

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。

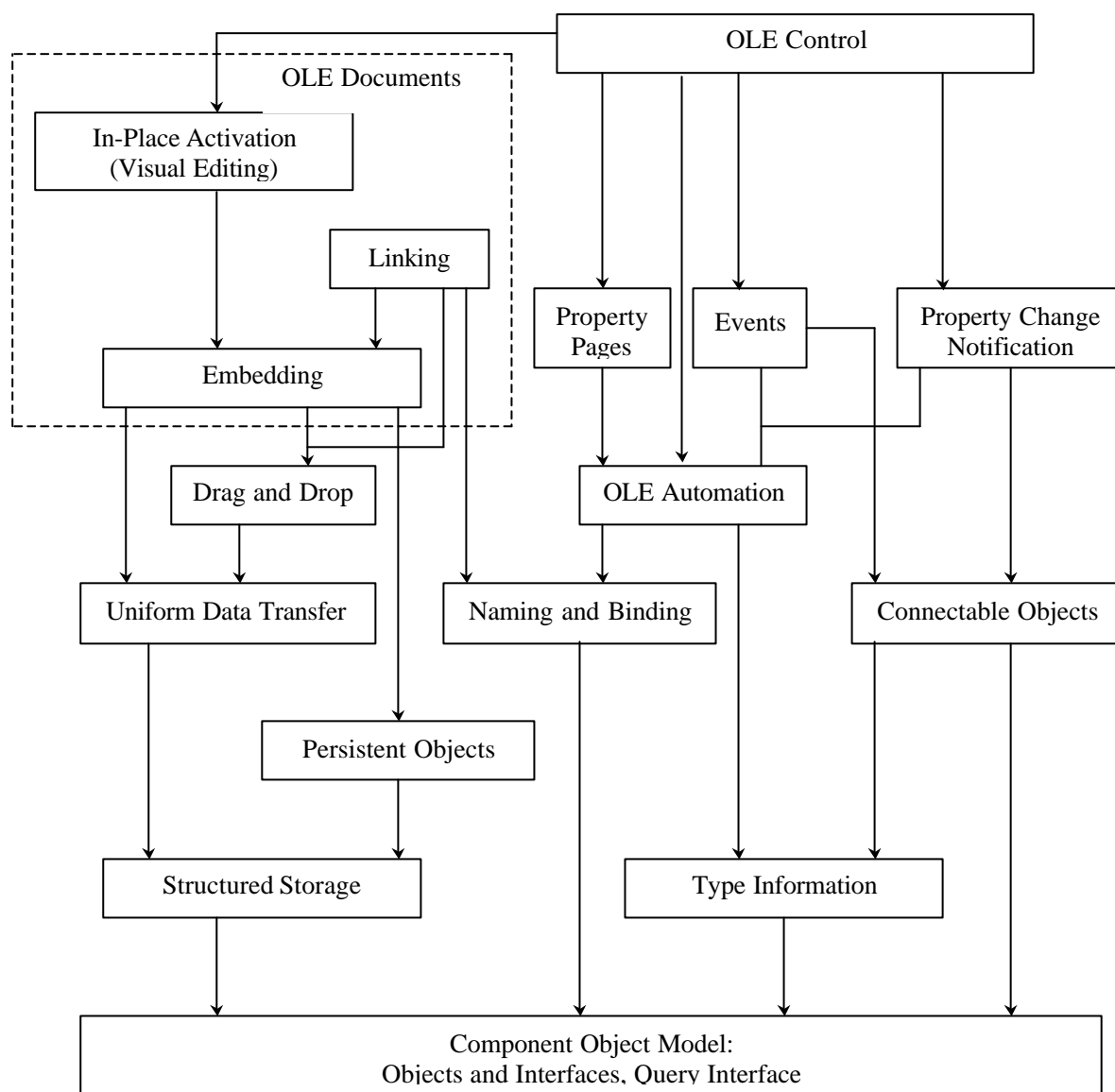


图 2-1 OLE 技术构成

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。

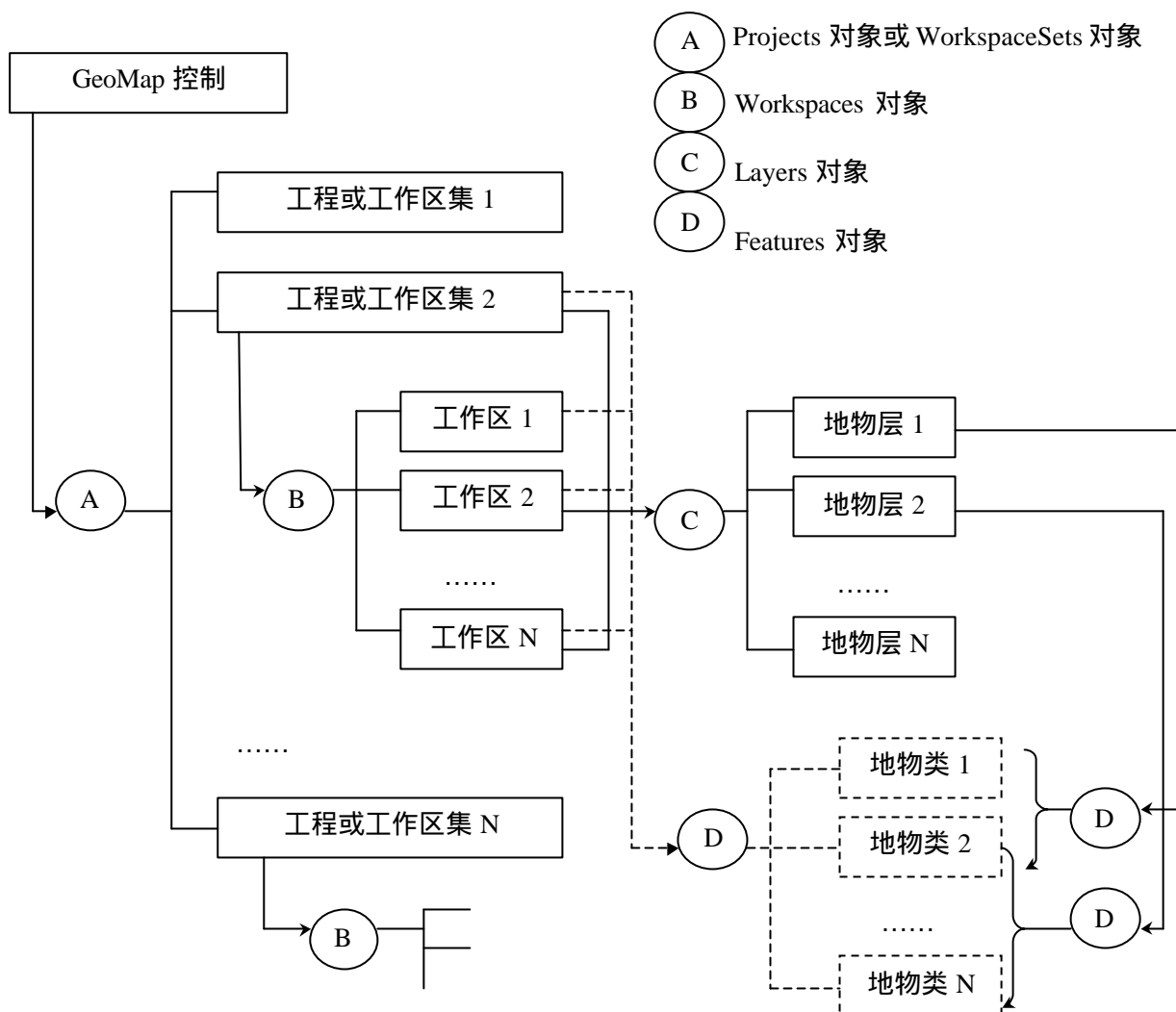


图 3-1 GeoMap 数据组织结构

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。

