Digital Mapping Techniques '12

Association of American State Geologists

United States Geological Survey

The following was presented at DMT'12
(May 20-23, 2012).

The contents are provisional and will be
superseded by a paper in the
DMT'12 Proceedings.

See also earlier Proceedings (1997-2011)
http://ngmdb.usgs.gov/info/dmt/

Migrating Abandoned Underground Mine Applications to ArcGIS Add-Ins

Robert H. Hanover
robert.hanover@dnr.state.oh.us

James McDonald
jim.mcdonald@dnr.state.oh.us

Ohio Department of Natural Resources, Division of Geological Survey
2045 Morse Rd., Bldg. C-1, Columbus, OH 43229-6693

INTRODUCTION

The Ohio Department of Natural Resources, Division of Geological Survey (Ohio Geological Survey) has made extensive use of ArcGIS customizations for various mapping projects. Custom applications have been developed for the distribution of geologic maps and for the editing of the bedrock geology GIS dataset (McDonald and others, 2003; McDonald, 2006), creation of 3-D surficial-geology maps (McDonald and others, 2008), project management applications for Ohio's Lake Erie coastal erosion program (McDonald 2009; McDonald and others, 2010), mapping of abandoned underground mines (Hoeffler and Gordon, 2009), and the analysis of mine subsidence claims (McDonald, 2012). All of these applications have greatly increased productivity of the mapping projects, during the same time that resources have been either flat or declining.

All of the preceding applications have been written either in Microsoft's Visual Basic for Applications (VBA) or in Microsoft's Visual Basic 6.0 and using ArcObjects for ArcGIS. The VBA applications using ArcObjects allow the creation of customized code that resides within the ArcMap or ArcCatalog environment. Applications developed in Visual Basic 6.0 using ArcObjects create either a compiled file (.dll) or a stand along executable file (.exe). These applications were first created starting with ArcGIS 8.2. The applications have been successfully migrated from ArcGIS 8.3 to ArcGIS 9.0. Within the past few years, Microsoft has dropped support for VBA and has introduced VB.NET as an applications development environment to replace VB 6.0. With the introduction of ArcGIS 10.0, the support for VBA applications has been dropped and the support for VB 6.0 applications will be dropped in the next version. To replace VBA customizations and VB 6.0 applications, ESRI has introduced two new applications development environments. The first is ArcPy, a Python-based scripting language that allows access to a subset of ArcObjects. The second applications development environment is the new ArcGIS Add-In technology (Wunderlich, 2012). The Add-In technology is designed to replace the need to develop applications in VB 6.0 or create dlls within the .NET technologies. Since all of the existing DGs applications have user interfaces and new access to the full range of ArcObjects, all of the existing Ohio Geological Survey applications are being converted to ArcGIS Add-Ins.

The Ohio Geological Survey uses two custom ArcMap applications critical to mapping and reporting of abandoned underground mines (AUMs). One is a toolbar that facilitates georeferencing of paper mine maps that only have one or two points of reference (Hoeffler and Gordon, 2009). The other is a toolbar that streamlines the creation of digital geologic maps and additional documents for describing the locations of abandoned underground mines that have collapsed, resulting in subsidence. Maps and documents may then be utilized by geologists for writing claim reports requested by property owners whose property has been damaged by subsidence. These reports are then evaluated by engineering firms or insurance companies for property remediation and insurance claims.

CONVERTING EXISTING APPLICATIONS TO ADD-INS

Beginning with ArcGIS 10, Add-Ins provide simplified development and deployment of custom functionality to ArcGIS Desktop applications. Add-Ins do not use Component Object Model (COM) categories and do not require registration. Instead, Add-Ins are automatically discovered and incorporated into their target application at startup. Here is an example of a class module with COM and the Add-In version without COM.

