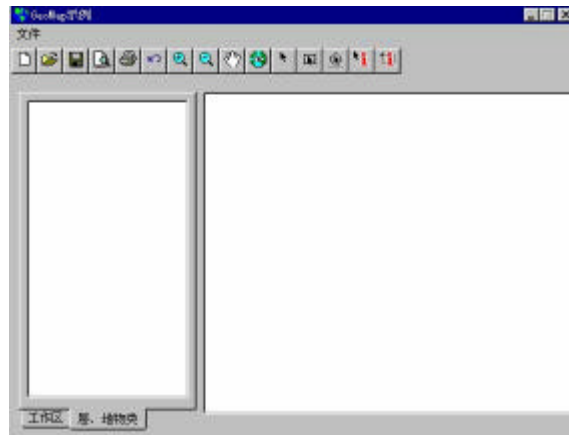


图 4-2-5

有关打开和关闭工作区的代码参考上节的内容。

接下来介绍 GeoMap 的图形打印输出设计。

图形的打印输出

Geomap 的打印输出包括打印输出到一页和按照给定的比例尺分页打印两种方式，提供打印输出前的预览功能。

第一步：在【文件|打印设置】菜单中写入如下代码；

```

Private Sub mnuPrintSetting_Click()
    '打印设置
    printsetFrm.Show
End Sub

```

运行示例 Form，单击【文件|打印设置】菜单项，系统就弹出“打印设置”对话框，如图 4-2-6 所示，输入正确的打印参数，单击【确定】按钮即可。

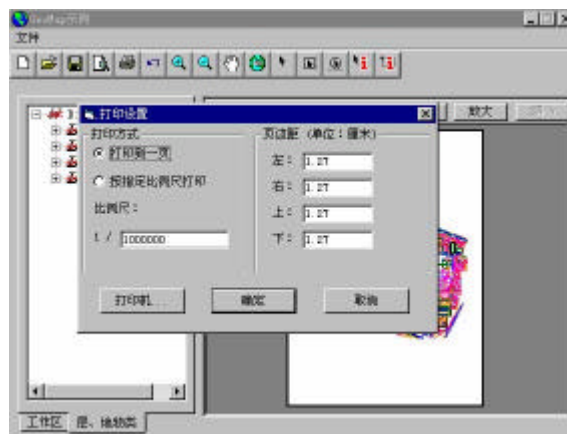


图 4-2-6

第二步：在【文件|打印预览】菜单中写入如下代码；

```
Private Sub mnuPrintPreview_Click()  
    '打印预览  
    Map.DoPrintPreview  
  
End Sub
```

运行示例 Form，单击【文件|打印预览】菜单项，系统就弹出打印预览窗口，如图 4-2-7 所示。

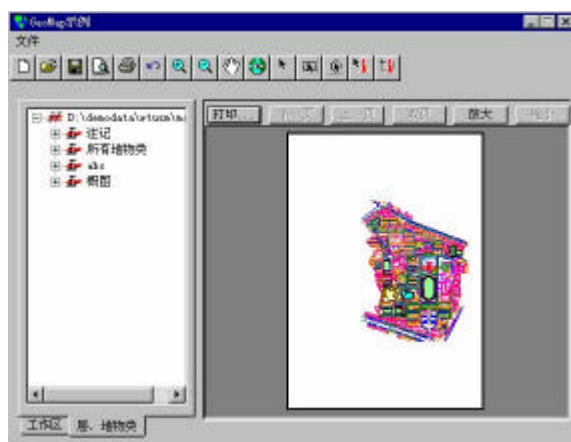


图 4-2-7

第三步：在【文件|打印】菜单中写入如下代码；

```
Private Sub mnuPrint_Click()  
    '打印  
    Map.DoPrint  
  
End Sub
```

运行示例 Form，单击【文件|打印】菜单项，系统就弹出打印预览窗口，如图 4-2-8 所示。

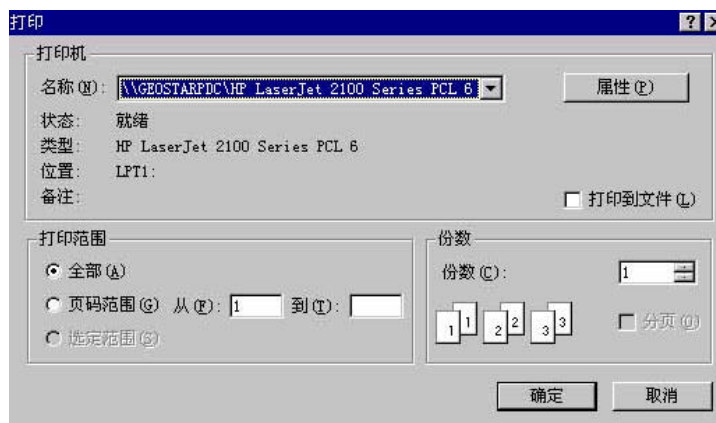


图 4-2-8

本节我们学习了 GeoMap 的文件管理设计，下一节我们来学习视图设计。

（三）Geomap 的视图操作

Geomap 控件的视图操作包括放大、缩小、漫游、注记显示、随图放大、地图符号化等功能，现在介绍如下：

下面介绍 GeoMap 的视图操作设计：

第一步：在示例 Form 中建立如下菜单，如图 4-3-1 所示：

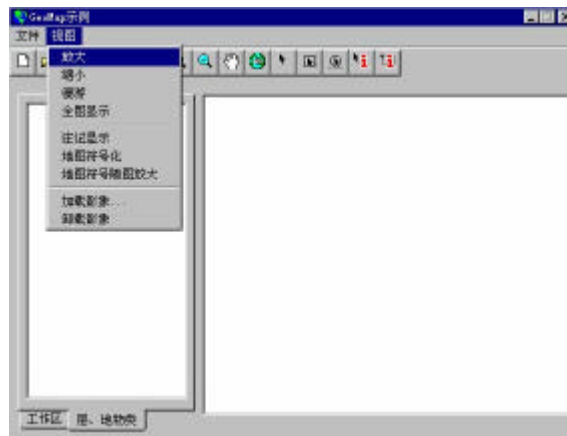


图 4-3-1

第二步：在【视图|放大】菜单中写入如下代码；

```
Private Sub mnuZoomIn_Click()  
    '鼠标变为放大时的形状  
    Map.MousePointer = geoZoomIn  
End Sub
```

第二步：在【视图|缩小】菜单中写入如下代码；

```
Private Sub mnuZoomOut_Click()  
    '鼠标变为缩小时的形状  
    Map.MousePointer = geoZoomOut  
    map.ViewScale = map.ViewScale * 2  
  
End Sub
```

第三步：在【视图|漫游】菜单中写入如下代码；

```
Private Sub mnuGrabber_Click()  
    '鼠标变为漫游时的形状  
    Map.MousePointer = geoZoomPan  
End Sub
```

当您在 GeoMap 控件窗口中按下鼠标按键时，示例系统将您所指定的当前操作类型和鼠标坐标送入 GeoMap 控件的 geoMouseDown、geoMouseUp 消息处理函数中，通过调用 GeoMap 控件上的相应方法，您将可以实现放大、缩小、漫游等功能。

```
Private Sub Map_geoMouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal X
```

```

As Long, ByVal Y As Long)
    If Map.TrackAction <> geoTrackNop Then Exit Sub
    If Button <> 1 Then Exit Sub
    Select Case Map.MousePointer
        Case geoZoomIn
            Map.TrackEnvelope
        Case geoZoomOut
            Map.viewScale = Map.viewScale * 2
        Case geoZoomPan
            Map.DragViewEnvelope
    End Select
End Sub

Private Sub Map_EnvelopeTracked(ByVal Envelope As GeoMap.Envelope)
    Select Case Map.MousePointer
        Case geoZoomIn
            '判断鼠标拉框时
            If Envelope.Width = 0 Or Envelope.Height = 0 Then
                Map.viewScale = Map.viewScale / 2
            Else
                Map.ViewEnvelope = Envelope
            End If
        Case geoZoomOut

    End Select
End Sub

```

运行示例 Form，单击【视图|放大】菜单项，在 Geomap 控件窗口中用鼠标拉个矩形框，系统放大地图，如图 4-3-2、图 4-3-3 所示。

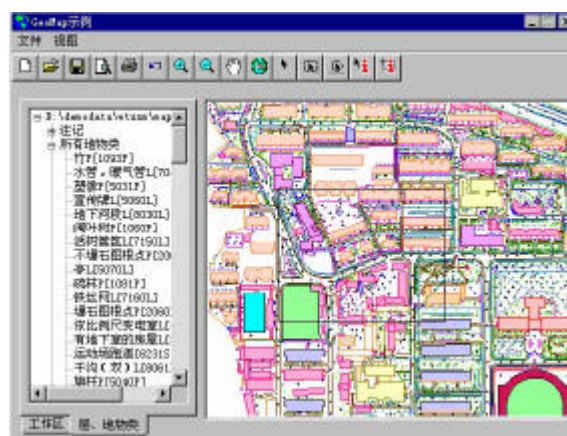


图 4-3-2

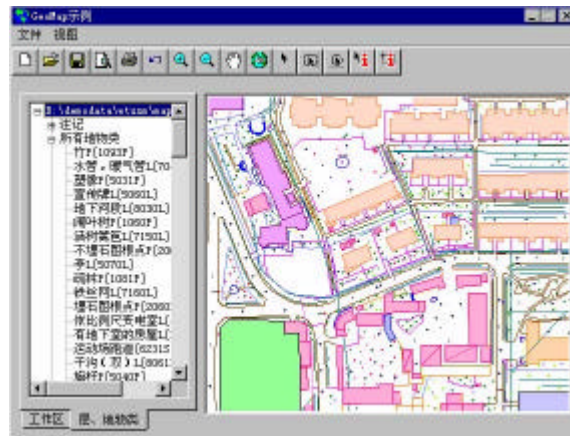


图 4-3-3

单击【视图|缩小】菜单项，在 Geomap 控件窗口单击鼠标，如图 4-3-4 所示。

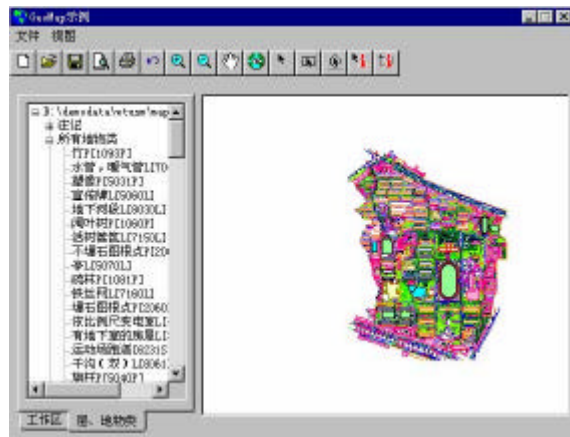


图 4-3-4

单击【视图漫游】菜单项，在 Geomap 控件窗口按下鼠标，拖动鼠标如图 4-3-5 所示。

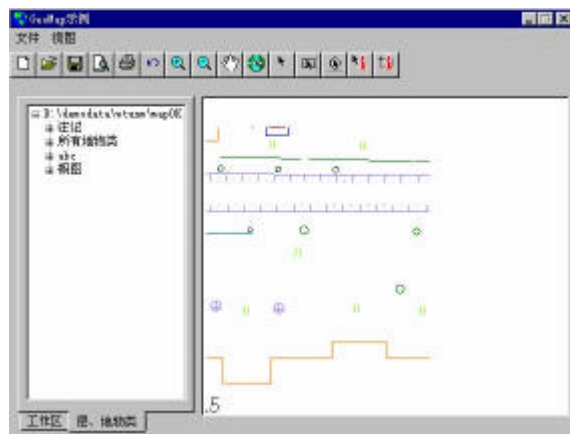


图 4-3-5

图 4-3-7

第五步：在【视图|地图符号化】菜单中写入如下代码；

```
Private Sub mnuSymbol_Click() '地图符号化

    Map.Symbolization = If(Map.Symbolization, False, True)

End Sub
```

运行示例 Form，单击【视图|符号化】菜单项，在 Geomap 控件窗口地图要素符号显示出来，如图 4-3-7 所示。

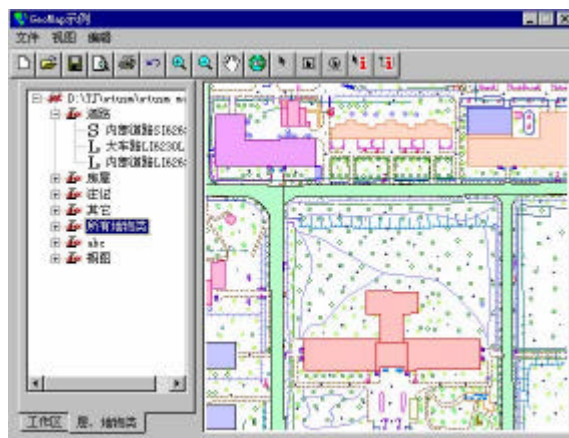


图 4-3-8

影像叠加

初始化代码：

```
Dim mbImageSettingsChanged As Boolean
Dim gImageOrientKey As String
```

在【视图|加载影像】菜单中写入如下代码：

```
Private Sub mnuImageOn_Click()
    Dialog1.DialogTitle = "选择需要加载的影像文件"
    Dialog1.Filter = "影像文件 (*.bmp)|*.bmp"
    Dialog1.DefaultExt = "*.bmp"
    Dialog1.FileName = "*.bmp"
    Dialog1.CancelError = True
    On Error GoTo cancelPressed
    Dialog1.ShowOpen
    On Error GoTo 0

    Dim paramfilename As String
    Dim pos As Long
    paramfilename = Dialog1.FileName
```

```

gImageOrientKey = "单幅影像"

pos = ReverseInStr(1, paramfilename, ".bmp", vbTextCompare)
paramfilename = Left(paramfilename, pos - 1) + ".dom"
Map.ImageOrientRemove gImageOrientKey
If Me.Map.ImageOrientLoad(Me.Dialog1.FileName, paramfilename, gImageOrientKey) =
False Then
    Map.ImageOrientNew Me.Dialog1.FileName, gImageOrientKey
    MsgBox "请在 GeoStar 中对选定的影像文件进行定向。"
Else
    mbImageSettingsChanged = False
    Map.Refresh
End If

cancelPressed:
Exit Sub
End Sub

```

运行示例 Form，单击【视图|加载影像】菜单项，系统弹出对话框。从对话框中选择需加载的影像文件。在 Geomap 控件窗口将影像作为背景显示出来，如图 4-3-8 所示。（注意：加载影像之前，必须先要对影像进行定向。）

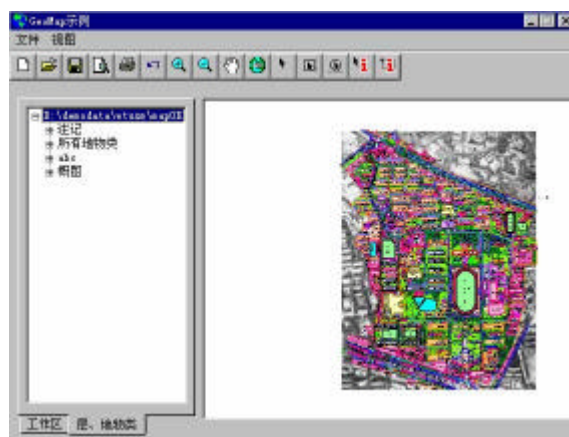


图 4-3-8

在【视图|加载影像】菜单中写入如下代码：

```

Private Sub mnuImageOff_Click()
    Map.ImageOrientRemove gImageOrientKey
    Map.Refresh

```

```

End Sub

```

运行示例 Form，单击【视图|卸载影像】菜单项，在 Geomap 控件窗口将影像从背景中去掉。

本节我们学习了 GeoMap 的视图操作设计，下一节我们来学习编辑功能设计。

（四）常用的编辑操作

人们通过地图数字化工作将图形数据输入到系统中，由于各种因素的影响，图形的质量总会有各种的问题；另外，地图数据的不断更新，都需要大量的图形编辑工作。常用的编辑操作包括：增加单点、增加线状地物对象（普通折线、矩形、三点圆等）、面状地物对象（普通多边形面状地物、矩形面状地物等）。选中需要编辑的地物；移动、删除选中的对象等。

下面我们学习 GeoMap 的常用的编辑操作设计。

第一步：在 GeoMap 示例 Form 中建立【编辑】菜单，添加 TreeView 控件，作用是提供选择当前要编辑、查询操作的地物类。如图 4-4-1 所示；



图 4-4-1

初始化代码如下：

```
Dim curworkspace As GeoMap.workspace
Dim Curfeature As GeoMap.feature
Dim clicknode As Node
Dim feaIndex As String
Dim seltype As String
Dim curprj As String

Dim feakey As String 索引树
Dim prjNode As Node
Dim prj As GeoMap.Project
```

在打开工作区或工程文件后，地物类索引树上罗列出所有的地物类，点击上面的节点，就获得当前地物类的编号，供以后调用。

‘索引树初始化

```
Private Sub map_AfterWorkspaceSetOpen(ByVal PathName As String) '工作区集打开后
    Dim wssNode As Node
    Dim lyrNode As Node
    Dim feaNode As Node
    Dim lyrCount, feaCount As Long
    Dim i, j As Long
    Dim lyrkey As String
    Dim feakey As String
```

```

Dim wss As GeoMap.WorkspaceSet
Dim lyr As GeoMap.Layer
Dim fea As GeoMap.feature

Set wss = Map.WorkspaceSets(PathName)
Set wssNode = featuretree.Nodes.Add(, , PathName, PathName, "WSS")
wssTree.Nodes.Add(, , PathName, PathName, "WSS")
wssNode.Tag = "wss"
Dim imgkey As String

    lyrCount = wss.Layers.Count
    For i = 0 To lyrCount - 1
        Set lyr = wss.Layers(i)
        lyrkey = PathName + "\" + lyr.LayerName
        Set lyrNode = featuretree.Nodes.Add(wssNode, tvwChild, lyrkey, lyr.LayerName,
"LYR1")
        feaCount = lyr.Features.Count
        For j = 0 To feaCount - 1
            Set fea = lyr.Features(j)
            feakey = lyrkey + "&" + fea.FeatureID
            Select Case fea.GeometryType
                Case geoSinglePoint, geoGroupPoint, geoAnnotation:
                    imgkey = "P1"
                Case geoLine:
                    imgkey = "L1"
                Case geoSurface, geoComplex:
                    imgkey = "S1"
            End Select
            Set feaNode = featuretree.Nodes.Add(lyrNode, tvwChild, feakey, fea.FeatureName +
"[" + fea.FeatureID + "]", imgkey)
            feaNode.Tag = "FEA"
        Next j
    Next i
    curprj = "wss"
End Sub

```

在 GeoMap 中进行编辑操作时，首先要选中要进行编辑的对象所在的地物类，利用各种选择方法选中要编辑的目标，才能编辑目标。

第二步：在 GeoMap 示例 Form【编辑|选择】菜单中写入如下所述代码；

'进行下面的各种选择之前必须选择当前地物类（空间目标所在的地物类）

Private Sub mnuEditSelpoi_Click() '点选择

Map.MousePointer = geoCROSSBOX

seltype = "point"

End Sub

Private Sub mnuEditSelline_Click() '线选择

Map.MousePointer = geoCROSSBOX

seltype = "line"

End Sub

Private Sub mnuEditSelcycle_Click() '圆选择

```

Map.MousePointer = geoCROSSBOX
seltype = "cycle"
End Sub

Private Sub mnuEditSelEnv_Click() '矩形选择
Map.MousePointer = geoCROSSBOX
seltype = "envelope"
End Sub

Private Sub mnuEditSelpolygon_Click() '多边形选择
Map.MousePointer = geoCross
seltype = "polygon"
End Sub

```

当您在 GeoMap 控件窗口中按下鼠标按键时，示例系统将您所指定的当前操作类型和鼠标坐标送入 GeoMap 控件的 geoMouseDown、geoMouseUp 消息处理函数中，通过调用 GeoMap 控件上的 TrackPolyline 方法、TrackCircle 方法、TrackPolygon 方法及 TrackEnvelope 方法，可以获得矩形框、圆、多边形、折线对象；然后，利用上一步获得的几何对象（Point、Envelope、Circle、Polygon、Polyline）分别作为 EditSelectObjectsByPoint 方法、EditSelectObjectsByEnvelope 方法、EditSelectObjectsByCircle 方法、EditSelectObjectsByPolygon 方法、EditSelectObjectsByPolyline 方法的参数，您可以选中指定地物类中的满足条件的地物对象。



图 4-4-2



图 4-4-3

第三步：在 GeoMap 示例 Form 【编辑|增加点】菜单中写入如下所述代码；

```
Private Sub mnuEditAddpoi_Click() '增加单点
    If SelCur(curworkspace, Curfeature) Then
        Map.EditTrackSinglePoint curworkspace, Curfeature
    End If
End Sub
```

```
Private Sub mEditAddpoigroup_Click() '增加点群

    If SelCur(curworkspace, Curfeature) Then
        Map.EditTrackGroupPoint curworkspace, Curfeature
    End If
End Sub
```

运行示例 Form，单击【编辑|增加点】菜单项，在 Geomap 控件窗口用鼠标点击的方法将线状的地物对象添加到地图窗口。

第四步：在 GeoMap 示例 Form 【编辑|增加线】菜单中写入如下所述代码；

```
Private Sub mnuEditAddlinepolyline_Click() 增加折线

    If SelCur(curworkspace, Curfeature) Then
        Map.EditTrackGeoLine curworkspace, Curfeature, geoLTCommon
    End If
End Sub
```

```
Private Sub mnuEditAddlineEnvelope_Click() 增加矩形

    If SelCur(curworkspace, Curfeature) Then
        Map.EditTrackGeoLine curworkspace, Curfeature, geoLTRectangle
    End If
End Sub
```

```
Private Sub mnuEditAddline3poicycle_Click() 增加三点圆
```

```

If SelCur(curworkspace, Curfeature) Then
    Map.EditTrackGeoLine curworkspace, Curfeature, geoLT3PointCircle
End If
End Sub

```

运行示例 Form，单击【编辑|增加线】菜单项，在 Geomap 控件窗口用鼠标点击的方法将线状的地物对象添加到地图窗口。如图 4-4-4 所示

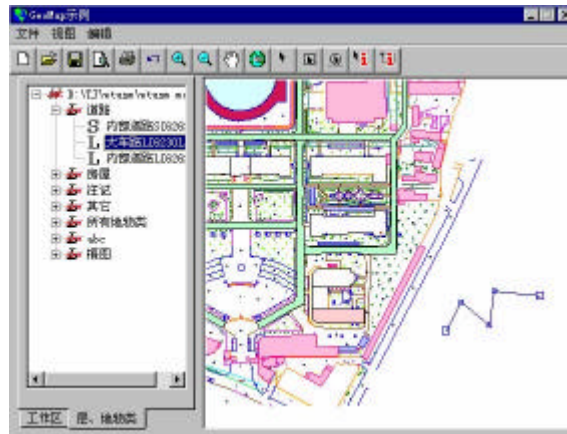


图 4-4-4

第五步：在 GeoMap 示例 Form 【编辑|增加面】菜单中写入如下所述代码；

```

Private Sub mnuEditAddfurEnvelope_Click() '增加矩形面

    If Not SelCur(curworkspace, Curfeature) Then Exit Sub
    Map.EditTrackGeoSurface curworkspace, Curfeature, geoSTRectangle
End Sub

Private Sub mnuEditAddfursingfur_Click() '增加普通多边形面

    If SelCur(curworkspace, Curfeature) Then
        Map.EditTrackGeoSurface curworkspace, Curfeature, geoSTCommon
    End If
End Sub