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

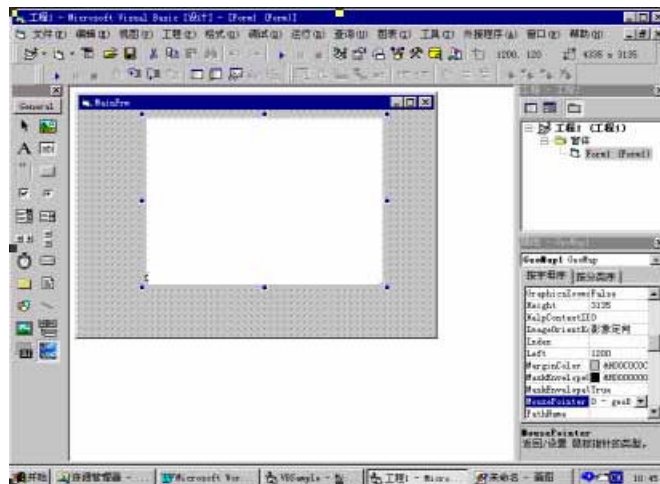


图 4-1-3

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：

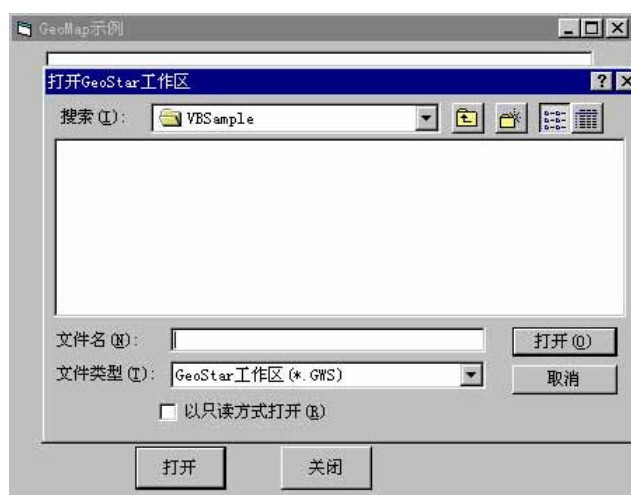


图 4-1-4

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：