With COM:

```
Imports System.Runtime.InteropServices
Imports System.Drawing
Imports ESRI.ArcGIS.ADF.BaseClasses
Imports ESRI.ArcGIS.ADF.CATIDs
Imports ESRI.ArcGIS.Framework
Imports ESRI.ArcGIS.esriSystem
Imports ESRI.ArcGIS.ArcMapUI

<ComClass(frmScalingCommand.ClassId, frmScalingCommand.InterfaceId, frmScalingCommand.EventsId), _
ProgId("ODNR_GeoreferenceDotNET.frmScalingCommand")> _
Public NotInheritable Class frmScalingCommand
    Inherits BaseCommand

#Region "COM GUIDs"
    ' These  GUIDs provide the COM identity for this class
    ' and its COM interfaces. If you change them, existing
    ' clients will no longer be able to access the class.
    Public Const ClassId As String = "832E9F36-84AB-4649-9E6A-0C4C8746DAA7"
    Public Const InterfaceId As String = "AC0A5719-8B6B-4B50-8A72-49736A1FAD30"
    Public Const EventsId As String = "189DEFB3-D6E1-4CC6-9BAA-1F33CF027FDA"
#End Region

#Region "COM Registration Function(s)"
    <ComRegisterFunction(), ComVisibleAttribute(False)> _
    Public Shared Sub RegisterFunction(ByVal registerType As Type)
        ' Required for ArcGIS Component Category Registrar support
        ArcGISCategoryRegistration(registerType)

        'Add any COM registration code after the ArcGISCategoryRegistration() call

    End Sub

    <ComUnregisterFunction(), ComVisibleAttribute(False)> _
    Public Shared Sub UnregisterFunction(ByVal registerType As Type)
        ' Required for ArcGIS Component Category Registrar support
        ArcGISCategoryUnregistration(registerType)

        'Add any COM unregistration code after the ArcGISCategoryUnregistration() call

    End Sub

#Region "ArcGIS Component Category Registrar generated code"
    Private Shared Sub ArcGISCategoryRegistration(ByVal registerType As Type)
        Dim regKey As String = String.Format("HKEY_CLASSES_ROOT\CLSID\{{0}}", registerType.GUID)
        MxCommands.Register(regKey)
        SxCommands.Register(regKey)
        GMxCommands.Register(regKey)
    End Sub
    Private Shared Sub ArcGISCategoryUnregistration(ByVal registerType As Type)
        Dim regKey As String = String.Format("HKEY_CLASSES_ROOT\CLSID\{{0}}", registerType.GUID)
        MxCommands.Unregister(regKey)
        SxCommands.Unregister(regKey)
        GMxCommands.Unregister(regKey)
    End Sub

#End Region
#End Region

    Public Shared m_application As IApplication
    Private m_dockableWindow As IDockableWindow
    Private m_pMxDoc As ESRI.ArcGIS.ArcMapUI.IMxDocument
```

With COM: (continued)

```
Public Shared m_mxApplication As ESRI.ArcGIS.ArcMapUI.IMxApplication

Private Const DockableWindowGuid As String = "{9a26bf09-6875-4857-82a3-2fb3f25e5d37}"

' A creatable COM class must have a Public Sub New()
' with no parameters, otherwise, the class will not be
' registered in the COM registry and cannot be created
' via CreateObject.
Public Sub New()
    MyBase.New()

    MyBase.m_category = ".NET Samples"
    MyBase.m_caption = "Scaling form"
    MyBase.m_message = "Scaling form"
    MyBase.m_toolTip = "Open Scaling form"
    MyBase.m_name = "Scalingform"

    Try
        Dim bitmapResourceName As String = Me.GetType().Name + ".bmp"
        MyBase.m_bitmap = New Bitmap(Me.GetType(), bitmapResourceName)
    Catch ex As Exception
        System.Diagnostics.Trace.WriteLine(ex.Message, "Invalid Bitmap")
    End Try
End Sub

'<summary>
''' Occurs when this command is created
''' </summary>
''' <param name="hook">Instance of the application</param>
Public Overrides Sub OnCreate(ByVal hook As Object)

    If Not (hook Is Nothing) Then
        If TypeOf (hook) Is IMxApplication Then
            m_mxApplication = CType(hook, IMxApplication)
        End If
    End If

    m_application = CType(m_mxApplication, ESRI.ArcGIS.Framework.IApplication)

End Sub

'<summary>
''' Toggle visibility of dockable window and show the visible state by its checked property
''' </summary>
Public Overrides Sub OnClick()

    Dim frmScaling2 As New frmScaling
    frmScaling2.Show()

End Sub
Public Overrides ReadOnly Property Enabled As Boolean
    Get
        Dim SelectImage2 As New SelectImage
        Dim TextBoxValue2 As String = SelectImage.TextBoxValue

        If TextBoxValue2 = "Add/Select Image" Or TextBoxValue2 = "" Then
            MyBase.m_enabled = False
        Else
            MyBase.m_enabled = True
        End If
        Return MyBase.Enabled
    End Get
End Property
Public Overrides ReadOnly Property Checked() As Boolean
    Get
        Return (m_dockableWindow IsNot Nothing) AndAlso (m_dockableWindow.IsVisible())
    End Get
End Property

End Class
```