```

运行示例 Form，单击【编辑|增加面】菜单项，在 Geomap 控件窗口用鼠标点击的方法将面状的地物对象添加到地图窗口。如图 4-4-5 所示。

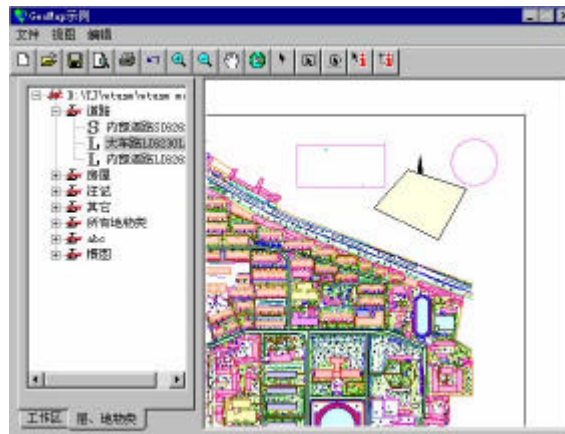


图 4-4-5

增加地物，首先调用函数 SelCur(workspace, feature)获取当前地物类，增加点状地物对象，调用 GeoMap.EditTrackSinglePoint 方法；增加线状地物对象，调用 GeoMap.EditTrackGeoLine 方法；增加面状地物对象，调用 GeoMap.EditTrackGeoSurface 方法。

```
Function SelCur(workspace As GeoMap.workspace, feature As GeoMap.feature) As Boolean
```

```
    Dim fealength As Long
    Dim wss As WorkspaceSet
    Dim prj As Project
    Dim feakeylen As Long
    Dim pos As Long
    Dim ms As String
    Dim ddd As Node
```

```
    Map.MousePointer = geoCross
    feakeylen = Len(feakey)
    pos = InStr(1, feakey, "&")
    feaIndex = Mid(feakey, pos + 1, feakeylen - pos)
    If Len(feaIndex) = 0 Then Exit Function
```

```
    If curprj = "wss" Then
        Set wss = Map.WorkspaceSets(0)
        Set curworkspace = wss.Workspaces(0)
        Set Curfeature = wss.Features(feaIndex)
    Else
        Set prj = Map.Projects(0)
        Set curworkspace = prj.Workspaces(0)
        Set Curfeature = prj.Features(feaIndex)
    End If
    fealength = Len(Curfeature)
    If fealength = 0 Then Exit Function
    SelCur = True
```

```
End Function
```

第六步：在 GeoMap 示例 Form【编辑|移动】菜单中写入如下所述代码；

```
Private Sub mnuMove_Click() 移动选中的目标
    Map.EditDragSelectedObjsForMove
```

End Sub

运行示例 Form，在选中了需要编辑的对象后，单击【编辑|移动】菜单项，在 Geomap 控件窗口用鼠标拖动的方法将选中的地物对象移动位置。如图 4-4-6 所示。

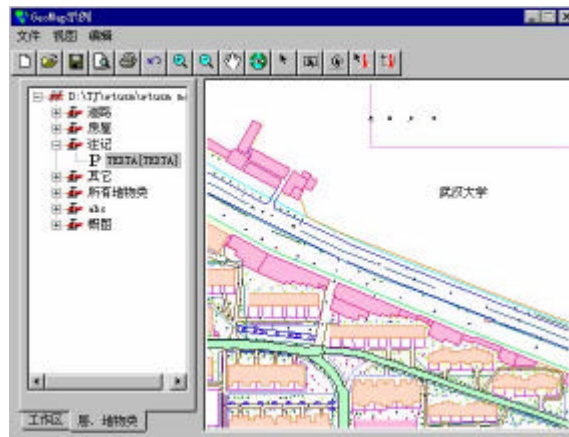


图 4-4-6

第七步：在 GeoMap 示例 Form **【编辑|删除】** 菜单中写入如下所述代码；

Private Sub mnuDel_Click() 删除选中的目标

```
Map.EditSelectObjectsDelete
End Sub
```

运行示例 Form，在选中了需要编辑的对象后，单击【编辑|删除】菜单项，在 Geomap 控件窗口将选中的地物对象删除。如图 4-4-7 所示。

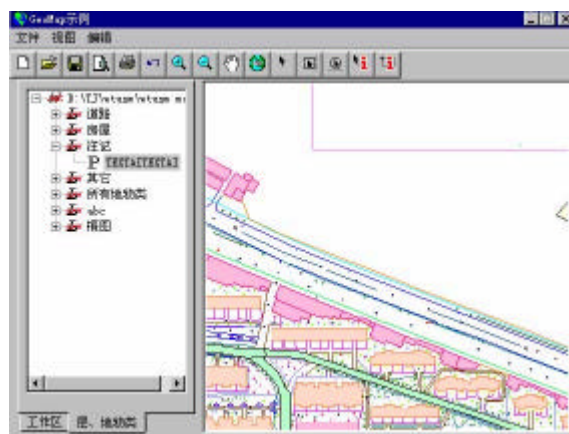


图 4-4-7

本节学习了利用 Geomap 控件实现常用的一些编辑功能。接下来，我们学习 Geomap 常用的查询功能设计。

（五）专题图

专题图以简明、完备的形式表达一种或几种相关联的社会、经济、自然现象的统计数据，使统计数据的内涵可视化，易于分析和获取信息。它不仅可以表示现象的发展动态变化和发展规律。专题图也是 GIS 的重要组成部分，既可以直接表示空间数据的特性，同时也可以将空间查询或空间分析的结果以适当的图形方式表示出来，这将比数据、表格的表达更加简单、明了。

下面我们学习 GeoMap 的常用的专题图功能设计.本节内容包括创建、打开、修改专题图等。

第一步：在示例 Form 中建立如下菜单，如图 4-5-1 所示：

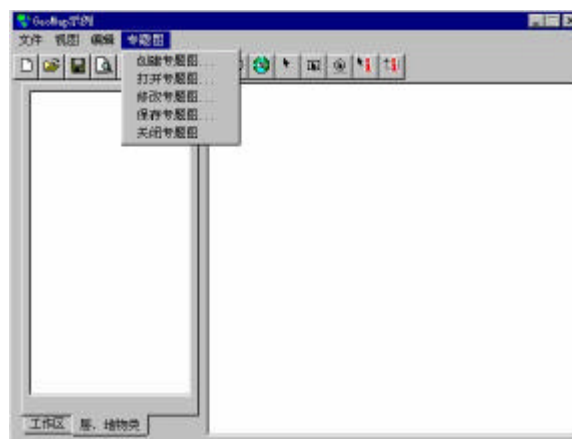


图 4-5-1

初始化代码：

Dim WithEvents geoengine As GeoMap.geoengine '用于处理外部属性数据源

Private Sub Form_Load()

...

Set geoengine = GeoMap.geoengine

...

End Sub

第二步：在 GeoMap 示例 Form 【专题图|生成专题图】菜单中写入如下所述代码；

Private Sub mnuThemeCreate_Click() 生成专题图

Dim wss As GeoMap.WorkspaceSet

Dim prj As GeoMap.Project

Dim ws As GeoMap.Workspace

Dim fea As GeoMap.Feature

Dim isWSS As Boolean

If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Sub

‘ 此函数（自己编写）作用在于获取用来生成专题图的地物类，必须是面状地物类。

If ws Is Nothing Then Exit Sub

Map.ThematicMapCreate ws, fea 调用 Create 函数开始创建

End Sub

' 调用 Map.ThematicMapCreate 方法，将产生 geoengine_QueryDataSourceForFeature 事件。在该事件中加入指明属性表信息存放的路径。

```
Private Sub geoengine_QueryDataSourceForFeature(ByVal Workspace As
GeoMap.Workspace, ByVal Feature As GeoMap.Feature, connectString As String, tableName As
String)
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim dsn As String

    Select Case Workspace.OwnerType
    Case geoWorkspaceSet
        Set wss = Me.Map.WorkspaceSets(Workspace.OwnerPathName)
        dsn = Workspace.WSDirectory & "WORKATTR"
        dsn = ConvertString(dsn)
        dsn = "ODBC;DSN=" & dsn & ";UID=;PWD=;SourceDB=" &
Workspace.WSDirectory & "WORKATTR;Driver=MicroSoft Visual FoxPro
Driver;SourceType=DBF"
    Case geoProject
        Set prj = Me.Map.Projects(Workspace.OwnerPathName)
        dsn = prj.PRJDirectory & "PRJATTR"
        dsn = ConvertString(dsn)
        dsn = "ODBC;DSN=" & dsn & ";UID=;PWD=;SourceDB=" & prj.PRJDirectory &
"PRJATTR\" & prj.PRJName & ".DBC;Driver=MicroSoft Visual FoxPro
Driver;SourceType=DBC"
    End Select

    connectString = dsn
    tableName = Feature.DBTableName

End Sub
```

运行示例 Form，先在地物索引树上选取用来生成专题图的面状地物类，（其应有属性信息。）单击【专题图|生成专题图】菜单项，弹出“制图符号类”对话框如图 4-5-2 所示，单击“下一步”按钮，在制作向导的帮助下生成专题图（图 4-5-3）。

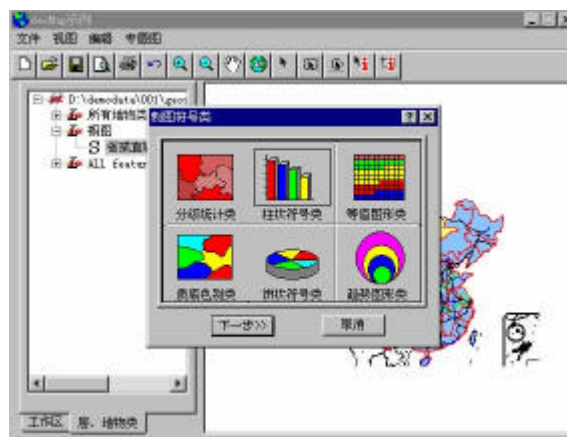


图 4-5-2

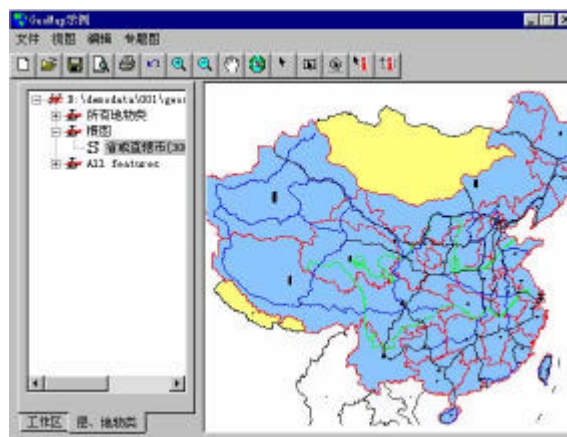


图 4-5-3

第三步：在 GeoMap 示例 Form【专题图|打开专题图】菜单中写入如下所述代码；

```
Private Sub mnuThemeOpen_Click()
    '打开的专题图

    Dialog1.DialogTitle = "选择需要打开的专题图"
    Dialog1.Filter = "专题图文件 (*.them)|*.them"
    Dialog1.DefaultExt = "*.them"
    Dialog1.FileName = "*.them"
    Dialog1.CancelError = True
    On Error GoTo cancelPressed
    Dialog1.ShowOpen
    PathName = Dialog1.FileName

    Map.ThematicMapOpen PathName

cancelPressed:
    Exit Sub
End Sub
```

运行示例 Form，单击【专题图|打开专题图】菜单项，弹出“选择打开的专题图”对话框如图 4-5-4 所示，选择专题图，单击“打开”按钮。

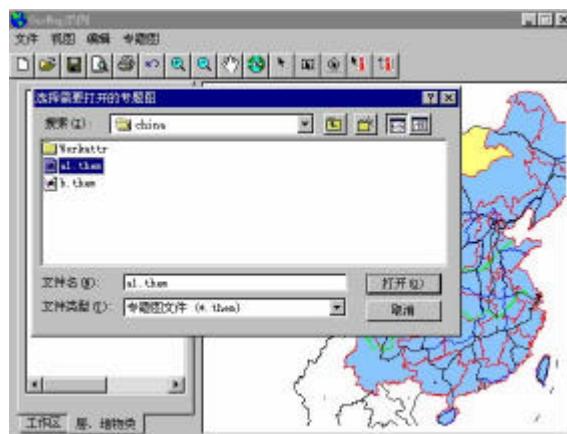


图 4-5-4

第四步：在 GeoMap 示例 Form【专题图|修改专题图】菜单中写入如下所述代码；

```
Private Sub mnuThemeEdit_Click()  
  
    '修改专题图  
    Map.ThematicMapModify  
    Map.Refresh  
End Sub
```

运行示例 Form，单击【专题图|修改专题图】菜单项，弹出“选择修改项目”对话框，如图 4-5-5、图 4-5-6 所示，选择专题图，单击“打开”按钮。

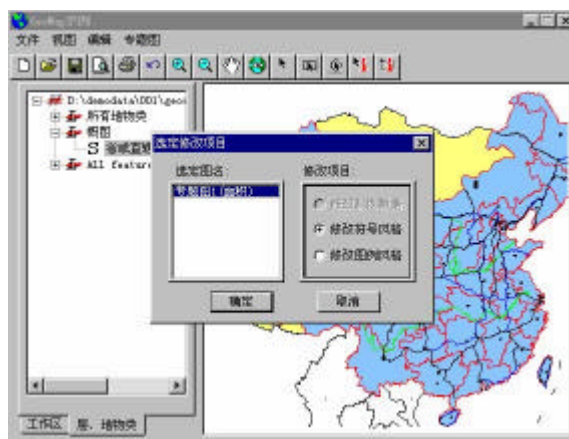


图 4-5-5



图 4-5-6

本节学习了利用 Geomap 控件实现常用的一些专题制图功能，可以用来分析有关的 GIS 信息，辅助决策。

(五) 查询操作

GeoMap 控件的查询操作，包括按照点（Pick）、线（Cross）、面（Contain）的查询操作，利用缓冲分析计算功能实现的缓冲查询操作，对查询结果的维护，对查询结果的突出显示，空间对象与相应对象的属性数据数据之间的联系的维护，并利用这种联系实现空间对象与属性数据的双向查询等。

下面我们学习 GeoMap 的常用的查询功能的设计。

第一步：在示例 Form 中建立如下菜单，如图 4-6-1 所示：

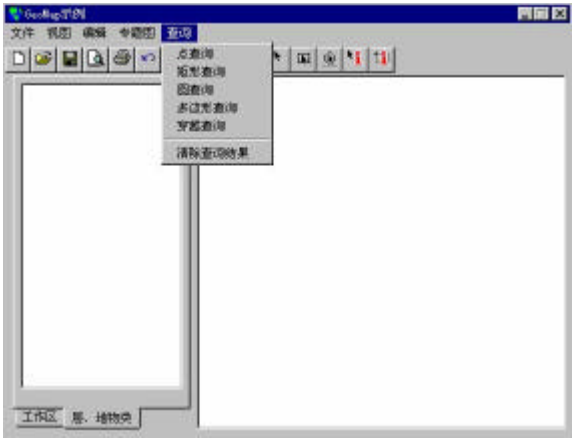


图 4-6-1

第二步：在 GeoMap 示例 Form 【查询|点查询】菜单中写入如下所述代码；

```
Private Sub mnuPointQuery_Click()  
  
    Map.MousePointer = geoCross 点查询时鼠标形状变为十字丝  
    QueryType = "PointQry"  
End Sub
```

当您在 GeoMap 控件窗口中按下鼠标按键时，示例系统将您所指定的当前操作类型和鼠标坐标送入 GeoMap 控件的 geoMouseDown、geoMouseUp 消息处理函数中，通过调用 GeoMap 控件上的 QueryObjectsByPoint 方法，可以获得点查询的结果。

运行示例 Form，单击【查询|点查询】菜单项，在 Geomap 控件窗口用鼠标点击某一位置，查询的结果显示如图 4-6-2 所示



图 4-6-2

第三步：在 GeoMap 示例 Form 【查询|矩形查询】菜单中写入如下所述代码；

```
Private Sub mnuPolygonQuery_Click()  
    '矩形查询时鼠标形状变为十字丝  
    Map.MousePointer = geoCross  
    QueryType = "PolygonQry"  
    Map.TrackPolygon
```

```
End Sub
```

当您在 GeoMap 控件窗口中按下鼠标按键时，示例系统将您所指定的当前操作类型和鼠标坐标送入 GeoMap 控件的 geoMouseDown、geoMouseUp 消息处理函数中，通过调用 GeoMap 控件上的 QueryObjectsByEnvelope 方法、QueryObjectsByCircle 方法或 QueryObjectsByPolygon 方法，可以获得各种查询查询的结果。

运行示例 Form，单击【查询|矩形查询】菜单项，在 Geomap 控件窗口用鼠标点击获得一个矩形，查询的结果显示如图 4-6-3、图 4-6-4 所示

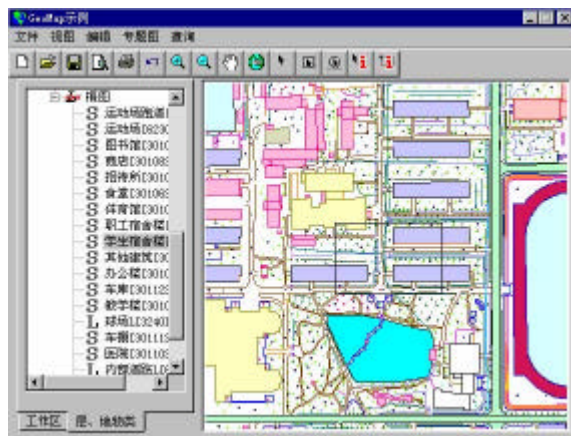


图 4-6-3

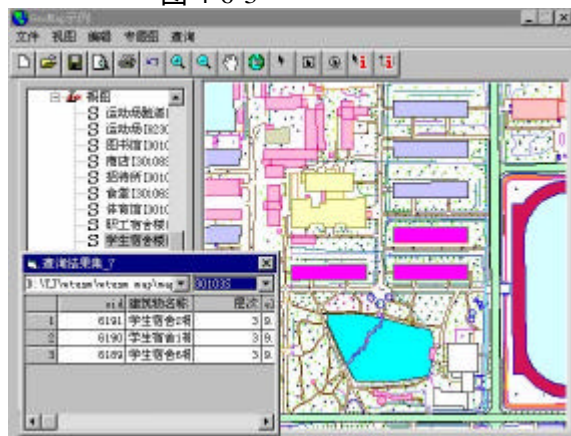


图 4-6-4

第四步：在 GeoMap 示例 Form 【查询|多边形查询】菜单中写入如下所述代码；

```
Private Sub mnuPolygonQuery_Click()  
    '矩形查询时鼠标形状变为十字丝  
    Map.MousePointer = geoCross  
    QueryType = "PolygonQry"  
    Map.TrackPolygon  
  
End Sub
```

运行示例 Form，单击【查询|多边形查询】菜单项，在 Geomap 控件窗口用鼠标点击获得一个多边形，查询的结果显示如图 4-6-5、图 4-6-6 所示

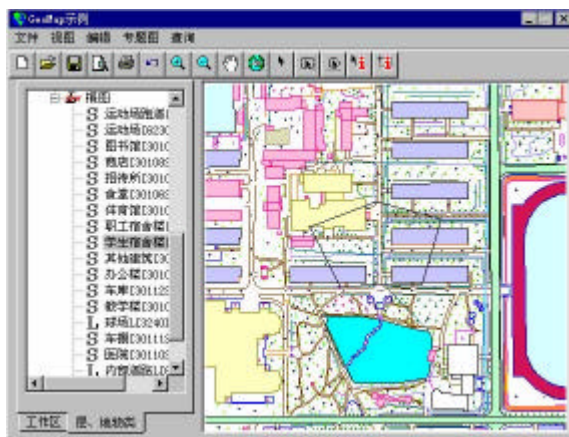


图 4-6-5

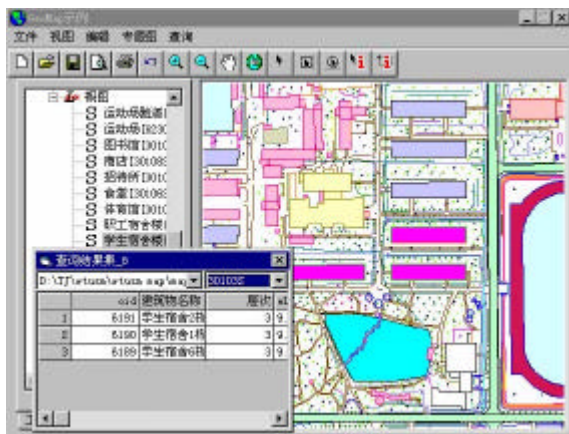


图 4-6-6

第五步：在 GeoMap 示例 Form 【查询|圆查询】菜单中写入如下所述代码；

```
Private Sub mnuCircleQuery_Click()  
  
    '圆查询时鼠标形状变为十字丝
```

```
Map.MousePointer = geoCross
QueryType = "CircleQry"
```

End Sub

运行示例 Form，单击【查询|圆查询】菜单项，在 Geomap 控件窗口用鼠标单击，获得一个圆，查询的结果如图 4-6-7、图 4-6-8 所示

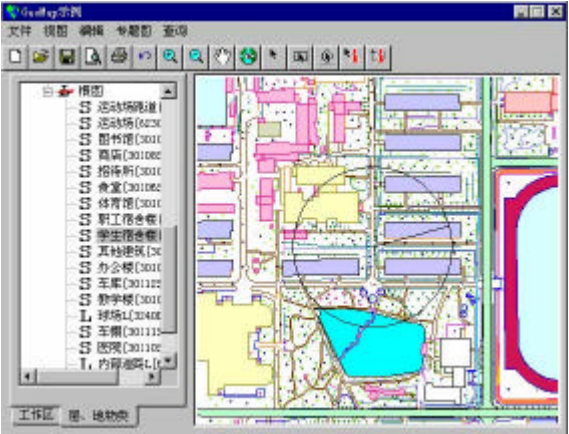


图 4-6-7

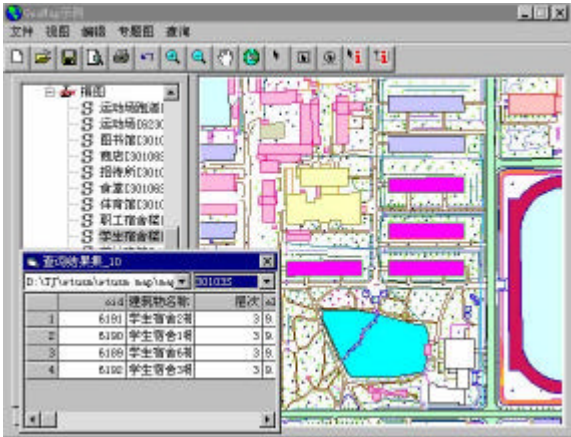


图 4-6-8

第六步：在 GeoMap 示例 Form【查询|清除查询结果】菜单中写入如下所述代码；

```
Private Sub mnuClearQuery_Click()

    Map.HighlightClearAll    清除所有高亮度显示的查询结果
End Sub
```

运行示例 Form，单击【查询|清除查询结果】菜单项，在 Geomap 控件窗口高亮度显示的地图内容恢复原来显示状态，如图 4-6-9 所示。

七 结束语

通过前面各节的学习，GIS 一步步功能的增加，一个小型的 GIS 系统已经建成，如图 4-7-1 所示。

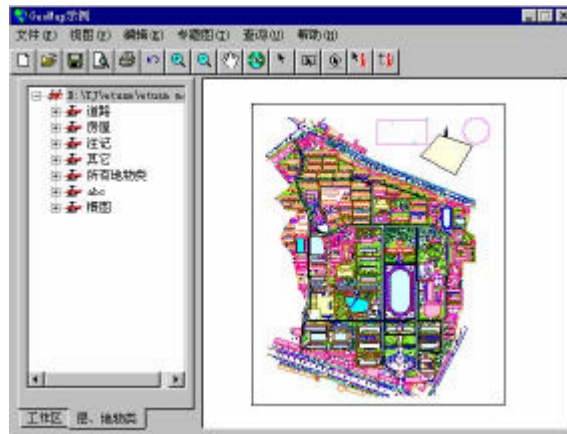


图 4-7-1

本教程所提供的示例仅为利用 Geomap 控件开发中常用的 GIS 功能，绝非 Geomap 的全部功能，仅供参考。多样化的具体功能还有待您来开发、丰富和完善。请参阅 Geomap 安装光盘所附带的 Geomap 示例和 Geomap 帮助文件。