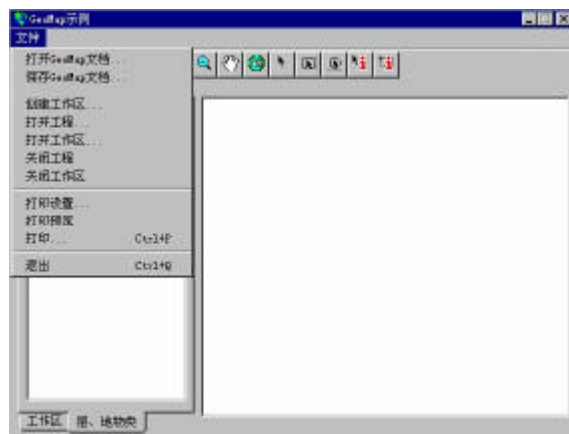


图 4-2-1

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：

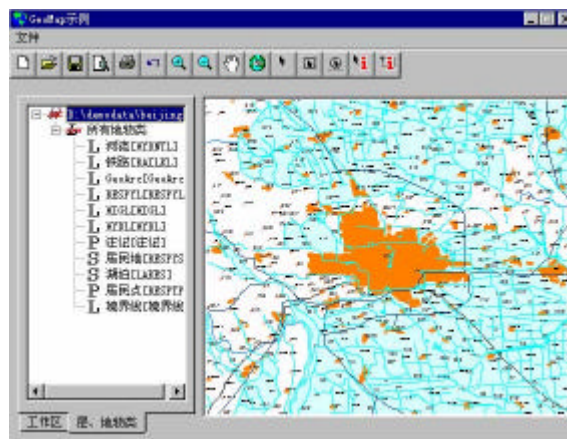


图 4-2-2

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 mFilecreate_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