Without COM:

```
Imports System.Runtime.InteropServices
Imports System.Drawing
Imports ESRI.ArcGIS.ADF.BaseClasses
Imports ESRI.ArcGIS.ADF.CATIDs
Imports ESRI.ArcGIS.Framework
Imports ESRI.ArcGIS.esriSystem
Imports ESRI.ArcGIS.ArcMapUI

Public Class frmScalingCommand
    Inherits ESRI.ArcGIS.Desktop.AddIns.Button

    Public Shared m_application As IApplication
    Private m_dockableWindow As IDockableWindow
    Private m_pMxDoc As ESRI.ArcGIS.ArcMapUI.IMxDocument
    Public Shared m_mxApplication As ESRI.ArcGIS.ArcMapUI.IMxApplication

    Public Sub New()

        m_mxApplication = My.ArcMap.ThisApplication
        m_application = My.ArcMap.Application
    End Sub
    Protected Overrides Sub OnClick()
        Dim frmScaling2 As New frmScaling
        frmScaling2.Show()
    End Sub

    Protected Overrides Sub OnUpdate()
        Dim SelectImage2 As New SelectImage
        Dim TextBoxValue2 As String = SelectImage.TextBoxValue

        If TextBoxValue2 = "Add/Select Image" Or TextBoxValue2 = "" Then
            Me.Enabled = False
        Else
            Me.Enabled = True
        End If
    End Sub
End Class
```

ArcGIS Desktop applications support a fixed set of Add-In types. The following Add-In types are supported in the current release (fig. 1):

- Buttons and tools
- ComboBox controls
- Menus (including root menus and context menus)
- Multi-items
- Toolbars (including premier toolbars)
- Tool palettes
- Dockable windows
- Application extensions
- Editor extensions

During installation of the ArcObjects Software Development Kit for .NET, Add-In project templates are installed into Microsoft Visual Studio for the following ArcGIS Desktop components: ArcCatalog, ArcMap, ArcScene, and ArcGlobe.

The first step of converting code to an Add-In is to create a new project using one of the Visual Studio Add-In templates (fig. 2). An Extensible Markup Language (XML) file called Config.esriaddinx is automatically created by the Add-In Project wizard. This file declares and describes the user interface components which control the functionality of the application. Users can add controls to the project using the wizard or edit the Config.esriaddinx file manually (fig. 3). The first section of the file contains metadata for the Add-In, including its name, ID, description, image, and version. The Commands tag declares and describes the controls for the application, in this case buttons and tools. The class attribute is used to associate the active code portion with the declarative portion of the Visual Studio solution.

Control declarations are visible on the Customize dialog box. For example, an ArcMap button declaration appears as a regular command item on ArcMap's Customize dialog box. From here, the button can be dragged onto any toolbar or menu like a regular COM command (fig. 4).

CONVERTING TOOLBARS AND MENUS

Add-In toolbars and menus are purely declarative. Unlike buttons and tools, toolbars and menus do not have an active portion and require no coding. Toolbar and Menu elements are the placeholders for all toolbars and menus in an Add-In and are defined under the target application element. Each Toolbar element defines a unique toolbar. The Items subelement within the Toolbar element lists all the toolbar constituents. Menus work exactly the same. Toolbars can host Button, Tool, Tool Palette, ComboBox, and Menu items. Menus can host Buttons and other Menu items. Each item must specify a refID attribute to associate the actual control with the toolbar or menu. The Toolbar tag groups the Add-In controls onto a toolbar (figs. 5 and 6).

The AUM applications were originally written within ArcMap using VBA. Then, the applications were converted to Visual Basic.NET (VB. NET) at ArcGIS 9.x. However, conversion can be migrated from VBA directly into Add-Ins. Forms used in applications that are being migrated from VBA need to be reconstructed in Visual Studio. If code is being migrated from VB.NET, the forms can be brought into the Add-In templates by right clicking the solution name and choosing Add / Existing Item and navigating to the folder where the VB.Net forms reside (fig. 7).

In the solution properties, the Debug / Start Action for the project must be set to "Start external program:" and the path to your ArcMap.exe (which has changed at version 10) must be entered (fig. 8). ArcMap document files that were created in ArcGIS 9.x can cause problems in ArcGIS 10, such as randomly crashing ArcMap for no apparent reason. So, document files should be reconstructed from scratch in ArcGIS 10 before debugging. When debugging, an untitled ArcMap document will open. Users open the document that has been constructed for the project. Next, class modules are created for all the classes being migrated and the old code copied into them, sans the COM code. Users will need to fix any changes to assemblies at version 10. For example, ESRI.ArcGIS.ADF needs to be changed to ESRI.ArcGIS.ADF.Local.