以上我们简单介绍了部件化思想的 GIS 软件工具 GeoMap 的结构、功能及特点 GeoMap 已经能适应许多 GIS 应用的需要，特别适应于与 MIS 系统结合的应用开发。随着 DCOM 技术的成熟，GeoMap 将致力于向用户提供方便简洁的分布式网络环境下的 GIS 解决方案。

目 录

第 1 章	建立程序的环境.....	2
第 2 章	文件功能设计	9
第 3 章	视图功能设计	22
第 4 章	编辑功能设计	32
第 5 章	查询功能设计	54
第 6 章	专题图功能设计	64
第 7 章	空间分析功能设计	73
第 8 章	参数设置功能	75
第 9 章	结束语	78
附录	如何使用控件 ActiveBar 创建工具条	79

第 1 章 建立程序的环境

本章对本书所附光盘中的用 Geomap 控件编写的示例 GeomapSample 做必要的解释，并一步步地予以实现，目的在于帮助读者尽快地学习和掌握这一 GIS 开发的便利工具。使用本书的预备知识是较熟练地掌握 VB 编程的基础。在本书中不再对 VB 的相关知识多作说明。请读者参考有关 VB 的学习书籍。由于该示例中的代码行数很多，限于篇幅，这里仅对有关 Geomap 控件使用的代码进行介绍，仅起指导的作用。该示例的具体代码请参阅本书所附的光盘。

1.1 建立多文档应用程序 GeoMapSample。

1) 有关窗体：

在本节示例中引入以下窗体：

docFrm mainFrm 的子窗体。在 docFrm 窗体中使用 SplitFrame 控件 splFrame，将窗体分割为三个窗口，分别用于显示地图主窗口、索引窗口、概图窗口。

idxmapFrm 概图窗中

mainFrm 主窗体，用于总体控制程序的处理

mapFrm 用于显示地图和处理有关 Geomap 操作的窗口，在 mapFrm 窗体中添加 Geomap 控件 map

步骤如下：

请在使用 GeoMap 之前先安装 GeoMap 控件，并注册许可证。

第一步：将 GeoMap 控件加入 V B 的工具箱。

1、选择【工程|部件】菜单项(如图 1-1 所示)：

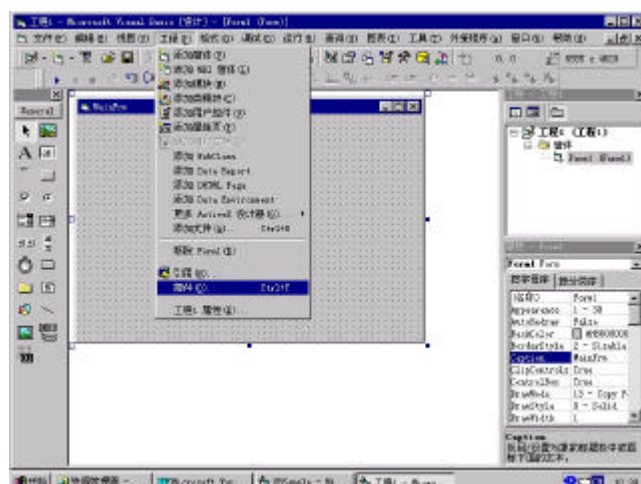


图 1-1

2、在弹出的“部件”对话框中选中 Geomap 控件 (如图 1-2 所示)：

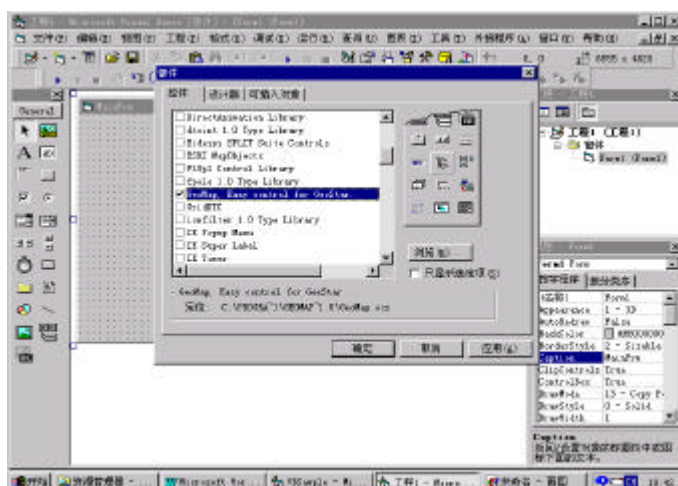


图 1-2

3、点击“确定”，Geomap 控件的就加入了 VB 工具箱。

第二步：将 Geomap 控件加入示例的 Form 中。(如图 1-3 所示)

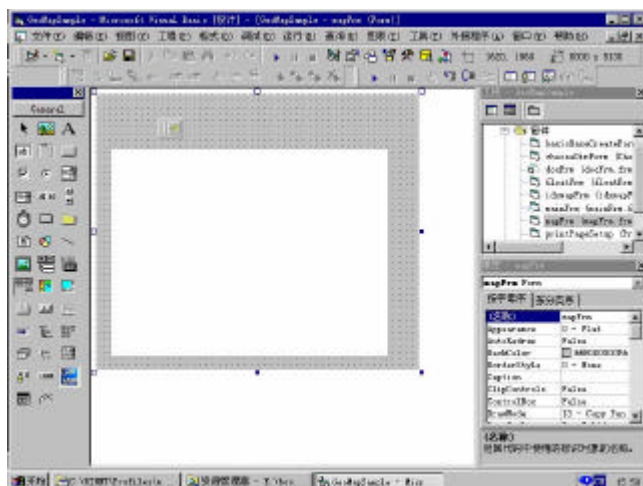


图 1-3

TvwFrm 用于维护 Geomap 数据的工作区、层、地物类的管理

2)先建立如下的程序界面，菜单框架，如图所示。

利用 Activebar 控件 barMain 建立菜单、工具条。(具体过程参考附录 1)

利用 Activebar 控件 infBar 显示当前地图窗口显示的实际比例尺。

先要建立地图显示的环境：(如图 1-4 所示)

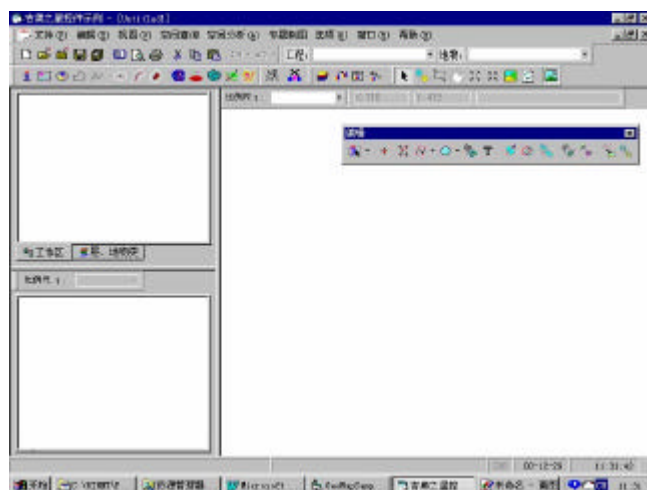


图 1-4 程序界面

菜单项的命名规则：

mnuFile 文件

mnuNewGeoMap: 新建 GeoMap 文档

mnuOpenGeoMap: 打开 GeoMap 文档

mnuCloseGeoMap 关闭 GeoMap 文档

mnuCreateWorkspace 创建工作区

mnuCreateProject 创建工程

mnuOpenWS: 打开工作区

mnuOpenPRJ: 打开工程

mnuCheckWSinPRJ : '向工程中递交工作区

mnuSaveGeoMap: '保存 GeoMap 文档'

mnuSaveAsGeoMap: '另存 GeoMap 文档'

mnuSaveAllGeoMap: '保存所有的 GeoMap 文档'

mnuCloseWS: 关闭工作区

mnuClosePRJ: '关闭工程'

mnuParamPRJWorkspaces: 拉框打开工程中的工作区，鼠标左键在概图

窗拉矩形框时触发

mnuClosePRJWorkspaces: 拉框关闭工程中的工作区，鼠标左键在概图

窗拉矩形框时触发

'tvwLayerMenu

mnutvwProperty: 层、地物类的维护，鼠标右键单击索引窗中的图标触

发

mnuPageSetup: 打印设置

mnuPrintPreview: 打印预览

mnuPrint: 打印

mnuWorkspaceIndex: 工作区显示顺序

mnuExit: '退出程序'

mnuView 视图

mnuZoomDefault: '
mnuZoomIn: 放大
mnuZoom: 缩放
mnuZoomPan: '漫游
mnuWholeMap: 全图显示
mnuRefresh: 刷新
mnuCenterZoomIn: 中心放大
mnuCenterZoomOut: 中心缩小
mnuContentVisible: 地图内容是否显示

mnuEdit 编辑

mnuEditSelectObjects: Me.ActiveForm.DoEditSelectObjectsEx '
mnuEditSelectObjectsByPoint: 点选择
mnuEditSelectObjectsByEnvelope: 矩形选择
mnuEditSelectObjectsByCircle: '圆选择
mnuEditSelectObjectsByPolygon: '多边形选择
mnuEditSelectObjectsByPolyline": 折线选择
mnuEditSelectToQueryResult: '选择结果转变为查询集
mnuCut: 剪切
mnuCopy: 复制
mnuPaste: 粘贴
mnuEditSelectObjectsMove: 移动
mnuAddAnnotation: 增加注记
mnuAddSinglePoint: 增加单点
mnuAddGroupPoint: 增加点群
mnuAddGeoLine: Me.ActiveForm.DoAddGeoLineEx '
mnuAddGeoLineCommon: '普通折线
mnuAddGeoLineCommonClose: '封闭折线
mnuAddGeoLineOrthogonal: '直角折线
mnuAddGeoLine3PointSmooths: '三点光滑线
mnuAddGeoLine5PointSmooths: '五点光滑线
mnuAddGeoLineRectangle: 矩形
mnuAddGeoLineRoundRectangle: 圆角矩形
mnuAddGeoLineEllipse: 椭圆
mnuAddGeoLineArc: 圆弧
mnuAddGeoLine3PointCircle: '三点圆
mnuAddGeoLineCenterCircle: 圆心圆
mnuAddGeoSurface: '
mnuAddGeoSurfaceCommon: 普通多边形面
mnuAddGeoSurfaceOrthogonal: 直角多边形面
mnuAddGeoSurfaceRectangle: 矩形面

mnuUndo: 撤消
mnuRedo: 重做
mnuEditSelectObjectsIncise: '分割选中的对象
mnuEditSelectSurfacesMerge: '合并选中的面对象
mnuEditSelectLinesUnite: 构建拓扑线
mnuLinesBuildSurfaces: '自动构面
mnuMakeSurfaceFromPolygons: 多边形构面
mnuEditSelectObjectsTransplant: '改变选中对象所属的地物类
mnuInciseQueryResult: '切割查询集中的对象
mnuEditSelectFeatureAll: 按地物类选择对象

mnuQuery 查询

mnuPointQuery: 点查询
mnuEnvelopeQuery: '矩形查询
mnuCircleQuery: 圆查询
mnuPolygonQuery: '多边形查询
mnuPolylineQuery: 折线查询
mnuPointBufferPick: '点缓冲区查询
mnuLineBufferPick: 线缓冲区查询
mnuSurfaceBufferPick: 面缓冲区查询
mnuContainQuery: '包含查询
mnuBelongQuery: 落入查询
mnuNeighborQuery: 邻接查询
mnuCrossQuery: 穿越查询
mnuBufferQuery: 缓冲区查询
mnuQuery: '属性查询
mnuClearQuery: 清除查询结果
mnuQueryResult: 查询结果集的维护

mnuThema 专题图

mnuThemeCreate: '创建专题图
mnuThemaOpen: '打开专题图
mnuThemaClose: '关闭专题图
mnuThemaSave: '保存专题图
mnuThemaModify: 修改专题图
mnuThemeLayout: '图面配置
mnuThemeOutline: 图廓线
mnuThemeDecorate: 文字说明
mnuThemeSettings: 图层控制
mnuThemeRefresh: 更新专题图

影象

mnuImage: '加载影象
mnuImageOrient: '影象重定向
mnuImageOff: '卸载影象
mnuMultiImage: '多影象处理
mnuImageDBOpen: 加载影象库
mnuImageDBCclose: 卸载影象库
mnuImageDBVisible: 显示影象库

analysis 空间分析

mnuBufferAnalysis: 缓冲区分析
mnuOverlayAnalysis: 叠置分析

window 窗口

mnuWindowCascade 层叠
mnuWindowTileHorizontal 水平排列
mnuWindowTileVertical 垂直排列
mnuWindowArrangeIcons 排列图标

mnuOption 选项

'设置颜色
mnubkclrmain: '主图背景色
mnubkclrsub: '索引图背景色
mnumaskclrsub: '索引图选中框颜色
mnuEditSelectObjectsLineColor: '选中对象线条颜色
mnuEditSelectObjectsHandleColor: 选中对象内点颜色
mnuEditSelectVertexColor: '选中内点颜色
mnuEditSelectAnnotationsColor: 选中注记颜色
mnuSymbolization: 地图符号化
mnuGraphicsZooming: 地图内容随图缩放
mnuAnnotationVisible: 注记显示
mnuPickTolerance: '设置查询容差
mnuBufferWidth: '设置缓冲区的宽度
mnuEditCursorSnappable: 节(内)点捕捉
mnuSimulateMouse: '鼠标模拟
mnuWholeContain: 完全包含
mnuWSInfo: 返回或改变工作区参数
mnuSelectFillTransparent", "mnuSelectFillSolid", "mnuSelectFillHorizon",
"mnuSelectFillVertical", "mnuSelectFillUpwardDiagonal",
"mnuSelectFillDownwardDiagonal", "mnuSelectFillCross",

"mnuSelectFillDiagonalCross"

在本程序中，引用了以下控件：ActiveBar，建立菜单和工具条。Splitframe，分割主窗体为三个窗口；Sstable，TreeView，用于显示工作区、层、地物类。Common Dialog, 等。在此不再作详细的说明。参考光盘中的示例，1-Start。

第 2 章 文件功能设计

在 Geomap 的应用程序中，数据组织是以工作区、工程进行分层管理的。一般地，一幅地图我们习惯地将它作为一个工作区来管理，多幅地图可以分作多个工作区来管理，也可以建立工程来管理。具体内容请参阅 GeoStar 手册。

下面介绍 GeoMap 的文件管理设计，本节的主要内容包括：创建、打开、关闭工作区，创建、打开、关闭工程，创建、打开、关闭 GeoMap 文档，向工程中递交工作区，层、地物类的维护，地图的打印：打印设置、预览、打印等。

在本示例中，引入以下窗体：

basicBaseCreateForm 设置新建工作区、工程的参数
chooseDirForm 选定要创建的工作区或工程的存放路径
featureFrm 地物类的维护：增加、选择地图符号、设置显示范围。
layerFrm 层的维护：添加、删除，层中地物类的增加、删除
WorkspaceCheckInFrm 向工程中提交工作区
printPageSetup 进行地图打印设置

模块：

baseFunction 用于编写公用变量和公用函数

2.1 具体功能实现：

1．数据组织

1) 打开工作区：

在 docFrm 的代码窗口中，加入代码：

```
Public Function OpenWorkspace() As Boolean

    Dim PathName As String
    ...
    从公共对话框中获得文件的路径名
    OpenWorkspace = mainmapFrm.map.OpenWorkspace(PathName)
    ...
End Function
```

提示：调用 map.OpenWorkspace 方法，打开工作区，将先后发生 map_AfterWorkspaceSetOpen 事件、map_AfterWorkspaceOpen 事件（map 为 GeoMap 对象的一个实例）。在这两个事件中加入维护层、地物类、工作区索引树的代码。

```
Private Sub mainmapFrm_AfterWorkspaceSetOpen(ByVal PathName As String)
    '维护地物类、工作区集名索引树
```

```

...
Set wssNode = tvw.tvwWorkspace.Nodes.Add(, LCASE(PathName), PathName,
"WSS")
Set wssNode = tvw.tvwLayer.Nodes.Add(, LCASE(PathName), PathName, "WSS")
Set wss = mainmapFrm.map.WorkspaceSets(wsskey)

Set lyrNode = tvw.tvwLayer.Nodes.Add(wssNode, tvwChild, LCASE(lyrKey),
lyr.LayerName, IIf(mainmapFrm.map.LayerVisible(lyr), "LYR1", "LYR2"))
Set feaNode = tvw.tvwLayer.Nodes.Add(lyrNode, tvwChild, LCASE(feaKey),
fea.FeatureLabel, imgKey)
...
End Sub

Private Sub mainmapFrm_AfterWorkspaceOpen(ByVal PathName As String, ByVal
IsProjectWorkspace As Boolean, ByVal OwnerPathName As String)
'维护工作区路径名索引树
...
tvw.tvwWorkspace.Nodes.Add(ownerNode, tvwChild, LCASE(PathName), PathName,
"WS").tag = "WS"
End Sub

```

在 mapFrm 的代码窗口中，加入代码：

```

Private Sub map_AfterWorkspaceSetOpen(ByVal PathName As String)
RaiseEvent AfterWorkspaceSetOpen(PathName)
End Sub

```

```

Private Sub map_AfterWorkspaceOpen(ByVal PathName As String, ByVal
IsProjectWorkspace As Boolean, ByVal OwnerPathName As String)
RaiseEvent AfterWorkspaceOpen(PathName, IsProjectWorkspace, OwnerPathName)
End Sub

```

2) 打开工程 (图 2-1)

在 docFrm 的代码窗口中，加入代码：

```

Public Function OpenProject() As Boolean
Dim PathName As String

'从公共对话框中获取工程文件路径 PathName
PathName = CommonDlg.FileName
OpenProject = mainmapFrm.map.OpenProject(PathName)

End Function

```

提示：使用 map.OpenProject 方法打开工程，将产生 map_AfterProjectOpen 事件。在该事件中加入维护地物类索引树的更新的代码。

```
Private Sub mainmapFrm_AfterProjectOpen(ByVal PathName As String)
    '此处写入维护地物类索引树的更新的代码，请参考示例
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Public Event AfterProjectOpen(ByVal PathName As String)
Private Sub map_AfterProjectOpen(ByVal PathName As String)
    RaiseEvent AfterProjectOpen(PathName)
End Sub
```

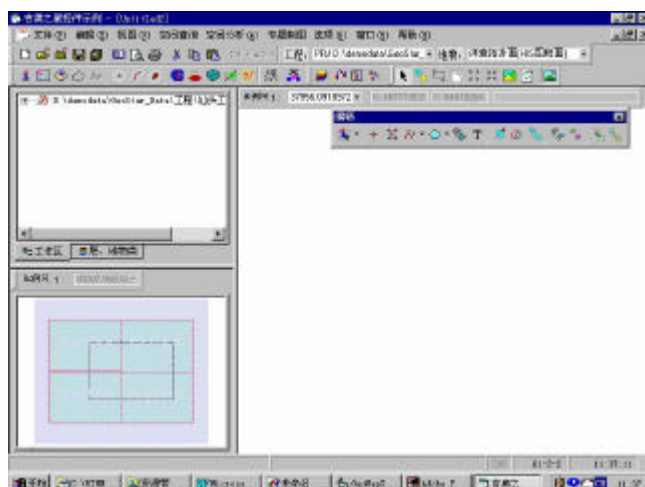


图 2-1

3) 拉框打开、关闭工程中的工作区，(图 2-2)

使用鼠标右键在概图窗内单击时弹出菜单，选择相应的菜单项后，按下并拖动鼠标获得矩形框作为 OpenPRJWorkspaces、ClosePRJWorkspaces 方法的参数，打开或关闭工程中的相应工作区。

在 idxmapFrm 的代码窗口中，加入代码：

```
Private Sub idxmap_geoMouseUp(ByVal Button As Integer, ByVal Shift As Integer,
    ByVal x As Long, ByVal y As Long)
    '在鼠标操作时，忽略。下行语句是必须的，保证鼠标按下结束上一次鼠标
    TrackAction 动作。
    If idxmap.TrackAction <> geoTrackNop Then Exit Sub
    If Not Button = 2 Then Exit Sub
    idxfrmBar.Bands("mnuPopup").TrackPopup -1, -1 将 ActiveBar 在鼠标的当前
    位置显示弹出菜单
End Sub

Private Sub idxfrmBar_Click(ByVal Tool As ActiveBarLibraryCtl.Tool)
```

```
Select Case Tool.name
Case "mnuOpenPRJWorkspaces": '拉框打开工程中的工作区
    DragOpenWorkspaces
Case "mnuClosePRJWorkspaces": '拉框关闭工程中的工作区
    DragCloseWorkspaces
End Select

End Sub

Sub DragOpenWorkspaces() '拉框打开工程中的工作区
    idxmap.MousePointer = geoCross
    mlTrackAction = 2
    idxmap.TrackEnvelope
End Sub

Sub DragCloseWorkspaces() '拉框关闭工程中的工作区

    idxmap.MousePointer = geoCross
    mlTrackAction = 3
    idxmap.TrackEnvelope
End Sub

Private Sub idxmap_EnvelopeTracked(ByVal Envelope As GeoMap.Envelope)
    Dim prjCount, i As Long
    Select Case mlTrackAction
    Case 1:
        RaiseEvent AfterTrackIndexEnvelope(Envelope)
    Case 2: 拉框打开工程中的工作区
        prjCount = idxmap.Projects.count
        For i = 0 To prjCount - 1
            idxmap.OpenPRJWorkspaces idxmap.Projects(i), Envelope
        Next i
        mlTrackAction = 1
    Case 3: 拉框关闭工程中的工作区
        prjCount = idxmap.Projects.count
        For i = 0 To prjCount - 1
            idxmap.ClosePRJWorkspaces idxmap.Projects(i), Envelope
        Next i
        mlTrackAction = 1
    End Select
    idxmap.MousePointer = geoDefault
End Sub
```

提示：在 GeoMapSample 中，打开工程的操作分为两步：第一步，打开工程后，在概图窗口中显示概图；第二步，打开工程中的工作区，在概图窗口中拉矩形框，在主窗口中显示地图数据。

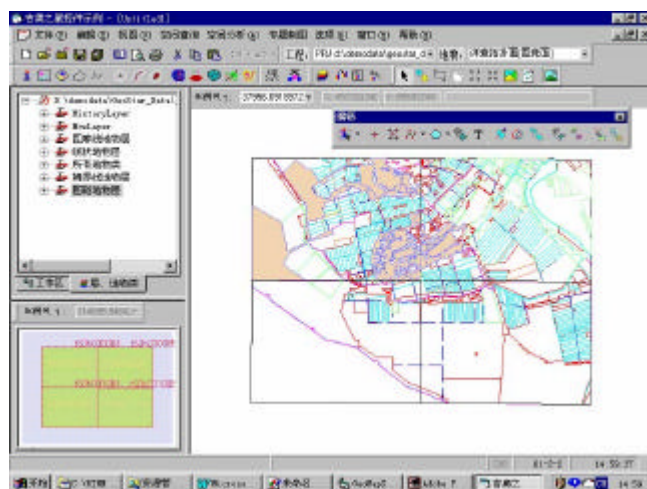


图 2-2

在 Geomap 中，我们可以使用 Geomap 文档随时保存当前地图窗口的状态信息，它是保存当前打开的工作区、工程的状态，如：显示的比例尺、地图窗口的中心、加载的影像、生成的专题图等，（特别对于需要调整地物类显示顺序的问题，特别有用）能够快速显示需要多次使用的地图窗口，极大地方便了用户的操作。

4) 新建 GeoMap 文档

在 docFrm 的代码窗口中，加入代码：

```
Public Function NewGeoMap() As Boolean
    Dim result As Boolean
    result = mainmapFrm.map.New
    submapFrm.idxmap.OpenSameAs mainmapFrm.map.Content    控制地图概图窗口与
主地图窗口内容一致，同步缩放
    submapFrm.idxmap.MaskEnvelope = mainmapFrm.map.ViewEnvelope    以掩膜来显
示概图窗口中地图的实际显示范围
    .....
End Function
```

5) 打开 GeoMap 文档

在 mainFrm 代码窗口中，加入代码：

```
Private Sub OpenGeoMap()
    Dim PathName As String
    Dim frmNewDoc As New docFrm
    .....
```