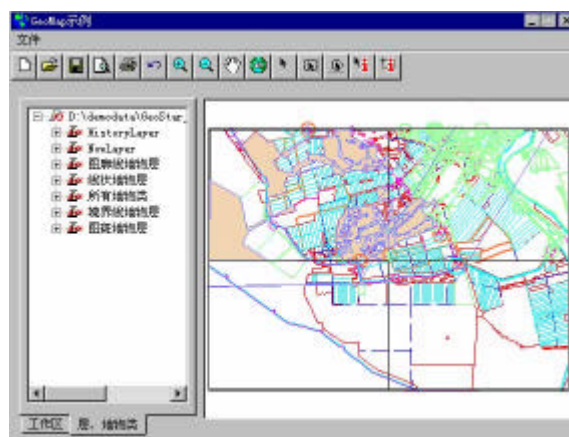


图 4-2-4

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

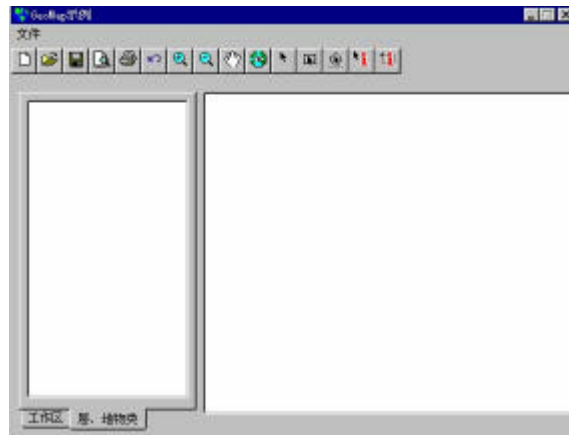


图 4-2-5

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。

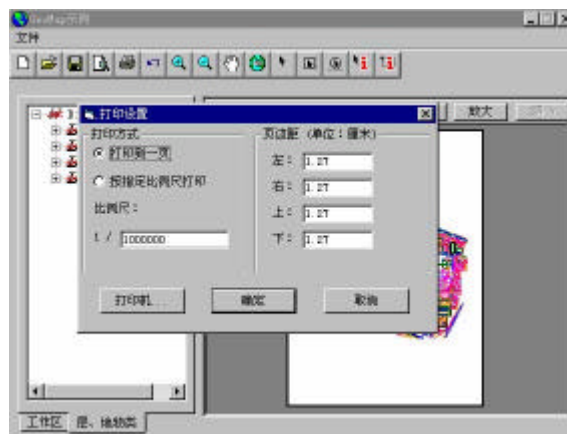


图 4-2-6

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。

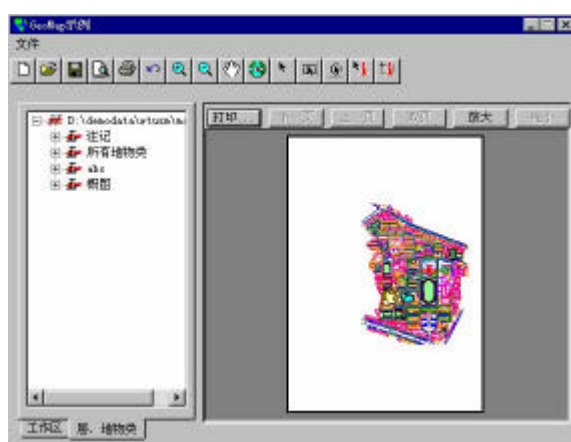


图 4-2-7

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。

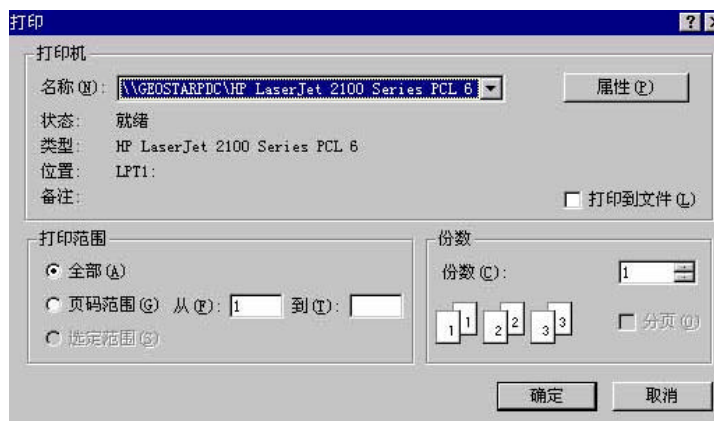


图 4-2-8

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：

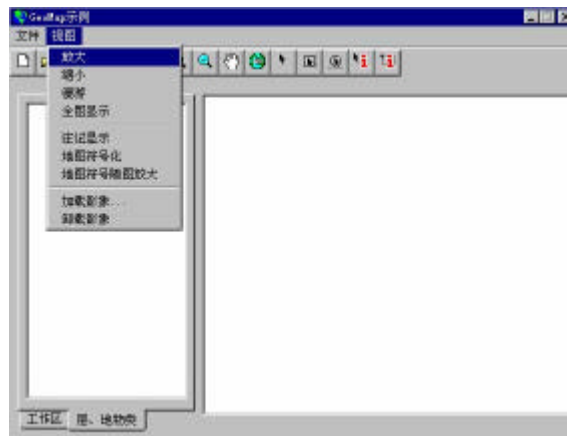


图 4-3-1

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。

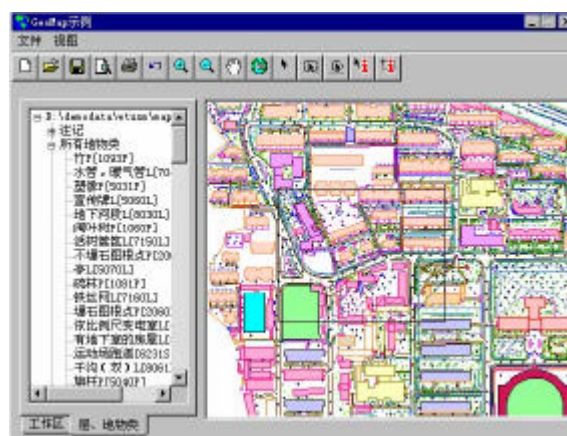


