The Technique of GIS Desktop Secondary Development

Mingkun Xu[1, a *], Shiyun Zhou[1, b] and Cheng Shuo[1, c]

[1]Network Technology Institute

Beijing University of Posts and Telecommunications, 100876,China

[a]henry7120@hotmail.com, [b]165605847@qq.com, [c]xcsbruce@hotmail.com

Keywords: GIS, Secondary Development, Desktop, COM, Plugin, Component.

Abstract. GIS is the abbreviation of Geographic Information System. GIS desktop is the GIS software platform running on the local computer. It is the core part of GIS product series. Secondary development of general desktop platform is the main approach to develop user specific GIS desktop. The secondary development of GIS desktop platform includes three kinds of technique: the specializing of desktop startup process, the specializing of desktop appearance and plugin developing. The three techniques are all based on the infrastructure of software component. The secondary development is implemented through editing corresponding XML configuration file and writing C# program of plugin with Microsoft Visual Studio tool. This paper gives the approaches and steps to design and develop user specific desktop efficiently on the basis of readymade general GIS desktop platform.

Introduction

Introduction to GIS. GIS is a kind of information processing system on the basis of computer software and hardware, which collects, manages, presents and analyzes spatial data, and aids planning and decision making. It is widely used in the field related to spatial information such as resource management, city planning, environment management, etc. [1].

Notice: Following items have almost the same meaning in the context of this paper: desktop, desktop platform, desktop software.

Introduction to Component. Software components are some executable binary code. They own open standard interfaces for communication with outside. Components and interfaces have the property of versatility and compatibility, which let them be independent from specific computer language, and can be applied in a variety of integrated development environment (IDE), such as Visual Studio, Delphi IDE [2].

Introduction to Component-Oriented GIS. Component-Oriented GIS is a software which utilizes Object-Oriented and Component-Oriented technique [3].

General GIS products series include three categories, the first category is desktop software running on local machine; the second one is GIS server, the third one is Mobile App. Plugin is dynamically linked library (dll) which is loaded and embedded into the desktop. The SuperMap Desktop startup picture is shown in Fig.1, the bottom left corner of which depicts a plugin named SuperMap.desktop.NetWorkAnalyst.dll is being loaded in. Currently all these three categories software are based on components. Main GIS products, such as ones published by ESRI and SuperMap Corporation, all adopt components infrastructure [4, 5].

GIS server, desktop and plugins embedded in the desktop, are constituted by components, and are components themselves.

Three Approaches of GIS Desktop Secondary Development

GIS desktop is the core part of GIS products of three categories above, and secondary development on general desktop is the preferential method to develop high quality user specific desktop [6], so this paper focuses on the secondary development technique of desktop.

The secondary development of GIS desktop platform consists of three kinds of technique: the specializing of desktop startup, the specializing of desktop appearance and plugin developing.

Specializing of Desktop Startup

Specializing of desktop startup is to start a pilot program first, which can realize some functionality, such as displaying copyright and welcome page. Then the pilot program calls desktop software [6].

Main development steps below takes SuperMap Desktop product for example.

Establishing Development Environment. First establish a new Widows Form project in Visual Studio2012, then add references to two components: SuperMap.Desktop.Core and SuperMap.Data[6].

Output directory is set to the \bin directory where Desktop.exe locates.

Main Code as Following [6].

```
static void Main()
  {        //new SuperMap desktop component
     SuperMap.Desktop.Application.ActiveApplication = new SuperMap.Desktop.Application ();
       MessageBox.Show ("Start Demo ");    // display pilot program tip
     SuperMap.Desktop.Application.ActiveApplication.Initialize ();
      SuperMap.Desktop.Application.ActiveApplication.Run ();     //run the Desktop.exe
                .....
       SuperMap.Desktop.Application.ActiveApplication.Exit ();   //exit
  }
```

Then compile and execute the C# program above. It popups “Start Demo" dialog box first, then starts Desktop.exe software.

Specializing Desktop Appearance [6]

There are two approaches of specializing the Desktop appearance.

WYSIWYG in Visual Design Environment. WYSIWYG means What You See Is What You Get. Press the [Maintenance] button in the Fig.2, to enter the design environment. Here square brackets delegate button. Then press [specialize appearance] -> [environment design] -> [work environment design].

All the menu items and buttons can be modified in WYSIWYG mode. After modifying the appearance, press [OK] button to save the change.

Modify Global Configuration File. Configuration\SuperMap.Desktop.Startup.xml is the global configuration file of the desktop, which manages desktop appearance and behavior.

```
<!-if display start picture-->
  <splash enabled="false">
   <script enabled="false" codeType="CtrlAction" assemblyName="" onAction=""
          xmlns=""><![CDATA[]]></script>
     <!-if tip when workspace close-->
    <worksapce closenotify="false" visoncheck="true"> </worksapce>.
```

Plugin Development and Deployment [6]

Main method of desktop specializing is accomplished by adding user specific plugins into desktop. So the development and deployment of plugin are the crucial technique for desktop secondary development.