'从“打开”对话框中获取 GeoMap 文档的路径

```
frmNewDoc.OpenGeoMap PathName
```

End Sub

Public Function OpenGeoMap(ByVal PathName As String) As Boolean

Dim result As Boolean

```
result = mainmapFrm.map.Open(PathName)
```

'下面两行代码是用于控制地图概图窗口与主地图窗口内容一致，同步缩放。以掩膜来显示概图窗口中地图的实际显示范围

```
submapFrm.idxmap.OpenSameAs mainmapFrm.map.Content
```

```
submapFrm.idxmap.MaskEnvelope = mainmapFrm.map.ViewEnvelope
```

End Function

提示：如果文档中保存的是工作区的信息，将先后产生 map_AfterWorkspaceSetOpen 事件、map_AfterWorkspaceOpen 事件。如果文档保存的是工程的信息，将产生 map_AfterProjectOpen 事件。

6) 保存文档

在 docFrm 的代码窗口中，加入代码：

```
Sub SaveDoc()
```

```
mainmapFrm.map.save '保存 GeoMap 文档
```

End Sub

提示：另存 GeoMap 文档，需调用 GeoMap.SaveAs 方法。GeoMap 文档中并没有保存工程和工作区（集）中的数据，而是记录了它们的位置（绝对路径），因此，如果它们的存放位置发生了变化，创建的这个 GeoMap 文档，就不能正常地打开，需要重新生成。

7) 创建工作区

在窗体 docFrm 的代码窗口中，加入代码：

```
Sub CreateWorkspace()
```

```
'新建窗体 basicBaseCreateForm 的实例
```

```
Dim createWorkspaceFrm As basicBaseCreateForm
```

```
Set createWorkspaceFrm = New basicBaseCreateForm
```

```
createWorkspaceFrm.createType = 0 '创建的类型是工作区
```

```
createWorkspaceFrm.Show vbModal, mainFrm
```

```
mainmapFrm.map.OpenWorkspace createWorkspaceFrm.ProjectPathName
```

End Sub

在 mapFrm 的代码窗口中，加入代码：

```
Public Event FullEnvelopeChange(ByVal Envelope As GeoMap.Envelope)
Private Sub map_FullEnvelopeChange(ByVal Envelope As GeoMap.Envelope)
    RaiseEvent FullEnvelopeChange(Envelope)
End Sub
```

在窗体 basicBaseCreateForm 上引用 GeoMap 控件的对象 map，(图 2-3) 注意：在该窗体上的 GeoMap 控件的对象 map 与 mapFrm 中的应相同，用于传递参数。



图 2-3

在其代码窗口中加入代码：

```
Dim createInfo1 As GeoMap.CreateInfo 'CreateInfo 为创建参数对象
Public ProjectPathName As String
Public createType As Long '创建的类型 0: 工作区; 1:工程

Private Sub Form_Load()
    ...
    Set createInfo1 = New GeoMap.CreateInfo ' 创建 CreateInfo 对象的实例
    ...
End Sub

Private Sub cmdOK_Click()
    '获取 CreateInfo 对象的成员
    createInfo1.CreateDirectory = Me.PRJPath '工作区目录，PRJPath 为文本框
    createInfo1.CreateName = Me.PRJName '工作区名称
    createInfo1.DBScale = Me.DBScale '定义比例尺分母
    createInfo1.ResolutionXY = Me.ResolutionXY 'XY 方向的坐标精度
    createInfo1.ResolutionZ = Me.ResolutionZ 'Z 方向的坐标精度
    Dim pt As New GeoMap.Point '定义坐标原点
    pt.x = Me.OriginX
```



```

pt.y = Me.OriginY
createInfo1.Origin = pt
Dim env As New GeoMap.Envelope
env.MinX = Me.swX 'X方向西南点坐标，其中 swX 为文本框
env.MinY = Me.swY 'Y方向西南点坐标
env.MaxX = Me.neX 'X方向东北点坐标
env.MaxY = Me.neY 'Y方向东北点坐标
createInfo1.Envelope = env
'定义坐标类型（米、度）
createInfo1.CoordinateType = IIf(Me.coorMeter, geoMetre, geoDegree)

Dim ok As Boolean
If createType = 1 Then '创建工程
    ProjectPathName = Me.PRJPath & "\" & Me.PRJName & ".GPJ"
    ok = Me.map.CreateProject(createInfo1)
Else '创建工作区
    ProjectPathName = Me.PRJPath & "\" & Me.PRJName & ".GWS"
    ok = Me.map.CreateWorkspace(createInfo1)
End If
End Sub

Private Sub cmdProjection_Click() '选择投影参数
    createInfo1.projectParameter.ParameterAssign
End Sub

```

提示：调用 createInfo.projectParameter.ParameterAssign 方法，系统将弹出如下对话框（图 2-4、图 2-5），读者可进行投影参数的设置。



图 2-4



图 2-5

Private Sub cmdGetParam_Click() '选择工作区获取参数

...

从公共对话框中获取工作区文件名

Me.map.New

ok = Me.map.OpenWorkspace(Me.cmnDlg.FileName)

Dim ws As GeoMap.Workspace

Dim env As New GeoMap.Envelope

Set ws = Me.map.Workspaces(0)

With ws

env.MinX = .WSEnvelope.MinX

env.MinY = .WSEnvelope.MinY

env.MaxX = .WSEnvelope.MaxX

env.MaxY = .WSEnvelope.MaxY

createInfo1.Envelope = env

Dim pt As New GeoMap.Point

pt.x = .WSOrigin.x

pt.y = .WSOrigin.y

createInfo1.Origin = pt

createInfo1.ResolutionXY = .WSResolutionXY

createInfo1.ResolutionZ = .WSResolutionZ

createInfo1.DBScale = .WSDBScale

createInfo1.CoordinateType = .WSCoordinateType

'从已有的工作区中获取地图投影参数

createInfo1.projectParameter.ParameterCopy .WSProjectParameter

End With

With createInfo1

Me.swX = .Envelope.MinX 'swX 为当前窗体上文本框名称

Me.swY = .Envelope.MinY

Me.neX = .Envelope.MaxX

```

        Me.neY = .Envelope.MaxY
        Me.seX = Me.neX
        Me.seY = Me.swY
        Me.nwX = Me.swX
        Me.nwY = Me.neY
        Me.OriginX = .Origin.x
        Me.OriginY = .Origin.y
        Me.ResolutionXY = .ResolutionXY
        Me.ResolutionZ = .ResolutionZ
        Me.DBScale = .DBScale
        Me.coorDegree = (.CoordinateType = geoDegree)
        Me.coorMeter = (.CoordinateType = geoMetre)
    End With
    Me.map.New
End Sub

```

提示 :创建工程与创建工作区的思路类似 ,创建工程调用 Map.CreateProject *CreateInfo* 方法 ,创建工作区需调用 Map.CreateWorkspace *CreateInfo* 方法。其中 *CreateInfo* 为创建工程或工作区参数信息的对象 ,包括比例尺 ,坐标单位类型、投影参数、XYZ 方向的坐标精度、坐标原点坐标等 ,可参考示例 GeoSample 加以声明。不同之处为创建工程时 createType = 2 ,调用 Geomap.CreateProject 方法。请参考创建工作区的相关代码。创建工程之后 ,将相应的工作区数据递交进去 ,完成数据建库的工作。

8) 向工程中递交工作区

在窗体 docFrm的代码窗口中 ,加入代码 :

```

Sub CheckWSinPRJ()      '向工程中递交工作区
    '新建窗体 WorkspaceCheckInFrm 的实例
    Dim checkInFrm As WorkspaceCheckInFrm
    Set checkInFrm = New WorkspaceCheckInFrm
    Set checkInFrm.map = mainmapFrm.map
    checkInFrm.Show vbModal, mainFrm
    Set checkInFrm.map = Nothing
End Sub

```

在窗体 WorkspaceCheckInFrm 的代码窗口中 ,加入代码 :

```

Public map As GeoMap.GeoMap

Private Sub cmdCheckIn_Click()      '向工程中递交工作区
    Dim prj As GeoMap.Project
    Set prj = map.Projects(Me.comboPRJ.Text)
    With Me.lstWS
        Dim i As Long

```

```
For i = .ListCount - 1 To 0 Step -1
    If .Selected(i) Then
        Dim a As String
        a = prj.WorkspaceCheckIn(.List(i))
        '通过返回值 a，来判断某个工作区是否递交成功
        If a <> vbNullString Then .RemoveItem i
    End If
Next
End With
map.Refresh
End Sub
```

9) 关闭工作区

在 docFrm 的代码窗口中，加入代码：

```
Sub CloseWorkspace()
    ...
    获取要关闭的工作区路径名 PathName
    mainmapFrm.map.CloseWorkspace PathName
End Sub
```

10) 关闭工程

在 docFrm 的代码窗口中，加入代码：

```
Sub CloseProject()
    ...
    获取要关闭的工作区路径名 PathName
    mainmapFrm.map.CloseProject PathName
End Sub
```

2.2 图形的打印输出

Geomap 的打印输出包括：打印输出到一页和按照给定的比例尺分页打印两种方式，并提供打印输出前的预览功能。

1) 打印设置

在 docFrm 的代码窗口中，加入代码：

```
Private WithEvents pageSetupFrm As printPageSetup
```

```
Private Sub Form_Initialize()
```

```
    '建立窗体 printPageSetup 的实例
```

```
    Set pageSetupFrm = New printPageSetup
```

```
End Sub
```

```
Sub DoPageSetup()
```

```
    pageSetupFrm.Show vbModal, mainFrm
```

```
End Sub
```

在窗体 printPageSetup 的代码窗口中，加入代码：

```
Public Event PageSetuped(printStyle As GeoMap.
```

```
PrintStyleConstants, marginLeft As Single, marginRight As Single, marginTop As Single,  
    marginBottom As Single, printScale As Double)
```

```
Public Event PrinterSetupPressed()
```

```
Private Sub pageSetupFrm_PageSetuped(printStyle As GeoMap.PrintStyleConstants,  
marginLeft As Single, marginRight As Single, marginTop As Single, marginBottom As Single,  
printScale As Double)
```

```
    mainmapFrm.map.printStyle = printStyle 打印和预览的方式
```

```
    mainmapFrm.map.PrintMarginLeft = marginLeft 打印时的左页边距
```

```
    mainmapFrm.map.PrintMarginRight = marginRight 打印时的右页边距
```

```
    mainmapFrm.map.PrintMarginTop = marginTop 打印时的上页边距
```

```
    mainmapFrm.map.PrintMarginBottom = marginBottom 打印时的下页边距
```

```
    If printStyle = geoScalePrinting Then
```

```
        mainmapFrm.map.printScale = printScale 打印比例尺的分母
```

```
    End If
```

```
End Sub
```

```
Private Sub pageSetupFrm_PrinterSetupPressed()
```

```
    Me.mainmapFrm.map.DoPrinterSetup
```

```
End Sub
```

2) 打印预览

在 docFrm 的代码窗口中，加入代码：

```
Sub DoPrintPreview()
```

```
    mainmapFrm.map.DoPrintPreview
```

```
End Sub
```

提示：调用 map.DoPrintPreview 方法，系统将自动弹出对话框，按确定的打印要求，进行预览。如图 2-6 所示。

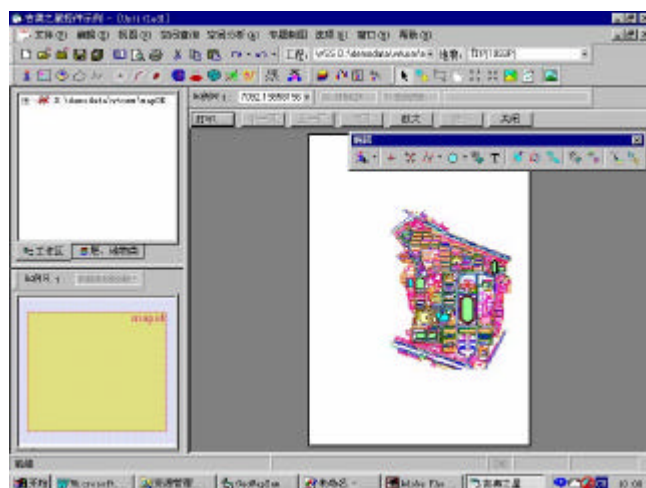


图 2-6

4) 打印

在 docFrm 的代码窗口中，加入代码：

```
Sub DoPrint()  
    mainmapFrm.map.DoPrint  
End Sub
```

提示：调用 map.DoPrint 方法，系统将自动弹出对话框（图 2-7）

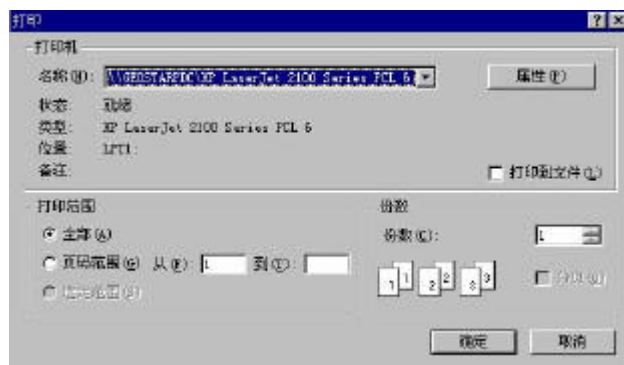


图 2-7

此示例的具体代码，请参考光盘中的 2-File。本节我们学习了 GeoMap 的文件管理设计，下一节我们来学习视图设计。

第 3 章 视图功能设计

Geomap 控件的视图操作包括地图窗口的放大、缩小、漫游、注记显示、地图符号化、随图放大、影象（库）加载等功能。

下面介绍 GeoMap 的视图操作设计：

在本示例中，引入以下窗体：

frmImgOrient 用于影像定向

3.1 功能实现：

1. 地图的缩放：

1) 放大、缩放、漫游

实现地图缩放的思路：

拉框缩放：鼠标在地图窗口中单击，产生 map_geoMouseDown 事件，在此事件中调用 map.TrackEnvelope 方法，可以用鼠标获得一个矩形框。完成后，将产生 map_EnvelopeTracked 事件，在该事件中将 ViewEnvelope 属性设置成该矩形框，则该矩形框中的内容将成为当前控件窗口中显示的内容，即实现了拉框缩放。

ViewCenterAt 方法用于在指定显示中心和显示比例尺分母的情况下，重新显示控件窗口中的内容，如果保持显示比例尺不变，改变显示的中心，即实现了漫游操作；如果保持中心不变，改变显示比例尺，即实现了中心缩放；适当地同时改变中心和显示比例尺，可以实现拉框缩放。

另外，在 map_geoMouseDown 事件中，调用 DragViewEnvelope 方法，将进行一次鼠标拖动的漫游操作。

在模块中加入代码

：

Public Enum MouseActionConstants 用枚举类型定义一个鼠标动作常数的集合

在 docFrm 的代码窗口中，加入代码：

```
Sub SetMouseAction(ByVal action As MouseActionConstants)
```

```
    mainmapFrm.map.TrackTerminate
```

```
    mainmapFrm.mlMouseAction = action ' mlMouseAction 为枚举类型变量，表示鼠标动作类型
```

```

Select Case action
Case geoMousedefault
    mainmapFrm.map.MousePointer = geoDefault
Case geoMousezoom
    mainmapFrm.map.MousePointer = geoZoom
Case geoMousezoomin
    mainmapFrm.map.MousePointer = geoZoomIn
Case geoMousezoompan
    mainmapFrm.map.MousePointer = geoZoomPan
End Select
End Sub

```

在 MapFrm的代码窗口中，加入代码：

```

Public mlMouseAction As MouseActionConstants
'mlMouseAction 为枚举类型变量，表示鼠标动作类型

Private Sub map_geoMouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal
    x As Long, ByVal y As Long)
    If map.TrackAction <> geoTrackNop Then Exit Sub    '在鼠标操作时，忽略
    geoMouseDownHandler Button, Shift, x, y

End Sub

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer,
    ByVal x As Long, ByVal y As Long)
Dim wss As WorkspaceSet
Dim prj As Project
Dim fea As Feature
Dim ws As GeoMap.Workspace
Dim isWSS As Boolean

If Button = 1 Then
    mlBtnDownX = x
    mlBtnDownY = y
    Select Case Me.mlMouseAction
    Case geoMousezoomin:
        map.TrackEnvelope
    Case geoMouseZoomOut:
        map.TrackEnvelope
    Case geoMousezoom:
        map.TrackEnvelope
    Case geoMousezoompan:

```



```

        map.DragViewEnvelope
        map.MousePointer = geoZoomPanning
    End Select
Exit Sub
End If
End Sub

Private Sub map_EnvelopeTracked(ByVal Envelope As GeoMap.Envelope)
    Select Case Me.mlMouseAction
    Case geoMousezoomin:
        If Envelope.Height = 0 Or Envelope.Width = 0 Then
            map.ViewCenterAt Envelope.CenterPoint, map.viewScale / 2
        Else
            map.ViewEnvelope = Envelope
        End If

    Case geoMouseZoomOut:
        If Envelope.Height = 0 Or Envelope.Width = 0 Then
            map.ViewCenterAt Envelope.CenterPoint, map.viewScale * 2
        Else
            Dim sscale As Double
            Dim vscale As Double
            Dim scaleChange As Double
            sscale = Envelope.Width / Envelope.Height
            vscale = map.ViewEnvelope.Width / map.ViewEnvelope.Height
            If sscale > vscale Then
                scaleChange = map.ViewEnvelope.Width / Envelope.Width
            Else
                scaleChange = map.ViewEnvelope.Height / Envelope.Height
            End If
            Dim scenter As Point 'trackenvelope 中心
            Dim vcenter As Point 原 viewenvelope 中心
            Dim dcenter As New Point 新 viewenvelope 中心

            Set scenter = Envelope.CenterPoint
            Set vcenter = map.ViewEnvelope.CenterPoint
            Set dcenter = New Point
            dcenter.x = vcenter.x + (vcenter.x - scenter.x) * scaleChange
            dcenter.y = vcenter.y + (vcenter.y - scenter.y) * scaleChange
            map.ViewCenterAt dcenter, map.viewScale * scaleChange
        End If

    Case geoMousezoom:
        If mlBtnDownY > Envelope.CenterPoint.y Then
            If Envelope.Width() <> 0 Then

```

```

        map.ViewEnvelope = Envelope
    End If
Else
    If Envelope.Height() <> 0 And Envelope.Width <> 0 Then
        sscale = Envelope.Width / Envelope.Height
        vscale = map.ViewEnvelope.Width / map.ViewEnvelope.Height
        If sscale > vscale Then
            scaleChange = map.ViewEnvelope.Width / Envelope.Width
        Else
            scaleChange = map.ViewEnvelope.Height / Envelope.Height
        End If

        Set scenter = Envelope.CenterPoint
        Set vcenter = map.ViewEnvelope.CenterPoint
        Set dcenter = New Point
        dcenter.x = vcenter.x + (vcenter.x - scenter.x) * scaleChange
        dcenter.y = vcenter.y + (vcenter.y - scenter.y) * scaleChange
        map.ViewCenterAt dcenter, map.viewScale * scaleChange

    Else
        map.ViewCenterAt Envelope.CenterPoint, map.viewScale
    End If
End If
End Select

End Sub

```

2) 中心放大

在 docFrm的代码窗口中，加入以下代码：

```

Sub DoCenterZoomIn() '中心放大
    mainmapFrm.map.viewScale = mainmapFrm.map.viewScale * K
End Sub

```

提示：此处的常数 K 可根据需要进行设置，实现中心放大，须 $K < 1$ ，如 9/10

3) 中心缩小

在 docFrm的代码窗口中，加入以下代码：

```

Sub DoCenterZoomOut() '中心缩小
    mainmapFrm.map.viewScale = mainmapFrm.map.viewScale * K

```

```

End Sub

```

提示：此处的常数 K 可根据需要进行设置，实现中心缩小功能，须 $K > 1$ ，如 $10/9$

提示：ViewScale 属性用于返回或设置当前显示比例尺的分母，增大或减小 ViewScale 属性的值，当前控件窗口中的内容将在保持中心不变的前提下缩小或放大，即实现了中心缩放。

4) 全图显示

在 docFrm 的代码窗口中，加入代码：

```
Sub SetWholeMap()      '全图显示
    mainmapFrm.map.ViewEnvelope = mainmapFrm.map.FullEnvelope
End Sub
```

5) 刷新

在 docFrm 的代码窗口中，加入代码：

```
Sub DoRefresh()
    mainmapFrm.map.Refresh
End Sub
```

6) 地图内容是否显示

在 docFrm 的代码窗口中，加入代码：

```
Sub docontentvisible()  '地图内容是否显示
    mainmapFrm.map.WorkspaceContentVisible = Not
    mainmapFrm.map.WorkspaceContentVisible
End Sub
```

提示：在本示例程序中，很多图形操作，如拉框缩放、图形查询、编辑操作等，需要频繁使用 map_geoMouseDown 事件，在这里对 map_geoMouseDown 事件作一说明：只要使用鼠标在地图窗口上点击就会产生此事件。

为了使程序清晰，在 map_geoMouseDown 事件中调用过程 sub geoMouseDownHandler。根据 mlMouseAction 的值，分别进行不同的处理。

2. 影象加载

在地图窗口中加载影象作为背景，可以充分发挥影象数据信息直观、更新迅速的优势，进行变化的动态监测和更新等，如利用航空相片、遥感影象进行土地利用、生态环境的动态监测等工作。

1) 加载影象

在模块中声明公用变量 gImageOrientKey

```
Public gImageOrientKey As String 影象处理参数
```

在 docFrm 的代码窗口中，加入代码：

```
Sub ImageOn() '加载影象
    ...从公共对话框获取影象文件名
    Me.CommonDlg.ShowOpen

    Dim paramfilename As String
    Dim pos As Long
    paramfilename = Me.CommonDlg.FileName

    gImageOrientKey = "单幅影象"

    pos = ReverseInStr(1, paramfilename, ".bmp", vbTextCompare)
    paramfilename = Left(paramfilename, pos - 1) + ".dom"
    ' 如果没有影象定位文件*.dom，拉框确定影像的大致范围
    If mainmapFrm.map.ImageOrientLoad(Me.CommonDlg.FileName, paramfilename,
    gImageOrientKey) = False Then
        mainmapFrm.map.ImageOrientNew Me.CommonDlg.FileName, gImageOrientKey
        MsgBox "选定的影像文件没有合适的定向参数，请在图上拉框确定影像的大
        致范围。"
        mainmapFrm.mlMouseDown = geoMouseImageOrientToEnvelope
        mainmapFrm.map.MousePointer = geoH_CROSS
    Else
        mainmapFrm.mbImageSettingsChanged = False
        mainmapFrm.map.Refresh
    End If

End Sub
```

在窗体 mapFrm 的代码窗口中，加入代码：

```
Public mbImageSettingsChanged As Boolean

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer,
ByVal x As Long, ByVal y As Long)
    Select Case Me.mlMouseDown
        ...
        Case geoMouseImageOrientToEnvelope
            map.TrackEnvelope
        ...
    End Sub
```

提示：在该事件中调用 `map.TrackEnvelope` 方法，系统产生 `map_EnvelopeTracked` 事件。

```
Private Sub map_EnvelopeTracked(ByVal Envelope As GeoMap.Envelope)
```

```
Select Case Me.mI_MouseAction
```

```
Case geoMouseImageOrientToEnvelope: '拉矩形框定位影像
```

```
Set map.ImageOrients(gImageOrientKey).Envelope = Envelope
```

```
map.Refresh
```

```
Me.mbImageSettingsChanged = True
```

```
...
```

```
End Sub
```