图 4-3-2

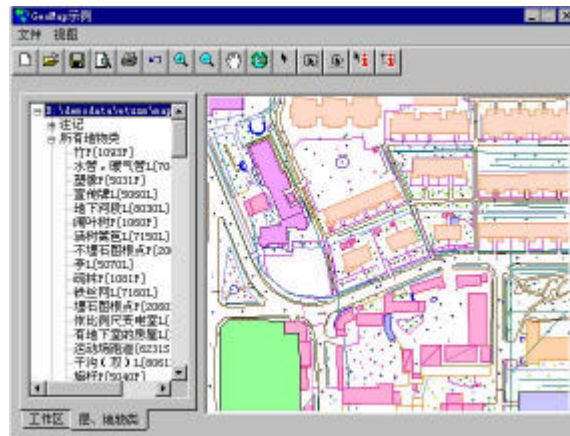


图 4-3-3

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。

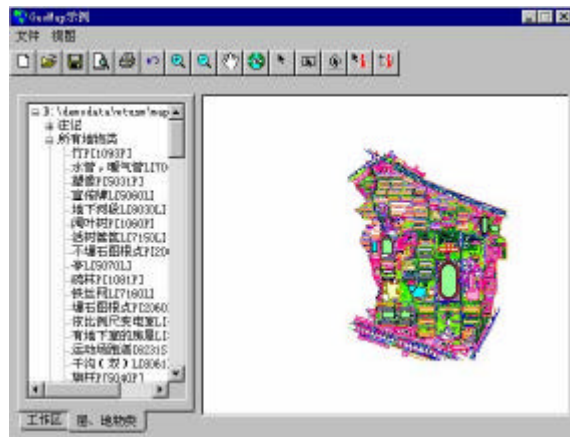


图 4-3-4

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。

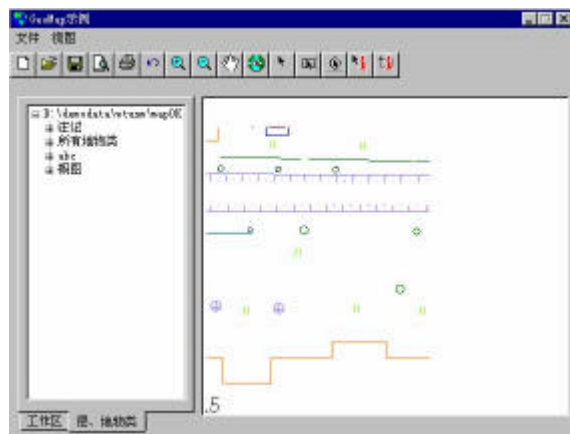


图 4-3-5

第四步：在【视图|注记】菜单中写入如下代码；

```
Private Sub mnuMark_Click()
    '显示注记
    Map.AnnotationVisible = IIf(Map.AnnotationVisible, False, True)
End Sub
```

运行示例 Form，单击【视图|注记】菜单项，在 Geomap 控件窗口地图注记显示出来，如图 4-3-6 所示。

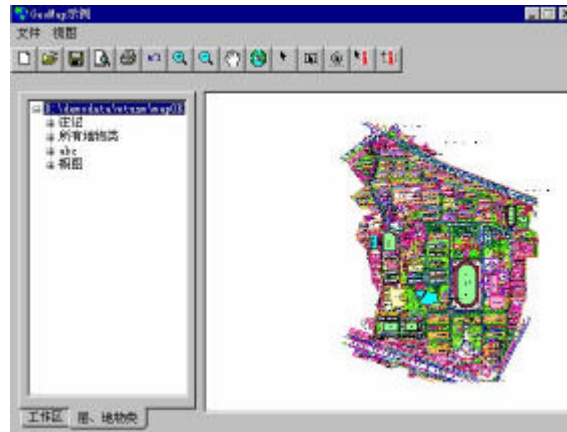


图 4-3-6

第四步：在【视图|随图放大】菜单中写入如下代码；

```
Private Sub mnuGraphicZoom_Click()
    '随图放大
    Map.GraphicsZooming = IIf(Map.GraphicsZooming, False, True)
End Sub
```