Main Code of the Plugin. There are three buttons at the upper left corner in Fig.2: [FullScreen], [2D], [3D].Their functionalities are to display full screen, to display 2D map, to display 3D scene respectively.

Here is the main responding code of the three buttons:

```
public override void Run()
{   (Application.ActiveApplication.MainForm as FormBase).FullScreen = true; //display full screen
    DataViewController.OpenMap (); //open 2D map
    DataViewController.OpenScene (); //open 3D scene;  }
```

Compiling it can generate the dynamic linked file DPlugin.dll, copy this dll file to the directory \Plugins\DPlugin. Each plugin owns a corresponding directory there.

Pay attention to the buttons at the upper left corner in Fig.2, where the buttons are specialized for the plugin, rather than the standard SuperMap Desktop appearance.

Plugin Configuration File. Each plugin is managed through its configuration file. Each plugin has a corresponding configuration file, the content of which includes configuration message of the plugin and its appearance element. The configuration file is written in XML format, and has the extension name *.config. The concrete plugin here involved is called DPlugin.

Following configuration file DPlugin.config is used to manage the appearance and responding function of DPlugin:

```
<tabs>
  <tab index="0" id="PMSBrowse" label="Map" formClass="" visible="true">
   <group index="0" id="RibbonGroup" label="Map Operation" >
   <button index="0" label="Select" visible="true" checkState="false"
      assemblyName="Plugins/DPlugin/DPlugin.dll" onAction="DPlugin.ActionSelect" />
   <button index="3" label="FullScreen" visible="true"
      checkState="false"assemblyName="Plugins/DPlugin/DPlugin.dll"
      onAction="DPlugin.ActionFullScreen" />
    </group>
    <group index="1" id="PipeLineBrowse" label="2D/3D Change" >
    <button index="0" label="2D" visible="true" checkState="false"
      assemblyName="Plugins/DPlugin/DPlugin.dll" onAction="DPlugin.ActionOpenMap" />
     <button index="1" label="3D" visible="true" checkState="false"
      assemblyName="Plugins/DPlugin/DPlugin.dll" onAction="DPlugin.ActionOpenScene" />
    </group>
   </tab>
</tabs>
```

This file should be copied to the directory \WorkEnvironment\Default.

Conclusions

If the functionality of readymade general desktop platform meets user' requirement mostly, only part of appearance and functionality needs to be modified, then the secondary development to the desktop can be a reasonable choice.

Secondary development described in this paper is efficient for following several reasons: The configuration files for secondary development are direct and clear. Appearance elements are WYSIWYG. And the most important is, to develop a complete desktop framework from the ground is not necessary, for plugins are designed to meet user's rather limited requirement.

Though this paper takes SuperMap Desktop as example, the secondary development of ArcGIS Desktop is almost the same

Acknowledgment

This paper is supported by the National Nature Science Fund, No.61477258. And thanks to ESRI and SuperMap Corporation for their software and technical document.

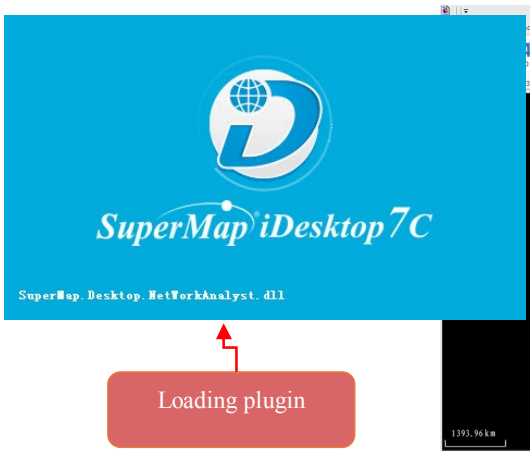

Fig.1. Plugin is loaded by the Desktop

Fig.2. DPlugin running screenshot

References

[1] Zaijun Xue, Juanjuan Ma, ArcGIS Geographic Information System Complete, QinHua University Press, 2013, pp.2-3.

[2] Mingkun Xu, COM Server Component thread Model Options, Computer Engineering and Design, 25(12), 2004, pp.2338-2340.

[3] Baidu Corporation, COM Component, http://baike.baidu.com/link?URL=twAahsTslQsdBYZVXFOYSfGZ_pc3bq4RlcnBxV9RKxpncIFXo5Jv_3fG7IHKKAF4ncjLX5JuWjf2gsFxEH0IN_, 2015

[4] Honggang Qiu, Qinglin Zhang, ArcGIS Engine Geographic Information System from Beginner to Expert, Posts Telecom Press, 2015, pp.2-5.

[5] SuperMap Corporation, SuperMap GIS 2D and 3D Union Development Practice, QingHua University Press,2013,pp.12-13.

[6] SuperMap Corporation, SuperMap Desktop .NET Plugin Development, QingHua University Press, 2012, pp.1-3, pp.157-181.