如图 3-1 所示。

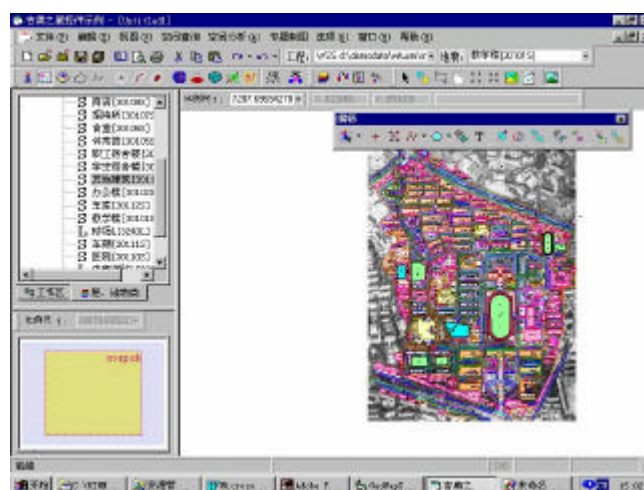


图 3-1

2) 卸载影像

在 `docFrm` 的代码窗口中，加入代码：

```
Sub ImageOff() '卸载影像
```

```
mainmapFrm.map.ImageOrientRemove gImageOrientKey
```

```
mainmapFrm.map.Refresh
```

```
End Sub
```

3) 影像重定向

影像重定向，如果影像没有定向参数或定向参数不正确，重新在影像上选取几组影像特征点与影像上征点相对应的矢量图上的点，记录其坐标，通过计算，获得其影像定向参数，使矢量图形和影像进行套合。如图 3-2 所示。



图 3-2

在 mapFrm 的代码窗口中，加入代码：

```
Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal
x As Long, ByVal y As Long)
    Select Case Me.mlMouseAction
        ...
        Case geoMouseImageOrient
            '按下 Ctrl 键，用鼠标获取矢量坐标点
            If (Shift And vbCtrlMask) Then map.TrackMapPoint
            Else:
                '直接用鼠标获取影像特征点
                map.TrackImagePoint gImageOrientKey
            End If
        ...
    End Sub
```

提示：调用 map.TrackImagePoint 方法，产生 map_ImagePointTracked 事件。
调用 map.TrackMapPoint 方法，将产生 map_MapPointTracked 事件。

```
Private Sub map_ImagePointTracked(ByVal Point As GeoMap.Point, ByVal ImageOrientKey
As String)
    frmOrient.ImagePointTracked Point, gImageOrientKey 'gImageOrientKey 是标识字符串
End Sub
```

```
Private Sub map_MapPointTracked(ByVal Point As GeoMap.Point)
    frmOrient.MapPointTracked Point '获得地图定位点坐标
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Public WithEvents frmOrient As frmImgOrient

Sub DoImageOrientAgain()
    Set frmOrient = New frmImgOrient
    Set frmOrient.frmmap = Me
    frmOrient.Show 0, Me
```

End Sub

在 frmImgOrient 的代码窗口中，加入代码：

```
Dim imageSettings As GeoMap.ImageOrientSettings 定向参数对象
```

```
Private Sub Form_Load()
```

```
    Set imageSettings = frmmap.map.ImageOrients(gImageOrientKey).Settings
```

```
End Sub
```

```
Private Sub cmdApply_Click()
```

```
    If imageSettings Is Nothing Then Exit Sub
```

```
    frmmap.map.ImageOrients(gImageOrientKey).Settings = imageSettings
```

```
    frmmap.map.Refresh
```

```
    frmmap.mbImageSettingsChanged = True
```

```
    If Not frmmap.map.ImageOrients(gImageOrientKey) Is Nothing Then
```

```
        Dim paramfilename As String
```

```
        Dim pos As Long
```

```
        paramfilename = frmmap.map.ImageOrients(gImageOrientKey).ImageFileName
```

```
        pos = ReverseInStr(1, paramfilename, ".bmp", vbTextCompare)
```

```
        paramfilename = Left(paramfilename, pos - 1) + ".dom"
```

```
        frmmap.map.ImageOrientSave gImageOrientKey, paramfilename '
```

```
    End If
```

```
End Sub
```

4) 加载影象库

在 docFrm的代码窗口中，加入代码：

```
Sub ImageDBOpen()
```

```
    ...
```

```
    '从公共对话框获取要打开影象库文件名
```

```
    Me.CommonDlg.ShowOpen
```

```
    mainmapFrm.map.ImageDBOpen Me.CommonDlg.FileName
```

```
End Sub
```

加载影象库如图 3-3、图 3-4 所示：

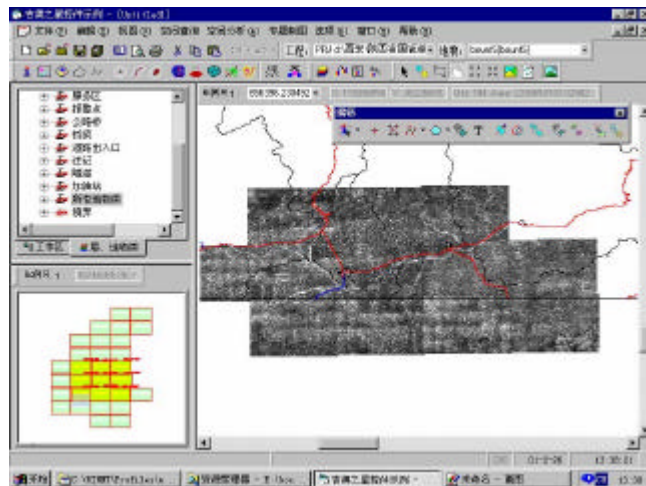


图 3-3

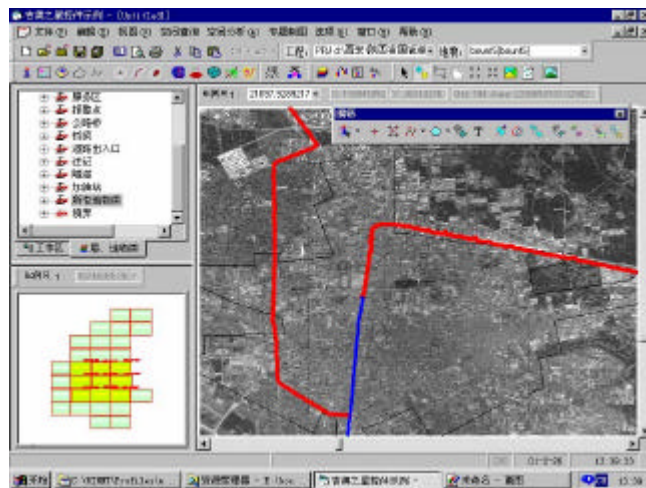


图 3-4

5) 卸载影像库

在 docFrm 的代码窗口中，加入代码：

```
Sub ImageDBCclose() '关闭影像库
    mainmapFrm.map.ImageDBCclose
End Sub
```

本节示例的具体代码，请参考光盘中的 3-View。本节我们学习了 GeoMap 的视图功能设计，下一节我们来学习编辑功能的实现。

第 4 章 编辑功能设计

人们通过地图数字化工作将图形数据输入到系统中，由于各种因素的影响，图形数据的质量可能会有各种各样的问题；另外，地图数据的不断更新，都需要大量的图形编辑工作。常用的编辑操作包括：增加单点、增加线状地物对象（普通折线、矩形、三点圆等）、面状地物对象（普通多边形面状地物、矩形面状地物等）、注记。多种方式选定需要编辑的地物；复制、粘贴、移动、删除选中的对象，分割、合并选中的面对象，多边形构面等。

下面我们学习 GeoMap 常用的编辑操作设计。

在本示例中，引入以下窗体：

AnnoAddFrm 用于添加注记

TopArgumentForm 用于保存分割面状地物的分割线

4.1 具体功能实现：

1. 选择功能

编辑操作的首要步骤就是选中要编辑的地物，点选择不受选择的当前地物类的限制，与当前鼠标的位置有关。其它各种选择操作之前，必须要选择要编辑的地物所属的地物类为当前操作的地物类。矩形选择、圆选择、多边形选择、折线选择类似。本示例以矩形选择地物为例，加以解释，圆选择、多边形选择、折线选择请参考矩形选择的相关部分。

1) 点选择、矩形选择

在模块中，加入函数 GetCurrentFeature，从工具条上获得当前操作的地物类，用于矩形选择、多边形选择、圆选择、折线选择和矩形、圆查询、多边形查询、折线查询、缓冲区查询等：

```
Function GetCurrentFeature(wss As GeoMap.WorkspaceSet, prj As GeoMap.Project, ws As  
GeoMap.Workspace, fea As GeoMap.Feature, bIsWss As Boolean) As Boolean
```

```
Dim curprj As String
```

```
Dim length As Long
```

```
Dim tag As String
```

```
Dim pos As Long
```

```
Dim curfeature As String
```

```
curfeature = mainFrm.barMain.Tools("mnuCurFeature").Text
```

```
curprj = mainFrm.barMain.Tools("mnuCurPRJ").Text
```

```
length = Len(curprj)
```

```
If length = 0 Then
```

```
GetCurrentFeature = False
```

```

Exit Function
End If
tag = Left(curprj, 3) ' "WSS", "PRJ"
curprj = Right(curprj, length - 4)
length = Len(curfeature)
pos = InStr(1, curfeature, "[")
curfeature = Mid(curfeature, pos + 1, length - pos - 1)

Select Case tag ' 数据操作的类型
Case "WSS"
    Set wss = mainFrm.ActiveForm.mainmapFrm.map.WorkspaceSets(curprj)
    Set ws = wss.Workspaces(0)
    Set fea = wss.Features(curfeature)
    bIsWss = True
Case "PRJ"
    Set prj = mainFrm.ActiveForm.mainmapFrm.map.Projects(curprj)
    Set ws = prj.Workspaces(0)
    Set fea = prj.Features(curfeature)
    bIsWss = False
End Select
GetCurrentFeature = True
End Function

```

在 docFrm 的代码窗口中，加入代码：

```

Sub DoEditSelectObjectsEx()
    If mainmapFrm.m_editSelectType < geoMouseEditSelectObjectsByPoint _
        Or mainmapFrm.m_editSelectType > geoMouseEditSelectObjectsByPolyline Then
        mainmapFrm.m_editSelectType = geoMouseEditSelectObjectsByPoint
    End If
    DoEditSelectObjects mainmapFrm.m_editSelectType
End Sub

Sub DoEditSelectObjects(ByVal action As MouseActionConstants)
    mainmapFrm.m_editSelectType = action
    Me.SetMouseAction geoMouseEditSelectObjects
End Sub

Sub SetMouseAction(ByVal action As MouseActionConstants)
    Select Case action
        ...
    Case Else
        mainmapFrm.mlOldMousePointer = mainmapFrm.map.MousePointer
        mainmapFrm.mlOldMouseAction = mainmapFrm.mlMouseAction
    End Select
End Sub

```

```
End Select
End Sub
```

在 mapFrm的代码窗口中，加入代码：

```
Public m_editSelectType As MouseActionConstants '鼠标选择操作类型

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer,
ByVal x As Long, ByVal y As Long)
    Select Case Me.mlMouseAction
        ...
    Case geoMouseEditSelectObjects
        StartEditSelectObjects
        ...
    End Select
End Sub

Private Sub map_geoMouseUp(ByVal Button As Integer, ByVal Shift As Integer, ByVal x
As Long, ByVal y As Long)
    If Button = 2 Then
        If map.TrackAction = geoEditSelectVertexMove Then
            map.EditSelectVertexMove geoVertexNext, changeOriginal
        End If
    End If
    If map.TrackAction <> geoTrackNop Then Exit Sub '在鼠标操作时，忽略
    Select Case Button
    Case 1: ' 鼠标左键
        Select Case map.MousePointer
        Case geoZoomPanning
            Me.mlMouseAction = geoMousezoompan
            map.MousePointer = geoZoomPan
        End Select
    Case 2: '鼠标右键
        ' RaiseEvent RightButtonClicked
    End Select
    '进行点选择时鼠标左键按下
    If Button = 1 And Me.mlMouseAction = geoMouseEditSelectObjects And _
        Me.m_editSelectType = geoMouseEditSelectObjectsByPoint And _
        mbEditSelectObjectsByPointDown Then
        mbEditSelectObjectsByPointDown = False
        Dim Point As GeoMap.Point
        Set Point = New GeoMap.Point
        Point.x = x
        Point.y = y
```

```

        '如果点选择没有选中地物，也可进行矩形选择
    If Not Me.DoEditSelectObjectsByPoint(Point) Then
        map.TrackEnvelope
        map.SimulateMouseMessage LButtonDown, Shift, x, y
    End If
End If
End Sub

Function DoEditSelectObjectsByPoint(ByVal Point As GeoMap.Point) As Boolean
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim count As Long
    Dim feacount As Long
    Dim i As Long, j As Long

    Dim bCtrlDown As Boolean
    bCtrlDown = ((map.KeyState(vbKeyControl) And geo_KeyDown) > 0)
    Dim bShiftDown As Boolean
    bShiftDown = ((map.KeyState(vbKeyShift) And geo_KeyDown) > 0)
    '如果 Shift 键、Ctrl 键均没有按下，清除所有选择
    If Not bCtrlDown And Not bShiftDown Then map.EditSelectObjectsReset

    '操作工作区集
    count = map.WorkspaceSets.count
    For i = 0 To count - 1
        Set wss = map.WorkspaceSets(i)
        feacount = wss.Features.count
        For j = 0 To feacount - 1
            Set fea = wss.Features(j)

            If Not bCtrlDown Then '如果 Ctrl 键没有按下去
                If map.EditSelectObjectsByPoint(Point, fea, False) Then
                    DoEditSelectObjectsByPoint = True
                Else
                    If map.EditDeselectObjectsByPoint(Point, fea) Then
                        DoEditSelectObjectsByPoint = True
                    End If
                End If
            End If
        Next j
    Next i

    '操作的是工程
    count = map.Projects.count
    For i = 0 To count - 1
        Set prj = map.Projects(i)

```

```

        feacount = prj.Features.count
    For j = 0 To feacount - 1
        Set fea = prj.Features(j)
        If Not bCtrlDown Then '如果 Ctrl 键没有按下去
            If map.EditSelectObjectsByPoint(Point, fea, False) Then
                DoEditSelectObjectsByPoint = True
            Else
                If map.EditDeselectObjectsByPoint(Point, fea) Then
                    DoEditSelectObjectsByPoint = True
                End If
            End If
        Next j
    Next i
End Function

Private Sub StartEditSelectObjects()
    Select Case Me.m_editSelectType
    Case geoMouseEditSelectObjectsByPoint '点选择
        If (map.EditCursorStatus And geoEditCursorVertexSnapped) <> 0 Then
            '开始操作捕获的节（内）点
            map.EditSelectVertexMove geoVertexStart, changeOriginal
            '如果 Ctrl 键按下，处于添加节点状态
            If (map.KeyState(geoKeyControl) And geo_KeyDown) > 0 Then
                If map.EditSelectVertexStatus = geoSelectVertexSingle Then
                    map.EditSelectVertexMove geoVertexDelete, changeOriginal
                End If
            End If
            '如果 Shift 键按下，处于添加节点状态
            If (map.KeyState(geoKeyShift) And geo_KeyDown) > 0 Then
                If map.EditSelectVertexCurrentStatus = geoSelectVertexSingle Then
                    map.EditSelectVertexMove geoVertexInsert, changeOriginal
                End If
            End If
        End If
    Else
        mbEditSelectObjectsByPointDown = True
        If (map.EditCursorStatus And geoEditCursorOnObjects) <> 0 Then
            mbEditCursorOnSelectObjects = True
        End If
        Exit Sub
    End If
End Select
Case geoMouseEditSelectObjectsByEnvelope '矩形选择
    map.TrackEnvelope
End Select
mbEditCursorOnSelectObjects = False
map.MousePointer = geoCROSSBOX

```

```

End Sub

Private Sub map_EnvelopeTracked(ByVal Envelope As GeoMap.Envelope)
    Select Case Me.mlMouseAction
        ...
    Case geoMouseEditSelectObjects
        Me.DoEditSelectObjectsByEnvelope Envelope
        ...
    End Select
End Sub

Function DoEditSelectObjectsByEnvelope(ByVal Envelope As GeoMap.Envelope) As
Boolean
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As Workspace
    Dim isWSS As Boolean

    '如果没有获取当前地物类，退出
    If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Function
    If Envelope.CenterPoint.y > mlBtnDownY Then
        If Not bCtrlDown Then
            DoEditSelectObjectsByEnvelope = map.EditSelectObjectsByEnvelope(Envelope,
fea, geoWholeContain, False)
        Else
            DoEditSelectObjectsByEnvelope = -
map.EditDeselectObjectsByEnvelope(Envelope, fea, geoWholeContain)
        End If
    Else
        If Not bCtrlDown Then
            DoEditSelectObjectsByEnvelope = map.EditSelectObjectsByEnvelope(Envelope,
fea, geoPartContain, False)
        Else
            DoEditSelectObjectsByEnvelope = map.EditDeselectObjectsByEnvelope(Envelope,
fea, geoPartContain)
        End If
    End If
End Function

Private Sub map_TrackActionFinished(ByVal TrackAction As GeoMap.TrackActionConstants)
    map.TrackColor = RGB(0, 0, 0)  黑色
    map.TrackStyle = geoLSSolid
    Select Case Me.mlMouseAction

```

Case geoMouseEditSelectObjects, geoMouseEditTrackAnnotation,
geoMouseEditTrackGeoLine, geoMouseEditTrackGeoSurface, geoMouseEditTrackGroupPoint,
geoMouseEditTrackSinglePoint

Me.mlMouseAction = geoMouseEditSelectObjects

map.MousePointer = geoCROSSBOX

If m_editSelectType < geoMouseEditSelectObjectsByPoint _

Or m_editSelectType > geoMouseEditSelectObjectsByPolyline Then

m_editSelectType = geoMouseEditSelectObjectsByPoint

End If

Case Else

map.MousePointer = Me.mlOldMousePointer

Me.mlMouseAction = Me.mlOldMouseAction

End Select

map.Refresh 此语句为消除缩放、选择后的残留线条

End Sub

矩形选择如图 4-1、4-2 所示。

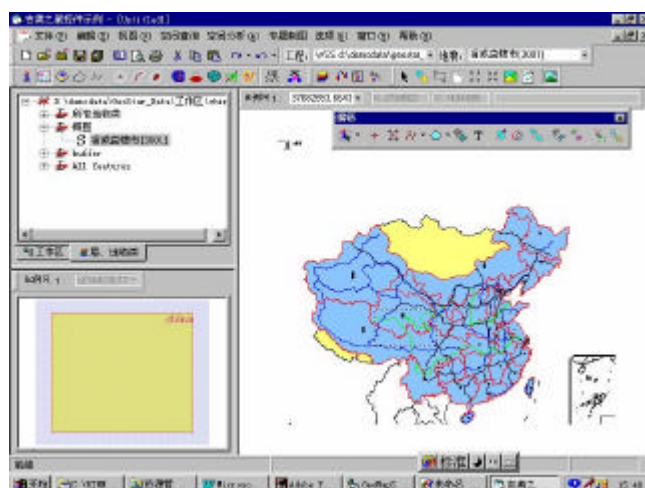


图 4-1 矩形选择前

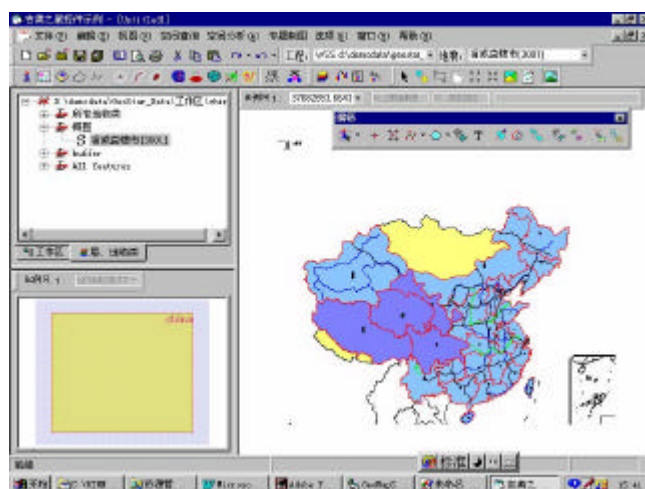


图 4-2 矩形选择后

3) 按地物类选择对象

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectFeatureAll() '全选某一地物类的所有对象
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim ws As GeoMap.Workspace
    Dim fea As GeoMap.Feature

    '从地物类索引树上，先选择当前地物类
    If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Sub
    mainmapFrm.map.EditSelectObjectsByFeature fea, False '全选某一地物类地物
End Sub
```

2 . 复制、剪切、粘贴、移动

在下面一组的编辑操作中，要进行“是否改变原始对象”的设置。

在模块中声明变量

```
Public changeOriginal As Boolean
```

在 mainFrm 的代码窗口中，加入代码：

```
Private Sub MDIForm_Load()
    ...
    changeOriginal = True '编辑操作时改变原始对象
    ...
End Sub
```

1) 剪切

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectObjectsCut() '剪切选中的对象
    mainmapFrm.map.EditSelectObjectsCut
End Sub
```

2) 复制

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectObjectsCopy() '复制选中的对象
    mainmapFrm.map.EditSelectObjectsCopy
```


End Sub

3) 粘贴

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectObjectsPaste() '粘贴选中的对象
    mainmapFrm.DoEditSelectObjectsPaste
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectObjectsPaste()
    '如果 GeoMap 剪贴板中存在选中的对象
    If (map.EditClipboardObjectsStatus And geoESSelected) > 0 Then
        Me.mI_MouseAction = geoMouseEditSelectObjectsPaste
        Me.map.MousePointer = geoDRAGCOPY
        Me.map.EditDragClipboardObjsForPaste
    End If
End Sub

Private Sub map_EditClipboardObjsDragged(ByVal deltaX As Long, ByVal deltaY As Long)
    If Me.mI_MouseAction = geoMouseEditSelectObjectsPaste Then
        map.EditSelectObjectsPaste deltaX, deltaY, changeOriginal
    End If
End Sub
```

提示：调用 EditDragClipboardObjsForPaste 方法，将产生 map_EditClipboardObjsDragged 事件，在此事件中，获取两次鼠标移动 deltaX、deltaY，作为 Geomap.EditSelectObjectsPaste 方法的参数，实现粘贴的效果。具体操作时要用鼠标拖动几何对象至一个新的位置即可。

4) 移动

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectObjectsMove()
    If (map.EditSelectObjectsStatus And geoESSelected) > 0 Then
        Me.mI_MouseAction = geoMouseEditSelectObjectsMove
        Me.map.MousePointer = geoSizeAll
        Me.map.EditDragSelectedObjsForMove
    End If
End Sub
```

```

Private Sub map_EditSelectedObjsDragged(ByVal deltaX As Long, ByVal deltaY As Long)
    If Me.mlMouseAction = geoMouseEditSelectObjectsMove Then
        map.EditSelectObjectsMove deltaX, deltaY, changeOriginal
    End If
End Sub

```

提示：移动选中的地物，需调用 Geomap.EditDragSelectedObjsForMove 方法，产生 Geomap_EditSelectedObjsDragged 事件。在此事件中，获取两次鼠标移动 deltaX、deltaY，作为 Geomap.EditSelectObjectsMove 方法的参数，实现移动的效果。具体操作时要用鼠标拖动几何对象至一个新的位置即可。

3. 添加地物

1) 增加单点

在 docFrm 的代码窗口中，加入代码：

```

Sub DoAddSinglePoint() '增加单点地物
    Me.SetMouseAction geoMouseEditTrackSinglePoint
End Sub

```

在 mapFrm 的代码窗口中，加入代码：

```

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer,
ByVal x As Long, ByVal y As Long)
    Select Case Me.mlMouseAction
        ...
        Case geoMouseEditTrackSinglePoint
            StartEditTrackSinglePoint
        Case geoMouseEditTrackGeoLine
            StartEditTrackGeoLine
        Case geoMouseEditTrackGeoSurface
            StartEditTrackGeoSurface
        Case geoMouseEditTrackAnnotation
            StartEditTrackAnnotation
        ...
    End Select
End Sub

Private Sub StartEditTrackSinglePoint()
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature

```

```

Dim ws As GeoMap.Workspace

If GetCurrentFeature(wss, prj, ws, fea, False) Then
    map.EditTrackSinglePoint ws, fea
End If
End Sub

```

提示：增加点群与增加单点实现思路相同，请参考相应的代码部分。

2) 普通折线

在 docFrm的代码窗口中，加入代码：