运行示例 Form，单击【视图|随图放大】菜单项，在 Geomap 控件窗口地图内容随图放大显示出来，如图 4-3-7 所示。

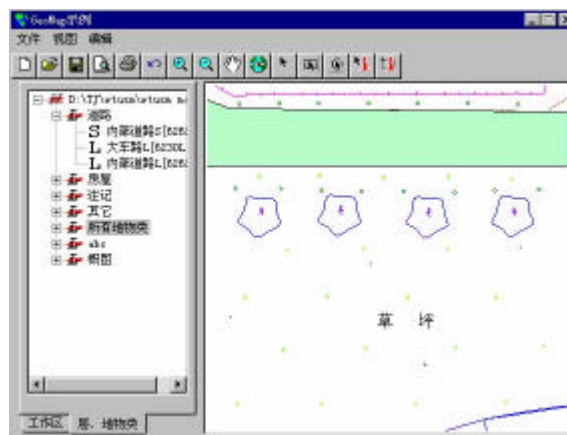


图 4-3-7

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。

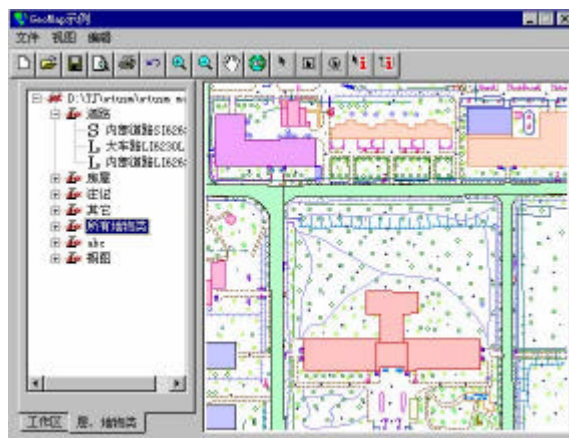


图 4-3-8

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。）

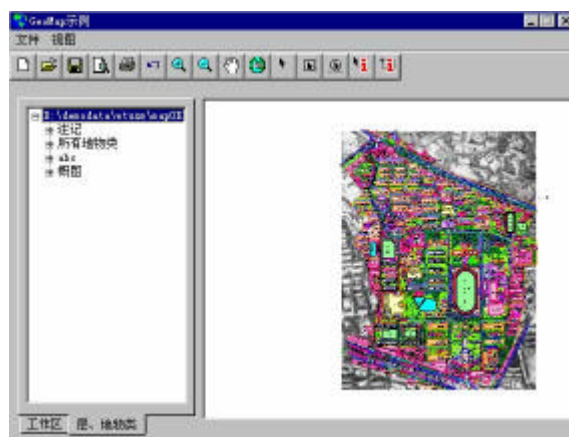


图 4-3-8

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。

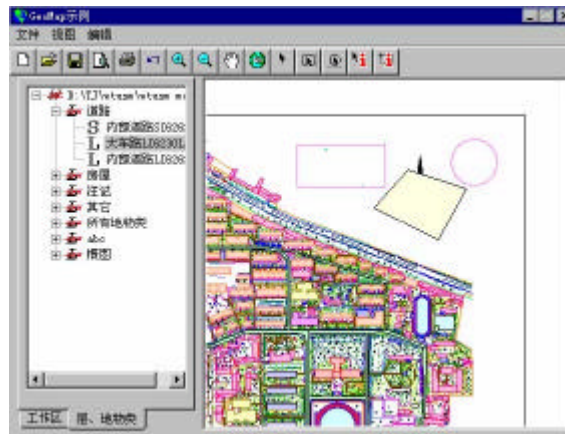


图 4-4-5

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。

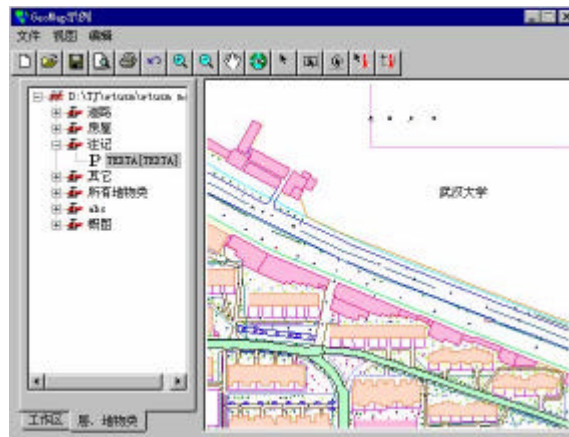


图 4-4-6

