利用OLE自动化技术实现MapInfo与VC++的集成

何晶 陈西宏 吕辉 (解放军空军工程大学导弹学院 713800)

摘要：本文主要介绍了怎样利用OLE自动化技术来实现MapInfo与VC++的集成。概要阐述了OLE自动化技术及MapInfo与VC++对这一技术的支持，详尽介绍了在VC++6.0中调用MapInfo的具体实现过程。

关键词：地址信息系统 OLE自动化 MapInfo VC++

随着社会信息化的飞速发展，地理信息系统(GIS)以地理空间数据库为基础，在计算机硬、软件环境的支持下，对空间相关数据进行采集、管理、操作、分析、模拟和显示，适时提供空间和动态的地理信息，为决策服务的一类信息系统已越来越受到世人瞩目。MapInfo是由美国MapInfo公司推出的地理系统开发平台，由于其简单易学，功能强大且可以和普通的关系数据库相连，已成为桌面地图信息系统的代表。尤其是它具有对象链接与嵌入功能(OLE)，允许Visual Basic、PowerBuilder、Visual C++等把MapInfo作为一个对象加以调用，即支持OLE自动化技术，从而对在其他环境下编写的应用程序提供了良好的帮助，具有广泛的应用前景。Visual C++以其优良的面向对象特性，在与MapInfo的集成应用方面有着得天独厚的优势。本文就如何利用OLE自动化技术实现MapInfo与VC++6.0的集成作一简单介绍。

一、OLE自动化技术概述

OLE(Object Linking and Embedding)是Microsoft公司为解决Windows下应用程序间的通信问题而提出的。但是到了今天，OLE已演变为一种协议或规范，成为软构件集成技术的基础。OLE自动化(OLE Automation)技术是OLE2.0新增加的技术，它使用户通过编程在一个应用程序中控制另一个应用程序的对象，从而实现了应用程序级别的可重用。

在OLE自动化这一技术领域中，由应用程序或OLE编程工具所展现的对象称为OLE自动化对象，访问操作并控制该对象的应用程序或OLE编程工具则称为OLE自动化控制器，展现这些对象的应用程序则称为OLE自动化服务器。OLE自动化服务器展现的自动化对象使通过编程方法操纵应用程序成为可能。通过使用OLE自动化，能够完成以下任务：

- 创建OLE自动化服务器
- 创建和操纵OLE自动化对象
- 创建OLE自动化控制器

OLE自动化服务器对象是编程的，其可编程能力主要是通过方法和属性这两个类的概念及两种类型的OLE自动化对象成员来实现。其中“方法”可以是带有许多参数的函数，它定义了一个对象所能达到并能完成的行为，同时可以获取或设置数据。“属性”则是不带有形参的函数，并能存取或设置有关对象当前状态的信息。实际上，OLE自动化技术就是使任何对象展示出一组可以由其他代码激活的方法和属性，而自动化提供了对象描述中参数和属性的名称和类型的方法。

使用OLE带来的好处是，不同应用程序展现的对象在同一编程环境中都是可见的，展现的对象可以用任何一个支持OLE自动化的编程工具或宏语言来访问，因此系统集成者可以选择最合适的任务开发工具。

二、MapInfo与VC++对OLE自动化的支持

无论是MapInfo还是VC++6.0，它们都对OLE自动化技术提供了良好的支持，下面将分别作以介绍。在本文中，我们将主要以MapInfo作为OLE自动化服务器，VC++6.0作为OLE自动化控制器，在VC++中实现对MapInfo功能的调用。但是，MapInfo同时具有回调能力，

将VC++应用程序作为自动化服务器，当MapInfo自身发生变化时，调用VC++应用程序，通知自己的改变。

1. MapInfo提供的支持

(1)MapInfo提供的OLE自动化对象模型如表1所示:

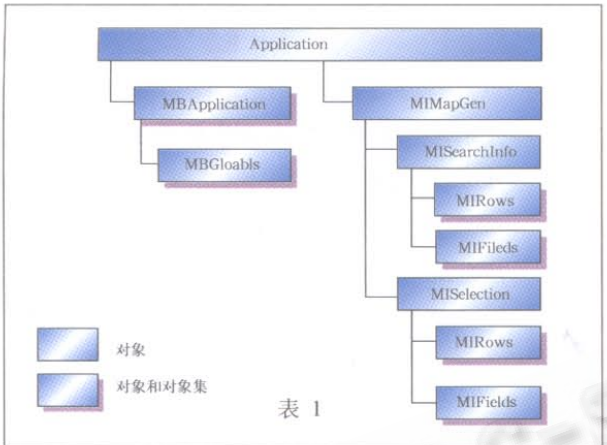

表 1

其中Application对象代表MapInfo的一个实例。在MBApplications集和MBGlobals集中，每一个对象分别代表一个正在运行的MapBasic应用程序和一个正在运行的MapBasic应用程序中定义的全局变量。MIMapGen对象主要用于MapInfo网络应用程序MISearchInfo对象代表查找的结果。MISelection对象代表MapInfo表。MIRows对象集中每一个对象代表表中的每一行，MIFields对象集中每一个对象代表表中的每一个域。

其中MapInfo的实例对象主要提供如下一些常用方法:

·Do(string): 向MapInfo出送命令字符串(MapBasic语句)，MapInfo将执行这些命令。

Evel(string): 应用程序通过此方法可获得MapInfo的执行结果

·RunMenuCommand(menuid): 应用程序可执行MapInfo的主菜单命令。

·RunCommand(string): 解释并执行MapBasic语句，与Do类似。

·SetCallBack(Idispatch): 将OLE自动化对象登记为一个接受器，当MapInfo窗口发生变化时，MapInfo将通知应用程序。将在后文详细叙述。

(2)MapInfo提供的回调功能。如果调用MapInfo的客户程序可作为自动化服务器，则当MapInfo发生变化时，例如地图窗口的变化，状态栏的改变或执行了某一菜单命令等等，MapInfo将自动的通知其派遣对象自动发生的改变。客户程序可根据给定的消息编制代码，相应的在应用程序窗口中作出响应和处理。

MapInfo提供的标准回调接口主要有

·SCODE SetStatusText(LPCTSTR lpszMessage)。当MapInfo的状态栏发生变化时调用，参数为新的状态栏字符串。

·SCODE WindowContentsChanged(Unsigned Long windowID)。当派遣地图窗口内容发生变化时MapInfo将调用WindowContentsChanged方法。参数是被改变的窗口的ID值。

2. VC++对OLE自动化的支持

Visual C++集成的MFC框架、Class Wizard和App Wizard都提供对OLE自动化的扩展支持，这些工具处理了创建自动化服务器和自动化控制器的大部分例行事务，极大的方便了编程者。

MFC框架对OLE自动化支持的基本机制是派遣映射表(Dispatch Map)。派遣映射表是一个宏集，用于扩展需要暴露属性和方法的调用。它给出了对象的函数和属性的内部名称和外部名，以及函数的参数数据类型和属性的数据类型。一个典型的派遣映射表如下所示:

```
BEGIN_DISPATCH_MAP
(CmyServerDoc,ColeServerDoc)
    //{{AFX_DISPATCH_MAP (CmyServerDoc)
     DISP_PROPERTY(CmyServerDoc,"Msg",m-strMsg,VT_BSTR)
 DISP_FUNCTION
(CmyServerDoc,"SetDirty",SetDirty,VT_EMPTY,VTS_14)
    //}} AFX_DISPATCH_MAP
    END--DISPATCH--MAP()
```

表2给出了各种派遣映射表

表 2

派遣映射表	描述
DECLARE_DISPATCH_MAP	声明将使用一个派遣映射表(在类声明中)
BEGIN_DISPATCH_MAP	开始派遣映射表定义
END_DISTATCH_MAP	结束派遣映射表定义
DISP_FUNCTION	定义一个OLE自动化函数
DISP_PROPERTY	定义一个OLE自动化属性
DISP_PROPERTY_EX	定义OLE自动化属性，并命名Get和Set函数
DISP_DEFVALUE	让一个已存在的属性为对象缺省值

ClassWizard用于帮助维护派遣映射表。当您增加一个新方法或属性到一个类时，ClassWizard增加相应的、具有指定参数的DISP_FUNCTION或DISP_PROPERTY

函数以及属性内部名和外部名和数据类型。同时，通过使用 Class Wizard 可以创建一个 ColeDispatchDriver 类，它是提供 OLE 自动化客户程序的主要支持。

AppWizard 简化了起始自动化服务器的步骤。若在AppWizard的OLE选项页中选择Automation,AppWizard将在 InitInstance 函数中增加注册自动化对象和作为自动化服务器运行的调用。

3.MapInfo 与 VC++ 的集成的实现

利用OLE技术在VC++6.0中调用MapInfo对象的具体实现步骤如下:

(1)在用 AppWizard 创建应用程序时，在 OLE 选项页中选择Automation。若没有选择,则首先需要加入OLE自动化的支持代码

①打开 STDAFX，h 加入

#include <afxole.h>#include <afxdisp.h>

②然后在应用程序的 InitInstance 函数中加入

if(!AfxOleInit()) {

AfxMessageBox("OLE initalization failed ");return FALSE;}

(2)创建 mapinfo 的支持类，并创建它的一个实例

①在应用程序类中加入头文件 #include"MapInfow.h"

②创建全局变量(自动化对象) DMapInfo mapinfo;

③在应用程序的 InitInstance 方法中加入

mapinfo.CreateDispatch("MapInfo.Application");

④打开 mapinfo.h 文件在底部加入

extern DMapInfo mapinfo;

#include"path-to-mapbasic-directory/mapbasic.h"

此时，已成功的利用 OLE 技术装载了 mapinfo

(3)获得 mapinfo 窗口的标识，使其成为您自己应用程序的子窗口，以便于实现两者的交互。

①在视类的头文件中加入成员变量

unsigned long m_windowid;

HWND m_windowhwnd;

②在视类的实现文件中加入头文件

#include"MapInofw.h"

③在视类的构造函数中初始化变量

m_windowid=0;

m_windowhwnd=0;

④在视类的 OnCreate 函数中，装入一幅地图，并使地图窗口变成应用程序的子窗口，获得其窗口标识及句柄。

SetWindowLong(m_hWnd,GWL_STYLE,

GetWindowLong(m_hWnd,GWL_STYLE)

|WS_CLIPCHILDREN);

char str [256];

mapinfo.Do("Open Table/"States/"Interactive");

sprintf(str,"Set Next Document Parent %lu Style 1 Map From States",(long)(UINT)m_hWnd);

mapinfo.Do(str);

m_windowid=atol(mapinfo.Eval("WindowID(0)"));

sprintf(str,"WindowInfo(0,%u)",WIN_INFO_WND);

m_windowhwnd=(HWND)atol(mapinfo.Eval(str));

⑤在视类的 OnDestroy 方法中加入代码，在应用程序终止时终止 mapinfo。

if(m_windowhwnd){

::DestroyWindow(m_windowhwnd);

m_windowhwnd=NULL;

m_windowid=OL;

}

(4)实现 mapinfo 的基本功能

①加入 mapinfo 菜单命令(以加入图层控制为例)

在应用程序主菜单中添加新项 layer contral 在主框架类中为其添加一个方法，在具体实现代码中加入

mapinfo.RunMenuCommand(M_MAP_LAYER_CONTROL)

其中M_MAP_LAYER_CONTROL宏的具体值可在mapbasic.h 中查到。

②加入mapinfo 工具条，与加入菜单命令类似，在处理函数中加入

mapinfo.RunMenuCommand(menuid)

menuid 的值根据需要在 mapbasic.h 中查询。

(5)添加回调功能(增加WindowContentsChanged方法)，当 mapinfo 窗口发生改变时，能够告知 VC++ 应用程序，相应的改变应用程序中的地图子窗口。

①在具有 OLE 自动化功能的文档类构造函数中加入

mapinfo.SetCallback(this->GetIDispatch(FALSE));

②在 Class Wizard 中选择 OLE Automation 标签，选择文档类及“Add Method”。

③在方法名中添加"WindowContentsChanged”，返回值类型为“SCODE”，参数表为“long |WindowID”。选择“确定”，并关闭对话框。则在文档类中添加一个新的方法 WindowContentsChanged。

④在WindowContentsChanged方法中填写代码，实现客户程序当MapInfo窗口发生变化时所做的处理。

三、结束语

MapInfo还提供其他的手段来实现地图窗口与其他应用程序的集成，如动态数据交换(DDE)等。但是，使用OLE自动化技术，无论是从安全性，还是从速度上看，其性能都优于DDE。同时，MapInfo还可以通过OLE属性来告知运行错误，但DDE却不具备这一功能，因而采用OLE自动化技术来实现MapInfo与VC++的集成，是一个较优的选择。■

参考文献

1 *MapInfo Professional User's Guide,MapInfo Corportion,Troy,New York*

2 *MapInfo Professional Reference,MapInfo Corportion,Troy,New York*

3 *开发Windows 95/NT4应用程序，Peter Norton Rob McGregor著，孙凤英 魏军 徐京 等译，清华大学出版社*