```

Sub DoAddGeoLine(ByVal lineType As GeoMap.GeoLineTypeConstants)
    '添加线形地物
    mainmapFrm.m_addGeoLineType = lineType
    Select Case lineType
        ...
        Case GeoMap.geoLTCommon
        ...
    End Select
    Me.SetMouseAction geoMouseEditTrackGeoLine
End Sub

```

在 mapFrm的代码窗口中，加入代码：

```

Public m_addGeoLineType As GeoMap.GeoLineTypeConstants
Public m_addGeoSurfaceType As GeoMap.GeoSurfaceTypeConstants

Private Sub StartEditTrackGeoLine() '添加线状地物
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As GeoMap.Workspace

    If GetCurrentFeature(wss, prj, ws, fea, False) Then
        map.EditTrackGeoLine ws, fea, Me.m_addGeoLineType
    End If
End Sub

```

提示：添加各种线状地物均需调用 Geomap.EditTrackGeoLine 方法，前两个参数分别为要添加的线状地物所在的工作区和地物类，最后的参数是线状的类型：封闭折线、直角折线、三点光滑线、五点光滑线、矩形、圆角矩形、椭圆、圆弧、三点圆、圆心圆等，具体取值，请参考线状地物类型常量 GeoLineType。

3) 添加面状地物

在 docFrm 的代码窗口中，加入代码：

```
Sub DoAddGeoSurface(ByVal surfaceType As GeoMap.GeoSurfaceTypeConstants)
    mainmapFrm.m_addGeoSurfaceType = surfaceType
    Select Case surfaceType
        ...
        Case GeoMap.geoSTCommon
            ...
    End Select
    Me.SetMouseAction geoMouseEditTrackGeoSurface
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Private Sub StartEditTrackGeoSurface()
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As GeoMap.Workspace

    If GetCurrentFeature(wss, prj, ws, fea, False) Then
        map.EditTrackGeoSurface ws, fea, Me.m_addGeoSurfaceType
    End If
End Sub
```

提示：添加各种面状地物均需调用 Geomap.EditTrackGeoSurface 方法，前两个参数分别为要添加的面状地物所在的工作区和地物类，最后的参数是面状地物的类型：普通多边形面、矩形面、直角多边形面等，具体取值，请参考面状地物类型常量 GeoSurfaceType。

提示：添加各种空间地物将产生 Geomap_GeoEditTrack 事件，在此事件中返回添加对象的 oid 和所属的工作区、地物类。

4) 增加注记

在 docFrm 的代码窗口中，加入代码：

```
Sub DoAddAnnotation() 添加注记
    If Not mainmapFrm.map.AnnotationVisible Then
        mainmapFrm.map.AnnotationVisible = True
        mainmapFrm.map.Refresh
    End If
```

```

If Not mainmapFrm.annoFrm.Visible Then mainmapFrm.annoFrm.Show , mainFrm
Me.SetMouseAction geoMouseEditTrackAnnotation
End Sub

```

在 mapFrm的代码窗口中，加入代码：

```

Public WithEvents annoFrm As AnnoAddFrm ' AnnoAddFrm 用于注记的添加

Private Sub Form_Load()
    ...
    Set annoFrm = New AnnoAddFrm
    Set annoFrm.mAnnotationSettings = New GeoMap.AnnotationSettings
    annoFrm.mAnnotationString = "注记"
    ...
End Sub

Sub SetMouseAction(ByVal action As MouseActionConstants)
    Select Case mainmapFrm.mlMouseAction
        ...
        Case geoMouseEditTrackAnnotation
            If action <> geoMouseEditTrackAnnotation Then Unload mainmapFrm.annoFrm
            ...
    End Select
End Sub

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal
x As Long, ByVal y As Long)
    Select Case Me.mlMouseAction
        ...
        Case geoMouseEditTrackAnnotation
            StartEditTrackAnnotation
            ...
    End Select
End Sub

Private Sub StartEditTrackAnnotation() '添加注记
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As GeoMap.Workspace

    '必须首先选择当前地物类
    If GetCurrentFeature(wss, prj, ws, fea, False) Then
        map.EditTrackAnnotation ws, fea, annoFrm.mAnnotationString,

```

annoFrm.mAnnotationSettings

End If

End Sub

如图 4-3、图 4-4、图 4-5。

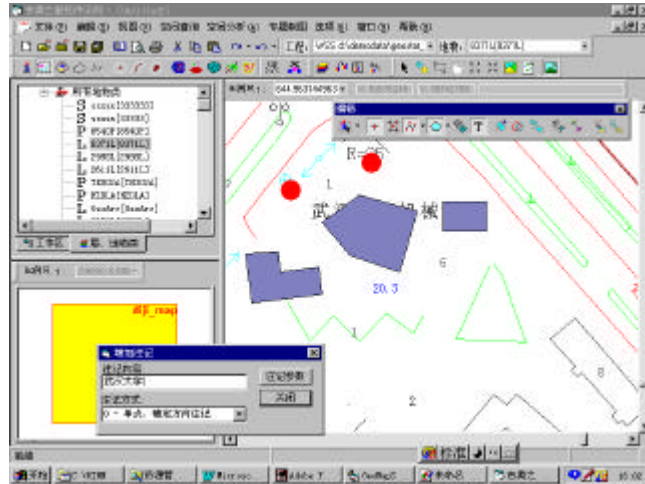


图 4-3

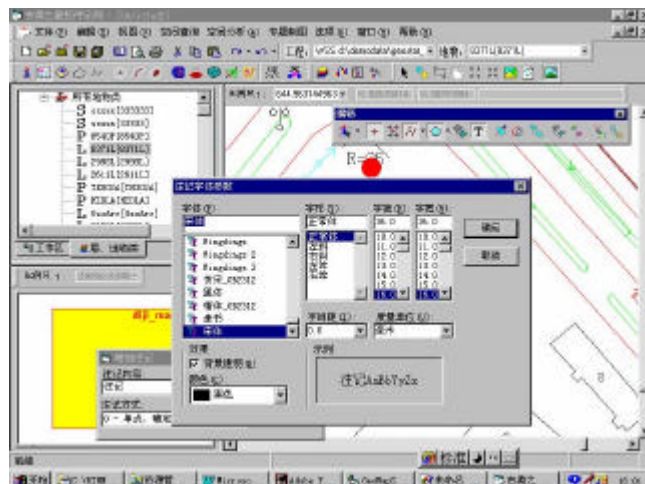


图 4-4

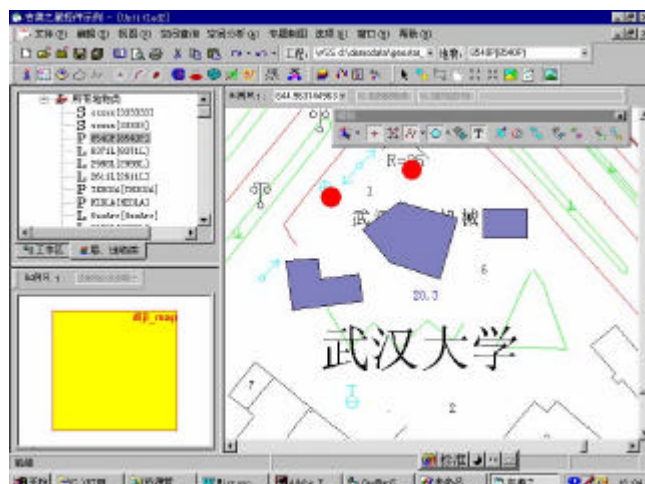


图 4-5

5) 撤消

在 docFrm 的代码窗口中，加入代码：

```
Sub DoUndo()  
    mainmapFrm.DoUndo  
End Sub
```

提示：GeoMap 控件中当前的内容经过编辑之后，方可进行 Undo 操作。
请参考 GeoMap.EditUndoable 属性。

6) 重做

在 docFrm 的代码窗口中，加入代码：

```
Sub DoRedo()  
    mainmapFrm.DoRedo  
End Sub
```

提示：GeoMap 控件中当前的内容经过 Undo 操作，方可进行 Redo 操作。
请参考 GeoMap.EditRedoable 属性。

7) 分割选中的对象

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectObjectsIncise()    '分割  
    Me.SetMouseAction geoMouseEditSelectObjectsIncise  
End Sub  
  
Sub SetMouseAction(ByVal action As MouseActionConstants)  
    mainmapFrm.mlMouseAction = action  
    Select Case action  
        ...  
        Case geoMouseEditSelectObjectsIncise    '分割选中的地物  
            mainmapFrm.map.MousePointer = geoPALETTE  
        ...  
    End Select  
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal  
x As Long, ByVal y As Long)  
    Dim topFrm As TopArgumentForm 用于面的分割操作  
    Select Case Me.mlMouseAction
```

```

...
Case geoMouseQueryPolyline, geoMouseEditSelectObjectsIncise,
geoMouseQueryLineBufferPick    折线查询、分割对象、线缓冲区选择
    DoClearQuery
    If currentPolylinesCount = 0 Then map.DrawClear
    map.TrackPolyline
Case geoMouseEditSelectObjectsIncise '
    If currentPolylinesCount <> 0 Then
        If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Sub
        Set topFrm = New TopArgumentForm
        If isWSS Then
            Set topFrm.curFeatures = wss.Features
        Else
            Set topFrm.curFeatures = prj.Features
        End If
        Set topFrm.curFea = fea
        topFrm.Show vbModal, mainFrm
        If Not topFrm.chosenCancel Then
            map.EditSelectObjectsIncise currentPolylines, topFrm.chosenID,
changeOriginal
        End If
        Set topFrm.curFeatures = Nothing
        Set topFrm.curFea = Nothing
        Set topFrm = Nothing
    End If
    map.DrawClear

Me.mlMouseAction = geoMouseEditSelectObjects
map.MousePointer = geoCROSSBOX
If m_editSelectType < geoMouseEditSelectObjectsByPoint _
    Or m_editSelectType > geoMouseEditSelectObjectsByPolyline Then
    m_editSelectType = geoMouseEditSelectObjectsByPoint
End If
...
End Select
End Sub

Private Sub map_PolylineTracked(ByVal Polyline As GeoMap.Polyline)
    Select Case Me.mlMouseAction
        Case geoMouseQueryPolyline, geoMouseQueryLineBufferPick,
geoMouseEditSelectObjectsIncise
            map.DrawPolyline Polyline, IIf(Me.mlMouseAction =
geoMouseQueryLineBufferPick, RGB(0, 0, 255), RGB(255, 0, 0))
            currentPolylinesCount = currentPolylinesCount + 1
    End Select
End Sub

```



```

ReDim Preserve currentPolylines(0 To currentPolylinesCount - 1)
Set currentPolylines(currentPolylinesCount - 1) = Polyline
If (map.KeyState(geoKeyShift) And geo_KeyDown) = 0 Then
    geoMouseDownHandler 2, 0, 0, 0
End If
Case geoMouseEditSelectObjects
    Me.DoEditSelectObjectsByPolyline Polyline
End Select

End Sub

```

分割选中的面状地物的过程，如图 4-6、图 4-7、图 4-8 所示。

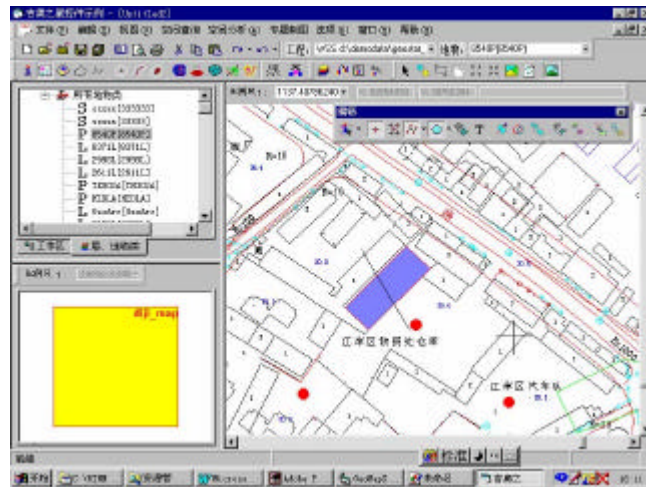


图 4-6

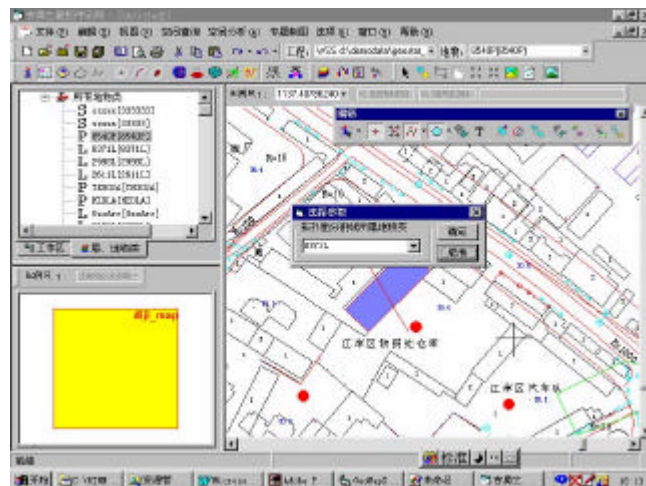


图 4-7

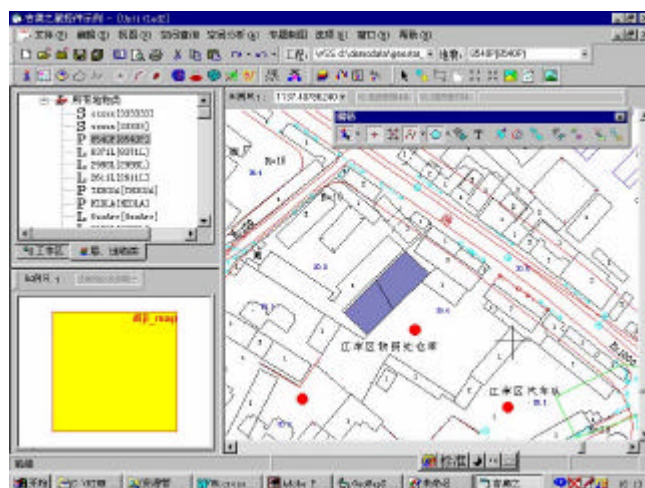


图 4-8

8) 合并选中的面对象

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectSurfacesMerge() '合并选中的面地物
    mainmapFrm.DoEditSelectSurfacesMerge
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectSurfacesMerge() '对选中的面进行合并
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As GeoMap.Workspace

    If GetCurrentFeature(wss, prj, ws, fea, False) Then
        If fea.GeometryType = geoSurface Then
            map.EditSelectSurfacesMerge fea.FeatureID, changeOriginal
        End If
    End If
End Sub
```

如图 4-9、图 4-10 所示。

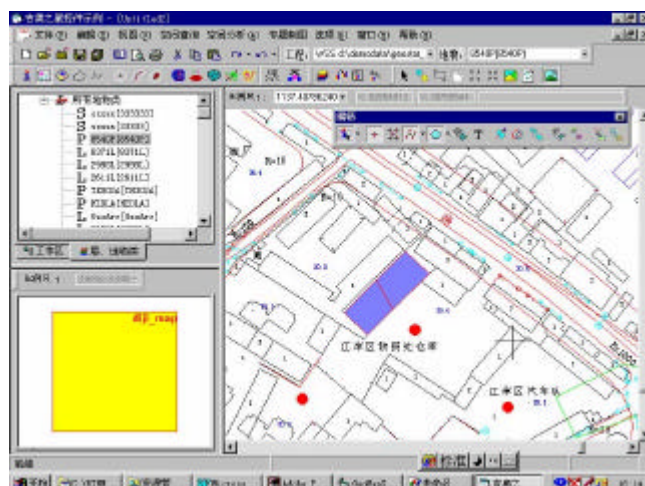


图 4-9 合并前

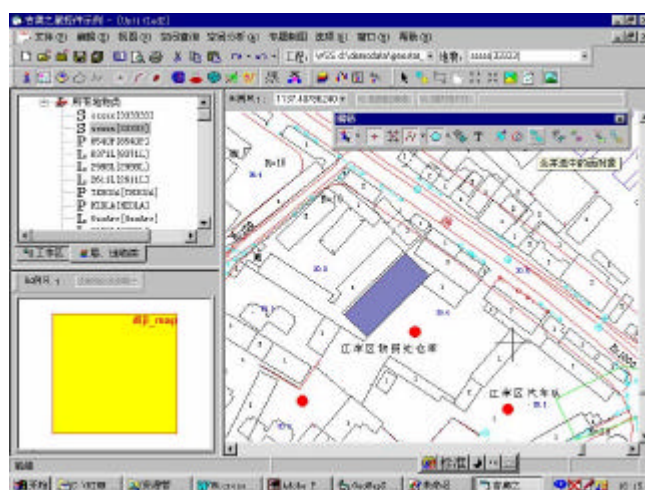


图 4-10 合并后

9) 构建拓扑线

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectLinesUnite() '线合并
```

```
    If mainmapFrm.map.WorkspaceSets.count = 0 And mainmapFrm.map.Projects.count = 0 Then Exit Sub
```

```
    mainmapFrm.DoEditSelectLinesUnite
```

```
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectLinesUnite() '线合并
```

```
    Dim wss As GeoMap.WorkspaceSet
```

```
    Dim prj As GeoMap.Project
```

```
    Dim fea As GeoMap.Feature
```

```
    Dim ws As GeoMap.Workspace
```

```

'必须首先选择线地物类为当前地物类
If GetCurrentFeature(wss, prj, ws, fea, False) Then
    If fea.GeometryType = geoLine Then
        map.EditSelectLinesUnite fea.FeatureID
    Else
        Beep
    End If
End If
End Sub

```

10) 自动构面

在 docFrm 的代码窗口中，加入代码：

```

Sub DoEditSelectLinesBuildSurfaces() '线构面
    mainmapFrm.DoEditSelectLinesBuildSurfaces
End Sub

```

在 mapFrm 的代码窗口中，加入代码：

```

Sub DoEditSelectLinesBuildSurfaces()
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As GeoMap.Workspace

    If GetCurrentFeature(wss, prj, ws, fea, False) Then
        If fea.GeometryType = geoSurface Then
            map.EditSelectLinesBuildSurfaces fea.FeatureID, changeOriginal
            'fea.FeatureID 为线构面后存放面的地物类的地物类编码
        Else
            Beep
        End If
    End If
End Sub

```

如图 4-11、图 4-12、图 4-13 所示。

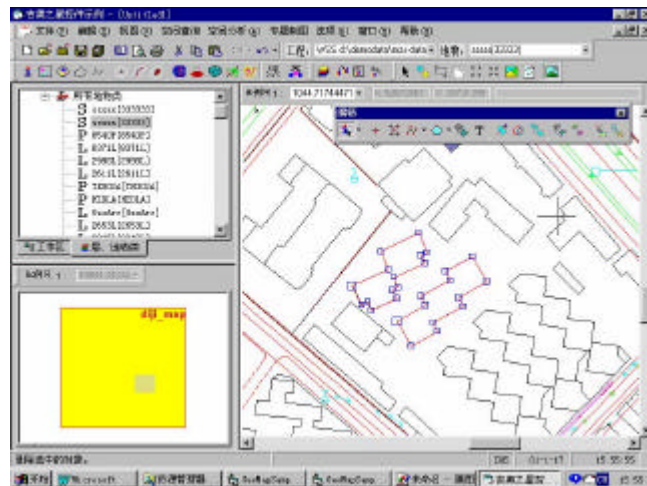


图 4-11 选中构面的线地物

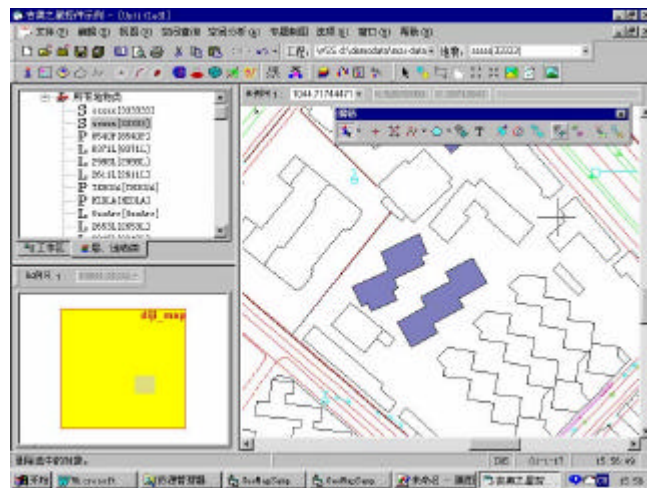


图 4-12 构面

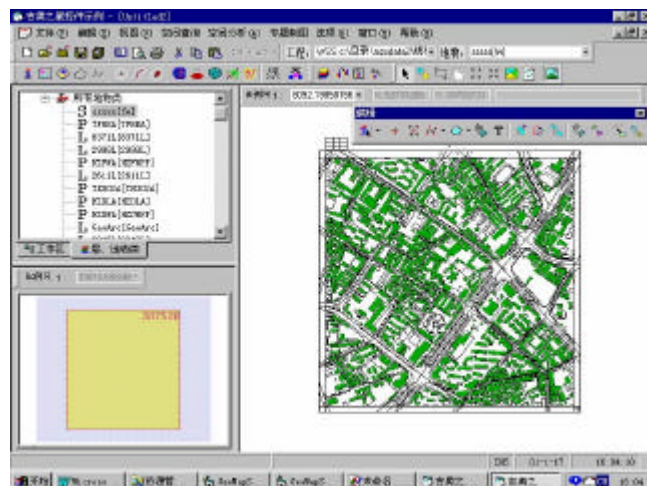


图 4-13 对一个工作区的数据构面

11) 分割查询集中的对象

在 docFrm 的代码窗口中，加入代码：

```
Sub DoEditSelectInciseQueryResult()    '分割查询集中的对象
    mainmapFrm.map.EditSelectLinesBreak mainmapFrm.currentSaveName,
changeOriginal
End Sub
```

本示例具体的代码请参考随书所带光盘 4-Edit。本节学习了利用 Geomap 控件实现常用的一些编辑功能设计。下一节介绍查询常用功能的设计。

第 5 章 查询功能设计

GeoMap 控件的查询操作，包括按照点查询、线查询、面查询操作方式，利用缓冲分析计算功能实现的缓冲查询操作，对查询结果的突出显示和维护，空间对象与属性数据的双向查询等。

下面我们学习 GeoMap 的常用的查询功能的设计。

在本节的示例中，引入以下窗体：

ptResultFrm 点查询结果
gridResultFrm 查询结果表格

5.1 功能实现：

本节以点查询和矩形查询为例，介绍如何实现其功能。

首先，在 VB 中添加类型库 msado20.tlb 到工程中去。从“工程”菜单中的“引用”菜单项中，引用 Microsoft ActiveX Data Objects 2.0 Library.