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。

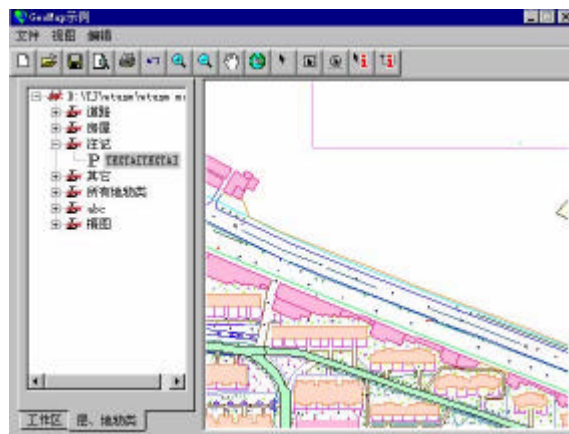


图 4-4-7

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）。

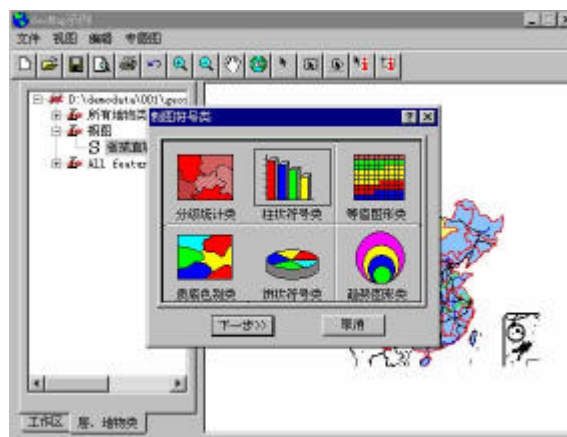


图 4-5-2

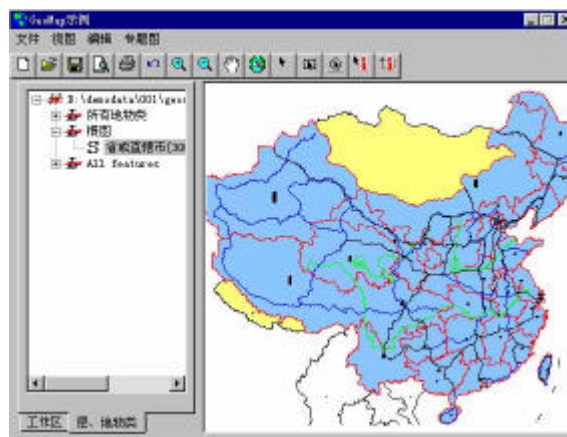


图 4-5-3

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。

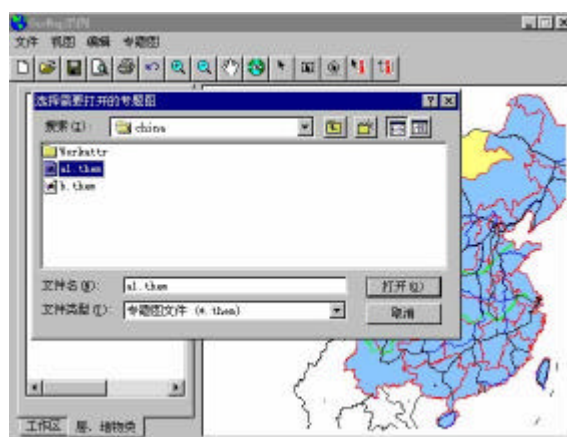


图 4-5-4

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。

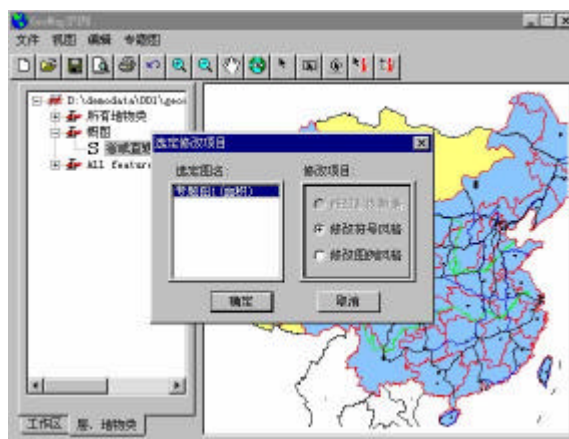


图 4-5-5



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：

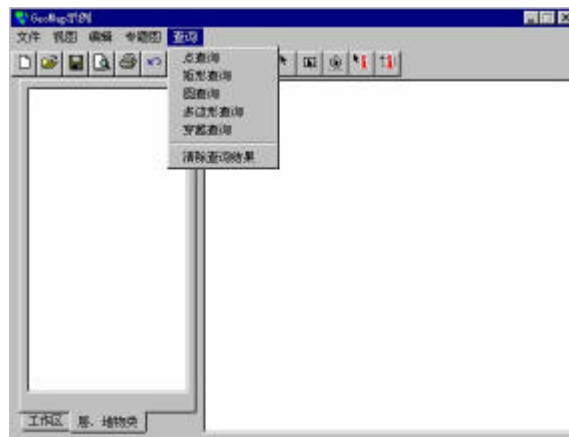


图 4-6-1

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

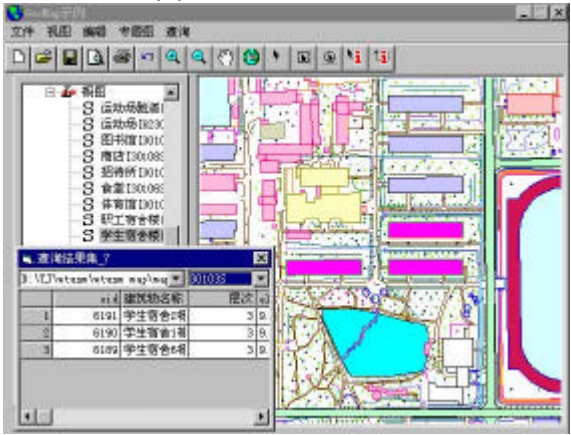


图 4-6-4

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

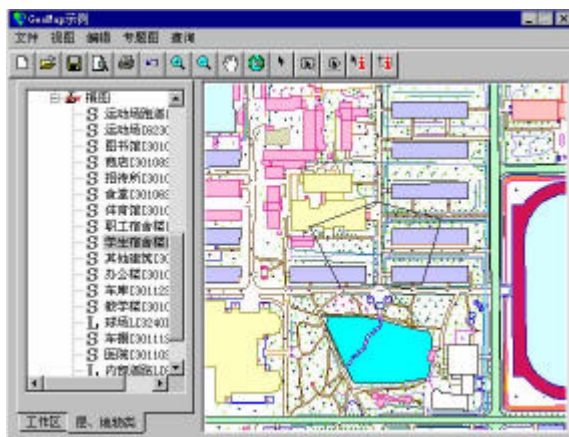


图 4-6-5

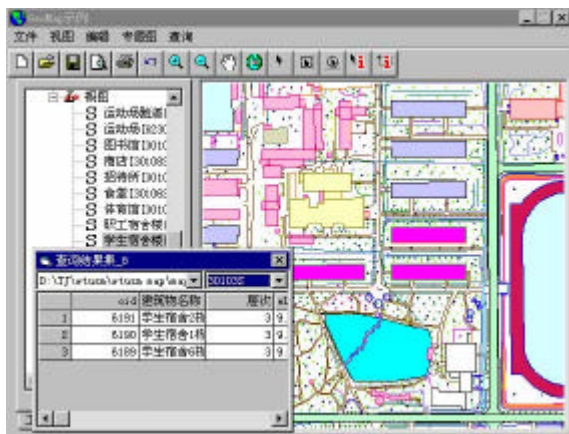


图 4-6-6

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

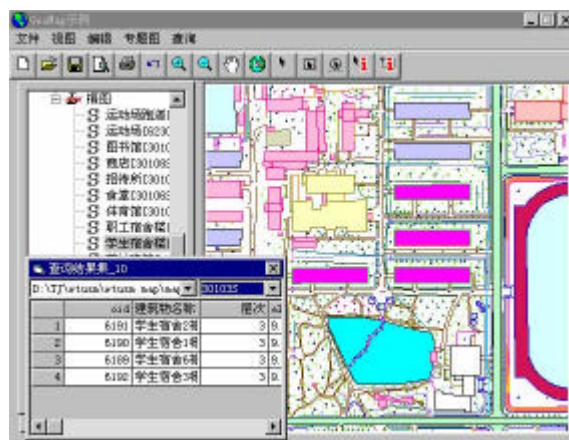


图 4-6-8

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。

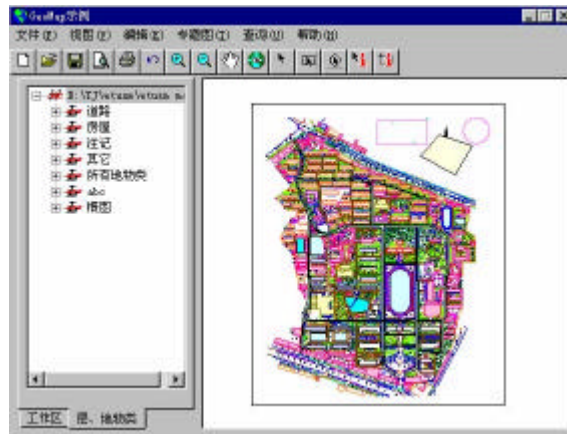


图 4-7-1

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。