1. 点查询

在 docFrm 的代码窗口中，加入代码：

```
Sub SetMouseAction(ByVal action As MouseActionConstants)
    mainmapFrm.m1MouseAction = action
    Select Case action
        ...
        Case geoMouseQueryPoint
            mainmapFrm.map.MousePointer = geoCross
        Case geoMouseQueryLineBufferPick
            mainmapFrm.map.MousePointer = geoCross
        ...
    End Select
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Public WithEvents ptQryFrm As ptResultFrm

Private Sub Form_Load()
    ...
    Set ptQryFrm = New ptResultFrm
    ...
End Sub

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal
```

```

x As Long, ByVal y As Long)
    Select Case Me.mlMouseDown
        ...
        Case geoMouseDown:
            DoClearQuery
            map.DrawClear
            QueryByPoint x, y
            ...
    End Select
End Sub

```

```

Sub QueryByPoint(ByVal x As Long, ByVal y As Long)
    Dim pt As New Point
    Dim wsStep, wscount As Long
    Dim wss As WorkspaceSet
    Dim prj As Project
    Dim fea As Feature
    Dim feaStep, feacount As Long
    Dim stepname As String
    Dim savename As String

    Set pt = New Point
    pt.x = x
    pt.y = y

    wscount = map.WorkspaceSets.count    '查询工作区集
    For wsStep = 0 To wscount - 1
        Set wss = map.WorkspaceSets(wsStep)
        feacount = wss.Features.count
        For feaStep = 0 To feacount - 1
            Set fea = wss.Features(feaStep)
            If map.featureVisible(fea) Then
                stepname = "点查询中间结果"
                map.QueryObjectsByPoint pt, fea, stepname
                If stepname <> vbNullString Then
                    map.QueryResults.Plus stepname, savename, savename
                    map.QueryResults.Remove stepname
                End If
            End If
        Next feaStep
    Next wsStep

    wscount = map.Projects.count    '查询工程
    For wsStep = 0 To wscount - 1

```



```

Set prj = map.Projects(wsStep)
feacount = prj.Features.count
For feaStep = 0 To feacount - 1
    Set fea = prj.Features(feaStep)
    If map.featureVisible(fea) Then
        stepname = "点查询中间结果"
        map.QueryObjectsByPoint pt, fea, stepname
        If stepname <> vbNullString Then
            map.QueryResults.Plus stepname, savename, savename
            map.QueryResults.Remove stepname
        End If
    End If
Next feaStep
Next wsStep
QueryResultOpenByPointQueryForm savename
End Sub

```

'打开点查询结果集

```

Sub QueryResultOpenByPointQueryForm(ByVal savename As String)
    Me.map.HighlightClearAll
    moldhighlightID = -1
    Me.QueryResultHighlight savename, RGB(255, 0, 255)
    '弹出查询结果表格
    Me.ptQryFrm.UpdateResultData Me, savename
    currentSaveName = savename
End Sub

```

'高亮度显示查询结果集

```

Sub QueryResultHighlight(savename As String, clr As OLE_COLOR)
    Dim queryResult As GeoMap.queryResult
    Dim qrWorkspace As GeoMap.QueryResultOfWorkspace
    Dim qrFeature As GeoMap.QueryResultOfFeature
    Dim i As Long
    Dim j As Long
    Dim wscount As Long
    Dim feacount As Long

```

```

Set queryResult = map.QueryResults(savename)
wscount = queryResult.WorkspaceCount
For i = 0 To wscount - 1
    Set qrWorkspace = queryResult(i)
    If Not qrWorkspace Is Nothing Then
        feacount = qrWorkspace.FeatureCount
        For j = 0 To feacount - 1

```

```

Set qrFeature = qrWorkspace(j)
map.HighlightObjects qrWorkspace.Workspace, qrFeature.oidArray, clr
Next j
End If
Next i
End Sub

```

点查询如图 5-1。

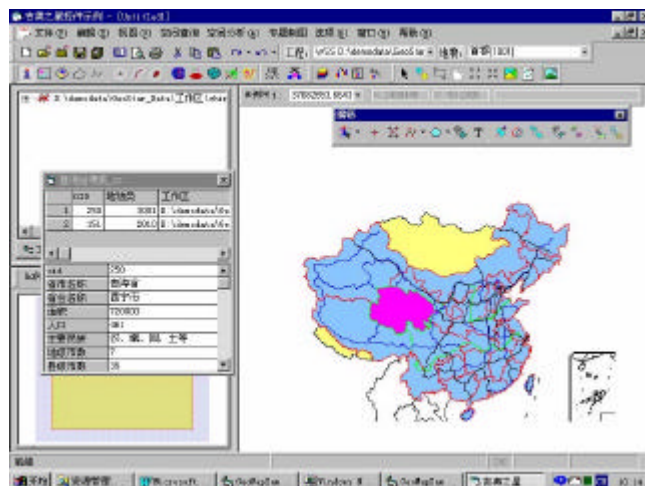


图 5-1 点查询

提示：实现点查询的思路：鼠标在地图窗口中单击，产生 map_geoMouseDown 事件，在该事件中调用 map.QueryObjectsByPoint 方法，将查询结果在地图窗口中高亮度显示，并弹出窗口显示查询结果。点查询操作之前不需要选取当前地物类，而其它查询方式必须先选择当前地物类。

2. 矩形查询

在 docFrm 的代码窗口中，加入代码：

```

Sub SetMouseAction(ByVal action As MouseActionConstants)
    Select Case action
        ...
        Case geoMouseQueryEnvelope
            mainmapFrm.map.MousePointer = geoCross
        ...
    End Select
End Sub

```

在 mapFrm 的代码窗口中，加入代码：

```

Dim currentEnvelopes() As GeoMap.Envelope
Dim currentEnvelopesCount As Long
Public WithEvents gridQryFrm As gridResultFrm

```

```
Private Sub Form_Load()
```

```
...
```

```
' 建立窗体 gridResultFrm 的实例
```

```
Set gridQryFrm = New gridResultFrm
```

```
...
```

```
End Sub
```

```
Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer,  
ByVal x As Long, ByVal y As Long)
```

```
Select Case Me.mlMouseAction
```

```
Case geoMouseQueryEnvelope
```

```
DoClearQuery
```

```
If currentEnvelopesCount = 0 Then map.DrawClear
```

```
map.TrackEnvelope
```

```
...
```

```
End Select
```

```
Select Case Me.mlMouseAction
```

```
Case geoMouseQueryEnvelope: '
```

```
QueryByEnvelopeArray currentEnvelopes
```

```
...
```

```
End Select
```

```
End Sub
```

```
Private Sub map_EnvelopeTracked(ByVal Envelope As GeoMap.Envelope)
```

```
Select Case Me.mlMouseAction
```

```
...
```

```
Case geoMouseQueryEnvelope:
```

```
map.DrawEnvelope Envelope, RGB(255, 0, 0)
```

```
currentEnvelopesCount = currentEnvelopesCount + 1
```

```
ReDim Preserve currentEnvelopes(0 To currentEnvelopesCount - 1)
```

```
Set currentEnvelopes(currentEnvelopesCount - 1) = Envelope
```

```
If (map.KeyState(geoKeyShift) And geo_KeyDown) = 0 Then
```

```
geoMouseDownHandler 2, 0, 0, 0
```

```
End If
```

```
map.DrawClear 消除空间查询所遗留的线画
```

```
...
```

```
End Select
```

```
End Sub
```

```
Sub QueryByEnvelopeArray(envelopearray As Variant)
```

```
Dim wss As GeoMap.WorkspaceSet
```

```
Dim prj As GeoMap.Project
```

```

Dim fea As GeoMap.Feature
Dim ws As GeoMap.Workspace
Dim querymode As QueryModeConstants
Dim isWSS As Boolean
Dim savename As String
Dim includename() As String
Dim includecount As Long
Dim i As Long
Dim stepname As String

includecount = 0
' 如果没有选择当前地物类,退出此过程
If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Sub
'从工具条上判断查询方式：完全包含查询或部分包含查询
querymode = IIf(mainFrm.barMain.Tools("mnuWholeContain").Checked,
geoWholeContain, geoPartContain)
For i = LBound(envelopearray) To UBound(envelopearray)
    '
    If Not envelopearray(i) Is Nothing Then
        includecount = includecount + 1
        stepname = "包含结果集" & includecount
        map.QueryObjectsByEnvelope envelopearray(i), fea, querymode, stepname
        ReDim Preserve includename(0 To includecount - 1)
        includename(includecount - 1) = stepname
    End If
Next I
'将包含结果集加入到 savename 中
If includecount > 0 Then map.QueryResults.Merge includename, savename
For i = 0 To includecount - 1
    '清除查询得到的包含结果集
    map.QueryResults.Remove includename(i)
Next i
If savename <> vbNullString Then QueryResultOpen savename
End Sub

Sub QueryResultOpen(ByVal savename As String) 打开查询结果集
    moldhighlightID = -1
    Me.QueryResultHighlight savename, RGB(255, 0, 255)
    Me.gridQryFrm.UpdateResultData Me, savename
    currentSaveName = savename
End Sub

```

矩形查询如图 5-2、图 5-3 所示。

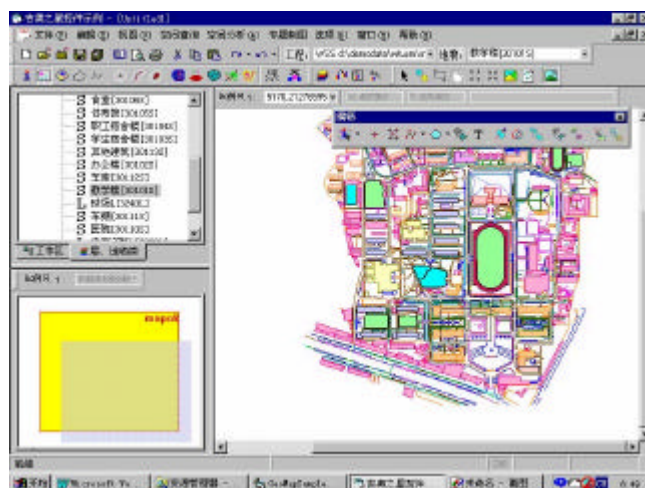


图 5-2 矩形查询前

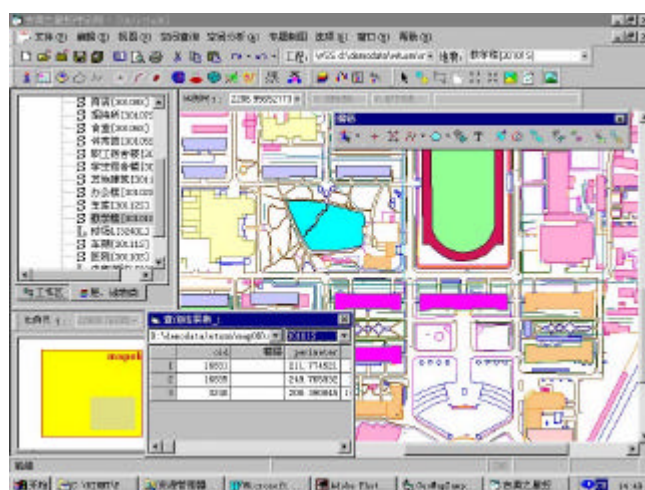


图 5-3 矩形查询后

提示：实现矩形查询的思路：鼠标在地图窗口中单击，产生 map_geoMouseDown 事件，在此事件中调用 map.TrackEnvelope 方法，获得矩形框。将产生 map_EnvelopeTracked 事件，在该事件中调用 map.QueryObjectsByEnvelope 方法，将查询结果在地图窗口中高亮度显示，并弹出窗口显示查询结果。

圆查询、多边形查询、折线查询与矩形查询的编程思路类似。不同之处为圆查询、多边形查询、折线查询分别调用 map.QueryObjectsByCircle、map.QueryObjectsByPolygon、map.QueryObjectsByPolyline 方法。

3. 线缓冲区查询

在 mapFrm 的代码窗口中，加入代码：

```
Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal x As Long, ByVal y As Long)
```

```

        Select Case Me.mlMouseAction
            ...
        Case geoMouseQueryPolyline, geoMouseQueryLineBufferPick
            '折线查询、线缓冲区选择
            DoClearQuery
            If currentPolylinesCount = 0 Then map.DrawClear
            map.TrackPolyline
            ...
        End Select
    End Sub

Private Sub map_PolylineTracked(ByVal polyline As GeoMap.polyline)
    Select Case Me.mlMouseAction
        Case geoMouseQueryLineBufferPick
            map.DrawPolyline polyline, IIf(Me.mlMouseAction = geoMouseQueryLineBufferPick,
            RGB(0, 0, 255), RGB(255, 0, 0))
            currentPolylinesCount = currentPolylinesCount + 1
            ReDim Preserve currentPolylines(0 To currentPolylinesCount - 1)
            Set currentPolylines(currentPolylinesCount - 1) = polyline
            If (map.KeyState(geoKeyShift) And geo_KeyDown) = 0 Then
                geoMouseDownHandler 2, 0, 0, 0
            End If
            ...
        End Select
    End Sub

```

在 MapFrm 的代码窗口中，加入代码：

```

Dim currentPolygons() As GeoMap.Polygon
Dim currentPolygonsCount As Long
Public m_bufferWidth As Long

Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal
x As Long, ByVal y As Long)
    Select Case Me.mlMouseAction
        Case geoMouseQueryLineBufferPick '线缓冲区查询
            DoLineBufferPick currentPolylines
            ...
        End Select
    End Sub

Private Sub DoLineBufferPick(polylinearray As Variant)
    Dim polygonarray
    '生成缓冲区

```

```

    polygonarray = map.CalcBufferOffPolyLines(polylinearray, m_bufferWidth, 12)
    QueryByPolygonArray polygonarray
    Dim i As Long
    For i = LBound(polygonarray) To UBound(polygonarray)
        map.DrawPolygon polygonarray(i), RGB(255, 0, 0)
    Next i
End Sub

```

```

Sub QueryByPolygonArray(polygonarray As Variant) '多边形查询
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim fea As GeoMap.Feature
    Dim ws As GeoMap.Workspace
    Dim querymode As QueryModeConstants
    Dim isWSS As Boolean
    Dim savename As String

    If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Sub
    querymode = IIf(mainFrm.barMain.Tools("mnuWholeContain").Checked,
geoWholeContain, geoPartContain)
    map.QueryObjectsByPolygon polygonarray(0), fea, querymode, savename
    If savename <> vbNullString Then QueryResultOpen savename
End Sub

```

线缓冲区查询如图 5-4 所示。

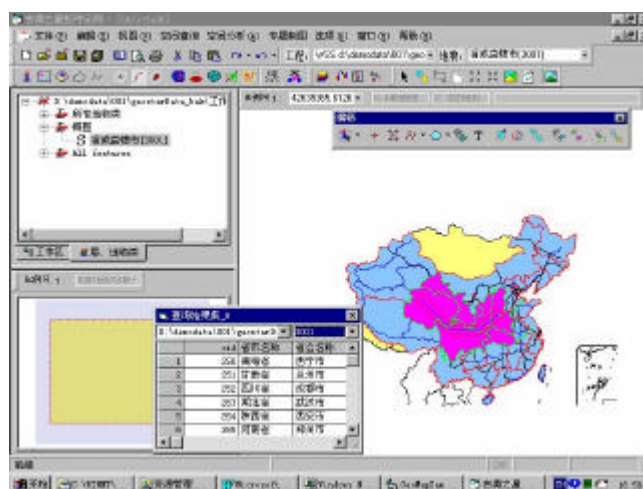


图 5-4 线缓冲区查询

提示：缓冲区查询的实质为多边形查询，先由几何图形生成其缓冲区（实际上其为多边形），然后调用 map.QueryObjectsByPolygon 方法，由多边形查询方式得到查询结果。点缓冲区查询、面缓冲区查询与线缓冲区查询实现的思路类似，请参考相关的部分。

4 . 清除查询结果

在 docFrm 的代码窗口中，加入代码：

```
Sub SetClearQuery()  
    mainmapFrm.DoClearQuery  
End Sub
```

在 MapFrm 的代码窗口中，加入代码：

```
Sub DoClearQuery()  
    map.HighlightClearAll  
    moldhighlightID = -1
```

```
End Sub
```

本节的具体代码请参考光盘上 5-Query。本节学习了利用 Geomap 控件实现常用的一些查询功能设计。下一节介绍专题图常用功能的设计。

第 6 章 专题图功能设计

专题图以简明、直观的形式表达一种或几种相关联的社会、经济、自然现象的统计数据，使统计数据的内涵可视化，易于分析和获取信息，长于表示现象的发展动态变化和发展规律。专题图也是 GIS 的重要组成部分，既可以直接表示空间数据的特性，同时也可以将空间查询或空间分析的结果以适当的图形方式表示出来，这将比数据、表格的表达更加简单、明了。本节内容包括创建、打开、关闭、修改、保存、更新专题图、增加图廓线和文字说明、图面配置等。

下面我们学习 GeoMap 的常用的专题图功能设计。

在本示例中，引入以下窗体：

SelectThematicMapFrm 选择专题图

6.1 功能实现：

1. 创建专题图

在 docFrm 的代码窗口中，加入代码：

```
Sub DoThemaCreate()      '生成专题图
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim ws As GeoMap.Workspace
    Dim fea As GeoMap.Feature
    Dim isWSS As Boolean

    ' 获取当前工作区用来生成专题图的面状地物类
    If Not GetCurrentFeature(wss, prj, ws, fea, isWSS) Then Exit Sub
    mainmapFrm.map.ThematicMapCreate ws, fea
End Sub
```

在 mapFrm 的代码窗口中，加入代码：

```
Dim WithEvents engine As GeoMap.geoengine 用于处理外部属性数据源

Private Sub Form_Load()
    '创建 geoengine 的实例
    Set engine = GeoMap.geoengine
End Sub

Private Sub engine_QueryDataSourceForFeature(ByVal Workspace As GeoMap.Workspace,
```

ByVal Feature As GeoMap.Feature, connectString As String, tableName As String)

Dim wss As GeoMap.WorkspaceSet

Dim prj As GeoMap.Project

Dim dsn As String

Select Case Workspace.OwnerType

Case geoWorkspaceSet '工作区集

Set wss = Me.map.WorkspaceSets(Workspace.OwnerPathName)

dsn = Workspace.WSDirectory & "WORKATTR"

dsn = ConvertString(dsn)

dsn = "ODBC;DSN=" & dsn & ";UID=;PWD=;SourceDB=" &

Workspace.WSDirectory & "WORKATTR;Driver=MicroSoft Visual FoxPro

Driver;SourceType=DBF"

Case geoProject '工程

Set prj = Me.map.Projects(Workspace.OwnerPathName)

dsn = prj.PRJDirectory & "PRJATTR"

dsn = ConvertString(dsn)

dsn = "ODBC;DSN=" & dsn & ";UID=;PWD=;SourceDB=" & prj.PRJDirectory &

"PRJATTR\" & prj.PRJName & ".DBC;Driver=MicroSoft Visual FoxPro

Driver;SourceType=DBC"

End Select

connectString = dsn 查询属性表的相应数据源的 ODBC 连接串。

tableName = Feature.DBTableName

End Sub

提示：在 geoengine_QueryDataSourceForFeature 事件中获得用以生成专题图的地物类的属性表，也可以连接其它数据库类型的属性表，用来生成专题图。调用方法 map.ThematicMapCreate 将自动弹出一系列对话框，供用户进行设置，在生成向导的指导下，一步步地生成专题图。（如图 6-1、6-2、6-3、6-4、6-5 所示）

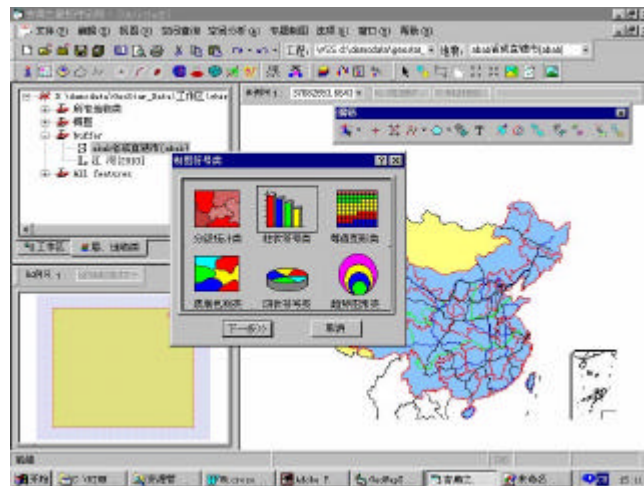


图 6-1

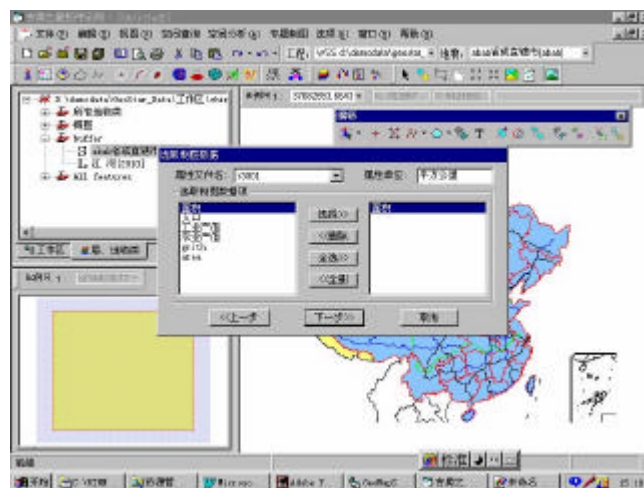


图 6-2

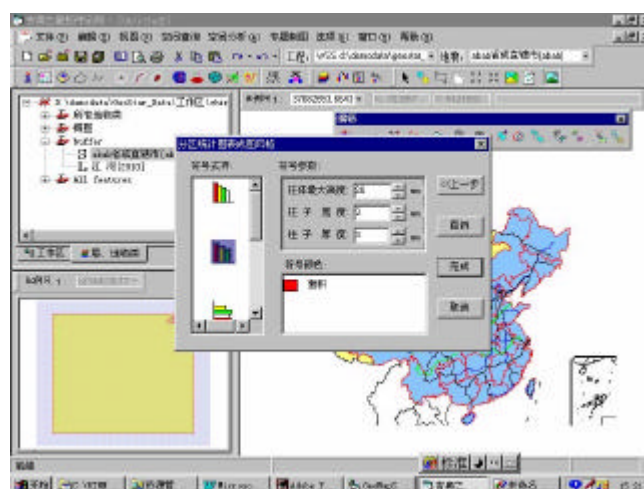


图 6-3

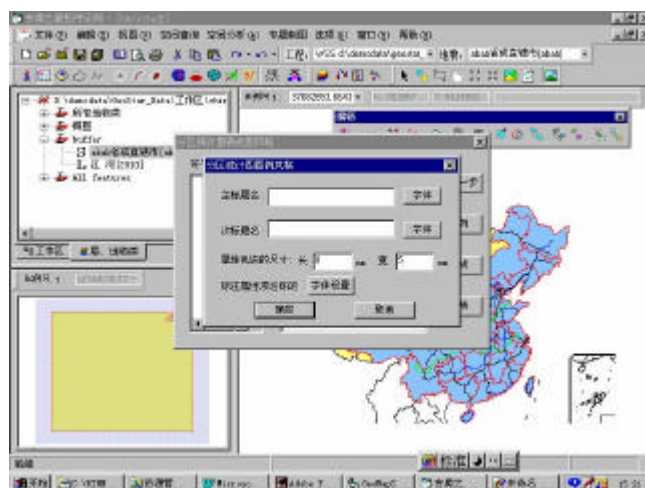


图 6-4

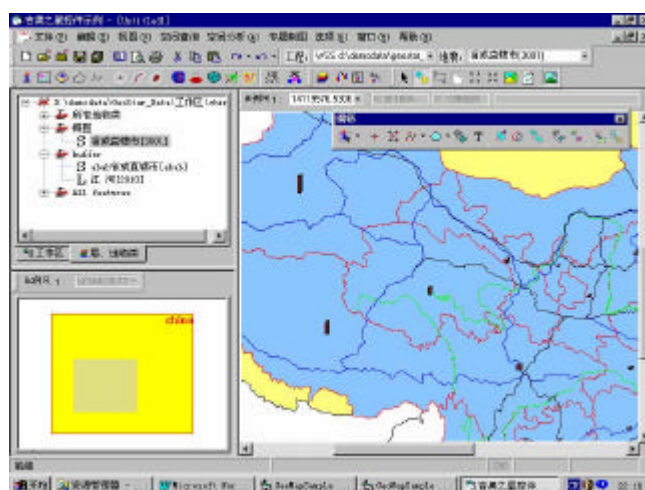


图 6-5

2. 打开专题图

在 docFrm的代码窗口中，加入代码：

Sub DoThemaOpen() '打开专题图

Dim wss As GeoMap.WorkspaceSet

Dim prj As GeoMap.Project

Dim ws As GeoMap.Workspace

Dim fea As GeoMap.Feature

Dim isWSS As Boolean

'从公共对话框中获取专题图文件名

mainmapFrm.map.ThematicMapOpen Me.CommonDlg.FileName

Dim mapNameArray() As String

ReDim mapNameArray(0 To mainmapFrm.map.ThematicMapCount - 1)

```

Dim i As Integer
For i = 0 To mainmapFrm.map.ThematicMapCount - 1
    mapNameArray(i) = mainmapFrm.map.ThematicMapNames(i)
Next i
mainmapFrm.map.ThematicMapRefresh ws, mapNameArray '
End Sub

```

提示：打开专题图之前，必须先打开生成该专题图的工作区。一个工作区数据可以生成多个专题图，因此，在打开、关闭、保存专题图时，要考虑到这一点。

3．关闭专题图

在 docFrm 的代码窗口中，加入代码：

```

Sub DoThemaClose()
    Dim mapNum As Integer
    Dim frm1 As New SelectThematicMapFrm
    mapNum = mainmapFrm.map.ThematicMapCount
    Dim mapName As String
    Dim intI As Integer
    For intI = 0 To mapNum - 1
        frm1.MapNameList.AddItem mainmapFrm.map.ThematicMapNames(intI)
    Next intI
    frm1.Show vbModal
    For intI = LBound(frm1.mapNames) To UBound(frm1.mapNames)
        mainmapFrm.map.ThematicMapClose frm1.mapNames(intI)
    Next intI
    mainmapFrm.map.Refresh
End Sub

```

提示：打开和关闭专题图所需的参数不一致。必须注意。

4．保存专题图

在 docFrm 的代码窗口中，加入代码：

```

Sub DoThemaSave()
    Dim mapNum As Integer
    Dim frm1 As New SelectThematicMapFrm
    mapNum = mainmapFrm.map.ThematicMapCount
    Dim mapName As String
    Dim intI As Integer
    For intI = 0 To mapNum - 1

```

```

        frm1.MapNameList.AddItem mainmapFrm.map.ThematicMapNames(intI)
    Next intI
    frm1.Show vbModal
    Me.CommonDlg.ShowSave
    Dim selectedCount As Integer
    selectedCount = UBound(frm1.mapNames) - LBound(frm1.mapNames)
    Dim mapNames() As String
    ReDim mapNames(selectedCount)
    For intI = LBound(frm1.mapNames) To UBound(frm1.mapNames)
        mapNames(intI) = frm1.mapNames(intI)
    Next intI
    mainmapFrm.map.ThematicMapSave mapNames, Me.CommonDlg.FileName
End Sub

```

5. 修改专题图

在 docFrm 的代码窗口中，加入代码：

```

Sub DoThemaModify()    '修改专题图
    mainmapFrm.map.ThematicMapModify
    mainmapFrm.map.Refresh
End Sub

```

提示：调用 Geomap.ThematicMapModify 方法，将自动弹出一系列对话框，用以修改专题图的选项。如图 6-6、6-7、6-8 所示。

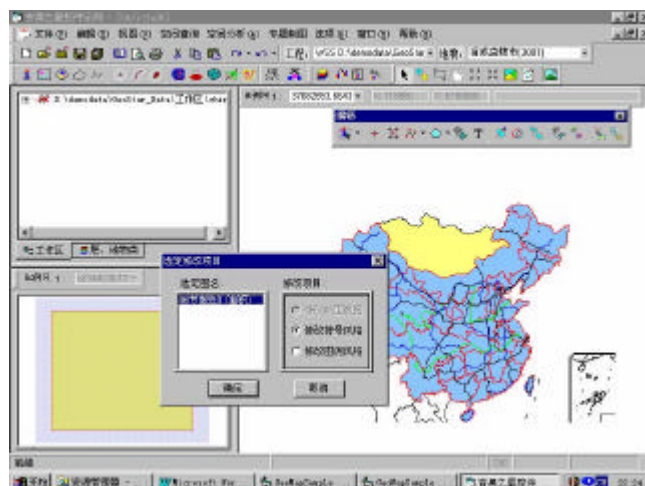


图 6-6

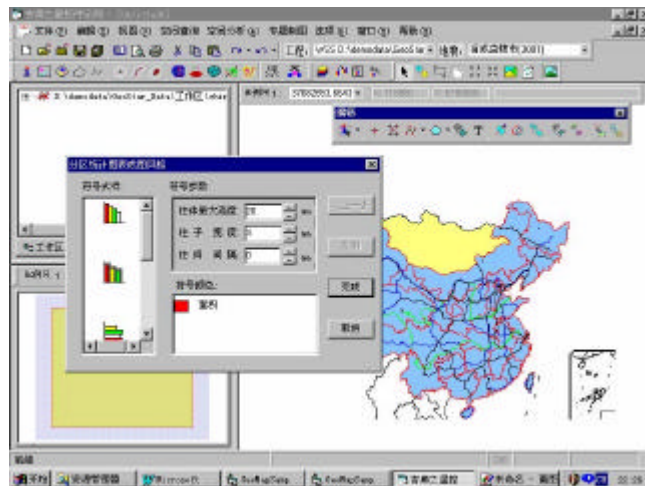


图 6-7

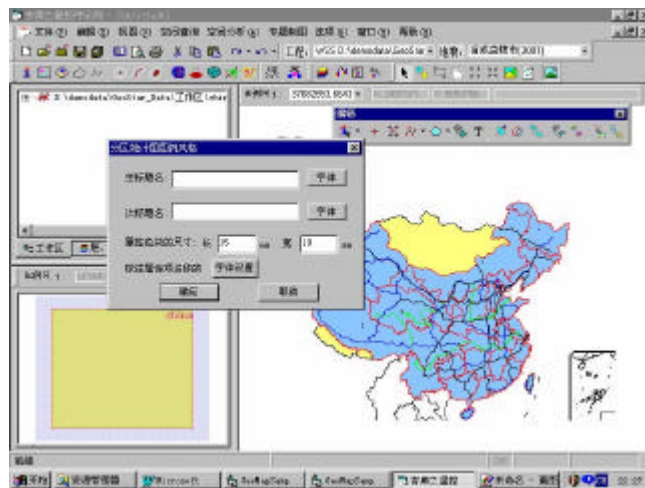


图 6-8

6. 图面配置

在 docFrm 的代码窗口中，加入代码：

```
Sub DoThematicMapLayout() '图面配置
    SetMouseAction geoMousethemelayout
End Sub
```

```
Sub SetMouseAction(ByVal action As MouseActionConstants)
    Select Case Me.m1MouseAction
        ...
        Case geoMousethemelayout
            mainmapFrm.map.MousePointer = geoDefault
        ...
    End Select
End Sub
```

在 mapFrm的代码窗口中，加入代码：

```
Private Sub geoMouseDownHandler(ByVal Button As Integer, ByVal Shift As Integer, ByVal
x As Long, ByVal y As Long)
    Select Case Me.mlMouseAction
        ...
        Case geoMousethemelayout    '图面配置
            map.TrackThematicMapLayout
        ...
    End Select
End Sub
```

提示：图面配置就是将不恰当位置的专题图符号、图例等用鼠标移动之后，使图面协调。因此，首先还是鼠标按下的处理。发生鼠标按下的事件。

7．添加图廓线

在 docFrm 的代码窗口中，加入代码：

```
Sub DoThematicMapOutline()    '添加图廓线
    If mainmapFrm.map.TrackThematicMapOutline Then mainmapFrm.map.MousePointer
= geoCross
End Sub

Private Sub map_ThematicMapOutlineTracked()
    map.MousePointer = Me.mlOldMousePointer
    Me.mlMouseAction = Me.mlOldMouseAction
End Sub
```

8．文字说明

在 docFrm 的代码窗口中，加入代码：

```
Sub DoThematicMapDecorate()    '图廓整饰
    If mainmapFrm.map.TrackThematicMapText Then mainmapFrm.map.MousePointer =
geoIBeam
End Sub
```

9．图层控制

在 docFrm 的代码窗口中，加入代码：

```
Sub DoThematicMapSettings()    '图层控制
    mainmapFrm.map.ThematicMapSettings
```


End Sub

提示：调用 map.ThematicMapSettings 方法，系统将自动弹出对话框，如图 6-9 所示。

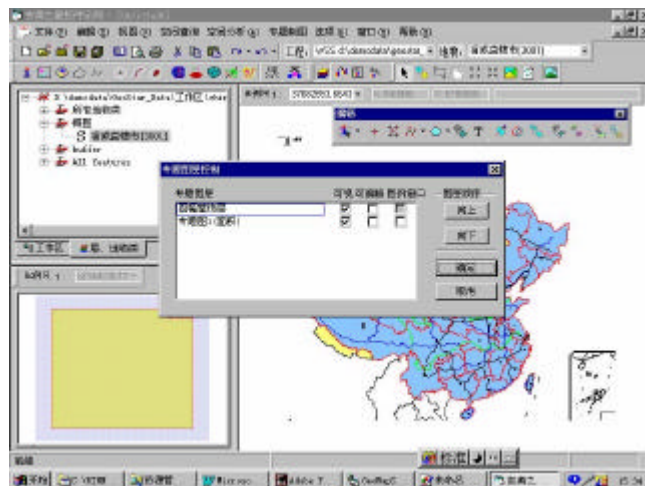


图 6-9

10 . 更新专题图

在 docFrm 的代码窗口中，加入代码：

```
Sub DoThematicMapRefresh()
    Dim wss As GeoMap.WorkspaceSet
    Dim prj As GeoMap.Project
    Dim ws As GeoMap.Workspace
    Dim fea As GeoMap.Feature
    Dim isWSS As Boolean

    Dim mapNameArray() As String
    ReDim mapNameArray(0 To mainmapFrm.map.ThematicMapCount - 1)
    Dim i As Integer
    For i = 0 To mainmapFrm.map.ThematicMapCount - 1
        mapNameArray(i) = mainmapFrm.map.ThematicMapNames(i)
    Next i
    mainmapFrm.map.ThematicMapRefresh ws, mapNameArray
End Sub
```

本示例具体的代码可参考光盘上 6-Thema。本节学习了利用 Geomap 控件实现常用的一些专题制图功能，可以用来分析有关的 GIS 信息，辅助决策。下一节我们学习空间分析的功能设计。

第 7 章 空间分析功能设计

本节主要学习缓冲区分析功能的实现。根据生成缓冲区数据的不同，分为按地物类或查询集进行缓冲区分析。

在本示例中，引入以下窗体：

frmBufferOut 用于生成缓冲区，进行缓冲区分析

7.1 具体功能实现：

1.1 缓冲区分析

在 docFrm 的代码窗口中，加入代码：

```
Sub DoBufferAnalysis()    '缓冲区分析
    Dim frmBuffer As frmBufferOut
    Set frmBuffer = New frmBufferOut
    Set frmBuffer.map = mainmapFrm.map
    frmBuffer.Show vbModal, mainFrm

    Dim feas(0) As GeoMap.Feature
    Set frmBuffer.map = Nothing
    If frmBuffer.returnOK Then
        If frmBuffer.bFeature Then
            Set feas(0) = frmBuffer.bufferfeature
            mainmapFrm.map.CalcBufferOfFeatures feas, mainmapFrm.m_bufferWidth, 8,
            frmBuffer.returnWS, frmBuffer.returnFea
        Else
            '由查询集进行缓冲区分析
            If Not frmBuffer.bufferQueryResult Is Nothing Then
                mainmapFrm.map.CalcBufferOfQueryResult frmBuffer.bufferQueryResult,
                mainmapFrm.m_bufferWidth, 8, frmBuffer.returnWS, frmBuffer.returnFea
            End If
        End If
    End If
    mainmapFrm.map.Refresh    刷新
End Sub
```

缓冲区分析的效果如图 7-1、图 7-2 所示

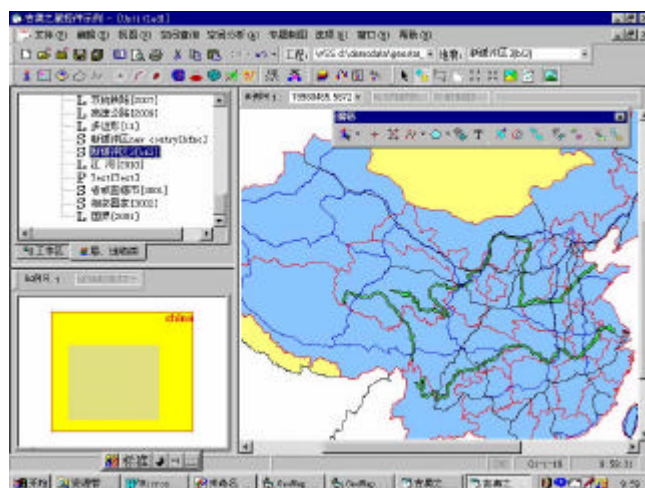


图 7-1 根据地物类进行缓冲区分析

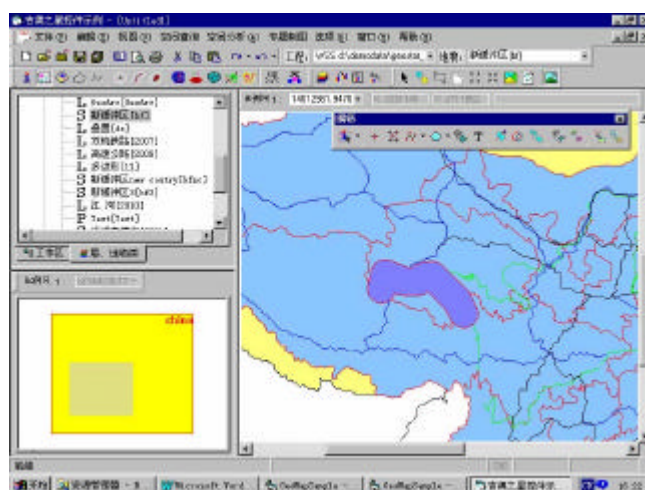


图 7-2 根据查询集进行缓冲区分析

本示例具体的代码请参考光盘上的示例 7-Analysis。下一节我们学习 GeoMap 功能实现的有关设置的实现。

第 8 章 参数设置功能

在本示例中要进行一系列的参数设置，影响各种操作不同的效果。

8.1 是否改变原始对象

在 mainFrm 的代码窗口中，加入代码：

```
Tool.Checked = Not Tool.Checked  
changeOriginal = Tool.Checked
```

8.2 设置颜色

8.2.1 设置处于编辑状态的线颜色

在 docFrm 的代码窗口中，加入代码：

```
Sub SetEditSelectObjectsLineColor() '处于编辑状态的线颜色  
    ...  
    '从颜色对话框中获取处于编辑状态的线颜色  
    CommonDlg.ShowColor  
    mainmapFrm.map.EditSelectObjectsLineColor = CommonDlg.Color  
End Sub
```

8.2.2 处于编辑状态的面填充颜色：

在 docFrm 的代码窗口中，加入代码：

```
Sub SetEditSelectObjectsFillColor() '处于编辑状态的面填充颜色  
    ...  
    '从颜色对话框中获取处于编辑状态的面填充颜色  
    CommonDlg.ShowColor  
    mainmapFrm.map.EditSelectObjectsFillColor = CommonDlg.Color  
End Sub
```

提示：本示例要进行一系列的颜色设置，如主图背景色、索引图背景色、索引图选取窗颜色、选中对象内点颜色、选中内点颜色、选中注记颜色等，与上面处于编辑状态的线状、面状颜色处理类似，都是从颜色对话框中获取需要的颜色。

8.3 地图符号化

在 docFrm 的代码窗口中，加入代码：

```
Sub SetSymbolization() '地图符号化显示
```

```

    mainmapFrm.map.symbolization = IIf(mainmapFrm.map.symbolization, False, True)
  If Not feaFrm Is Nothing Then
    feaFrm.SetSymbolization mainmapFrm.map.symbolization
  End If
End Sub

```

8.4 . 地图内容随图缩放

在 docFrm 的代码窗口中，加入代码：

```

Sub SetGraphicsZooming() '地图内容随图放大
    mainmapFrm.map.GraphicsZooming = IIf(mainmapFrm.map.GraphicsZooming, False,
True)
End Sub

```

8.5 . 注记显示

在 docFrm 的代码窗口中，加入代码：

```

Sub SetAnnotationVisible() '地图注记显示
    mainmapFrm.map.AnnotationVisible = IIf(mainmapFrm.map.AnnotationVisible, False,
True)
End Sub

```

8.6 . 设置查询容差

在 docFrm 的代码窗口中，加入代码：

```

Sub SetPickTolerance() '设置查询容差
    Dim str As String
    '从输入对话框获取查询容差
    str = InputBox("请输入查询容差: ", , mainmapFrm.map.PickTolerance)
    mainmapFrm.map.PickTolerance = CSng(str)
End Sub

```

8.7 . 设置缓冲区的宽度

在 docFrm 的代码窗口中，加入代码：

```

Sub SetBufferWidth() '设置缓冲查询宽度
    Dim str As String
    ...
    '从输入对话框获取缓冲查询宽度
    str = InputBox("请输入缓冲查询宽度: ", , mainmapFrm.m_bufferWidth)
    mainmapFrm.m_bufferWidth = CSng(str)
End Sub

```

提示：此缓冲查询宽度用于缓冲区查询和缓冲区分析时生成缓冲区，其单位是库坐标。当然，也可以通过转化，设置为地图坐标。

8 8 . 节（内）点捕捉

在 docFrm 的代码窗口中，加入代码：

```
Sub SetEditCursorSnappable()  
    '是否进行节点捕捉  
    mainmapFrm.map.EditCursorSnappable = Not  
mainmapFrm.map.EditCursorSnappable  
End Sub
```

8 9 . 是否完全包含

在 mainFrm 的代码窗口中，加入代码：

```
barMain.Tools("mnuWholeContain").Checked = Not  
barMain.Tools("mnuWholeContain").Checked  
barMain.Tools("mnuPartContain").Checked = Not  
barMain.Tools("mnuPartContain").Checked
```

8.10 . 部分包含

在 mainFrm 的代码窗口中，加入代码：

```
barMain.Tools("mnuWholeContain").Checked = Not  
barMain.Tools("mnuWholeContain").Checked  
barMain.Tools("mnuPartContain").Checked = Not  
barMain.Tools("mnuPartContain").Checked
```

提示：几何选择、空间查询都涉及操作方式是完全包含和部分包含的问题，不同的选项将会有不同的结果。

附录 如何使用控件 ActiveBar 创建工具条

步骤如下：

- 1) 将控件 ActiveBar 添加到你的工程中（图 1）

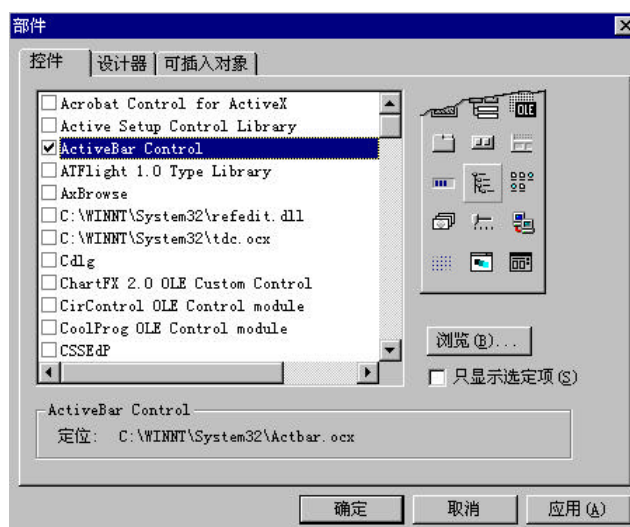


图 1

- 2) 将其图标放置于你的窗体上（图 2）

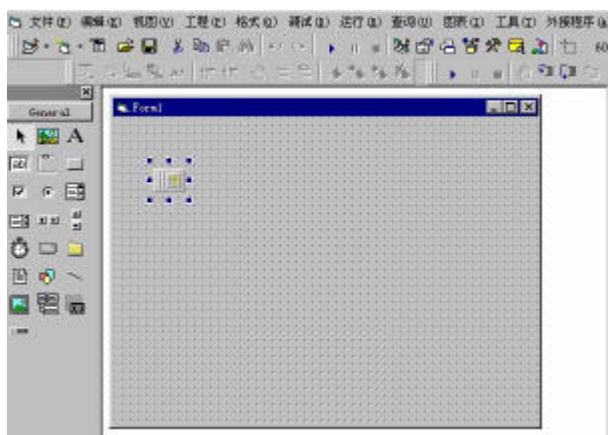


图 2

- 3) 用鼠标右键单击控件的图表，弹出上下文菜单。选取“Designer”菜单项，打开设计窗口。（图 3、图 4）

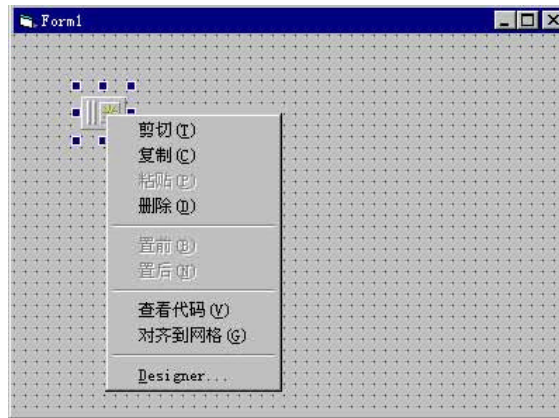


图 3

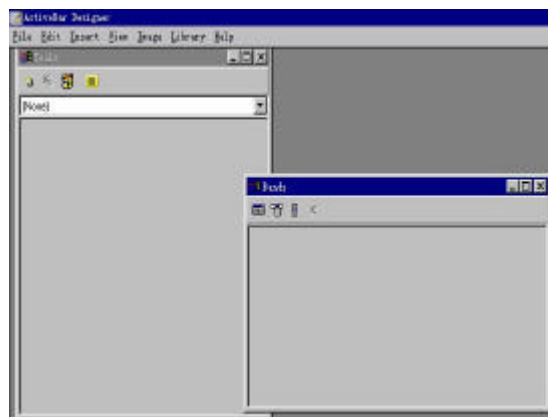


图 4

4) 首先, 创建 category。单击 categories 图标, 弹出 "Manage Categories" 对话框。添加 "Category 1"。(图 5)



图 5

5) 单击 tool 图标, 弹出对话框, 填写属性。可以用手工绘制或抓图工具获得图标。(图 6)

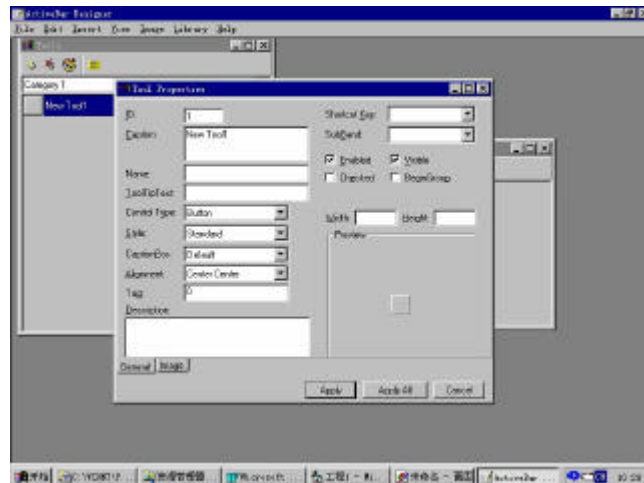


图 6

6) 手工将 "New" tool 拖到 Band1 窗口中, (图 7)

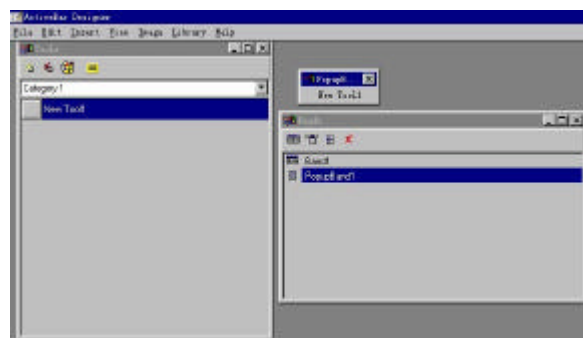


图 7

7) 用鼠标右键弹出上下文菜单, 选取 "Band Properties", 设置其属性。(图 8、图 9)

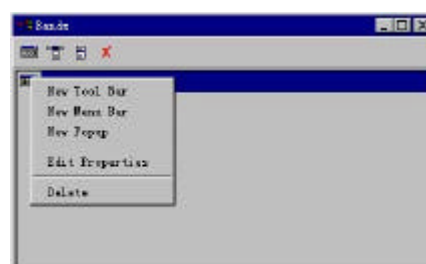


图 8



图 9

8) 关闭控件的设计窗口。