Once a project has been successfully built, an Add-In folder containing the project Add-In file (basically a Zip file) is created. The Add-In can also be managed in the Add-In Manager (fig. 9), which is available under the Customize menu in ArcMap. The Add-In file can now be e-mailed or copied to a network drive. For end users to install the application, simply double click the file. No administrator rights or registration are needed. The application will be available the next time ArcMap is started.

SUMMARY

The Ohio Geological Survey continues to extensively use ESRI customizations in the ArcGIS environment. The applications create a tremendous amount of efficiencies for geologic mapping and analysis of geologically induced hazards. With changes in the Microsoft application development environment, there were subsequent changes in the ArcGIS environment that required migration of applications from the VBA environment to the new ArcGIS Add-In technologies. The Ohio Geological Survey has successfully migrated two applications that are of critical importance to the operation of the Division. These two applications allow the Ohio Geological Survey to continue increasing the accuracy of the mapping of abandoned underground mines and to pull into one document all the known geologic information for a mine subsidence claim.

REFERENCES

Hoeffler, Paul, and Gordon, Chris, 2009, Ohio underground mine map georeferencing project, in Digital Mapping Techniques '09, Morgantown, W.Va., May 10–13, 2009, Proceedings: Ohio Department of Natural Resources, Division of Geological Survey poster presentation, last accessed May 15, 2012, at <http://www.dnr.state.oh.us/portals/10/pdf/Posters/DMT2009_Hoeffler.pdf>.

McDonald, James, 2006, GIS data access and distribution using a GIS map application labs), in Abstracts with Programs, 2006 Annual Meeting of the Geological Society of America, October, 22–25, 2006, Philadelphia, Pa.: Geological Society of America, v. 38, no. 7, p. 163, last accessed May 15, 2012, at <http://gsa.confex.com/gsa/2006AM/finalprogram/abstract_113917.htm>.

McDonald, James, 2009, Project-management GIS applications for coastal-erosion mapping in Ohio, in Digital Mapping Techniques '08—Workshop Proceedings, Moscow, Idaho, May 18–21, 2008: U.S. Geological Survey Open-File Report 2009-1298, p. 177–184, last accessed May 15, 2012, at <http://pubs.usgs.gov/of/2009/1298/pdf/usgs_of2009-1298_mcdonald.pdf>.

McDonald, James, 2012, Evaluating mine subsidence using a GIS software application, in Digital Mapping Techniques '10—Workshop Proceedings, Sacramento, Calif., May 16–18, 2010: U.S. Geological Survey Open-File Report, 25 p., last accessed May 15, 2012, at <http://ngmdb.usgs.gov/Info/dmt/docs/DMT10_Draft_McDonald.pdf>.

McDonald, James, Harbula, Paul, Mackey, S.D., 2010, New GIS tools for mapping Ohio's Lake Erie coastal erosion areas, in Digital Mapping Techniques '09—Workshop Proceedings, Morgantown, W.Va., May 10–13, 2009: U.S. Geological Survey Open-File Report 2010-1335, p. 195–205, last accessed May 15, 2012, at <http://pubs.usgs.gov/of/2010/1335/pdf/usgs_of2010-1335_McDonald.pdf>.

McDonald, James, Pavey, R.R., Vorreis, E.R., and Wells, J.G., 2008, GIS tools for 3D surficial mapping in Ohio, in Digital Mapping Techniques '07—Workshop Proceedings, Columbia, S.C., May 20–23, 2007: U.S. Geological Survey Open-File Report 2008-1385, p. 109–121, last accessed May 15, 2012, at <http://pubs.usgs.gov/of/2008/1385/pdf/mcdonald.pdf>.

McDonald, James, Swinford, E.M., Wickstrom, L.H., Slucher, E.R., Powers, D.M., and Berg, T.M., 2003, Bedrock geology and bedrock topography GIS of Ohio—New data and applications for public access, in Digital Mapping Techniques '03—Workshop Proceedings, Millersville, Pa., June 1–4, 2003: U.S. Geological Survey Open-File Report 03-471, p. 3–16, last accessed May 15, 2012, at <http://pubs.usgs.gov/of/2003/of03-471/pdf/mcdonald.pdf>.

Wunderlich, A.L., 2012, Automation in ArcGIS 10—Understanding changes in methods of customization and options for migration of legacy code, in Digital Mapping Techniques '10—Workshop Proceedings, Sacramento, Calif., May 16–18, 2010: U.S. Geological Survey Open-File Report, 14 p., last accessed May 15, 2012, at <http://ngmdb.usgs.gov/Info/dmt/docs/DMT10_Draft_Wunderlich.pdf>.
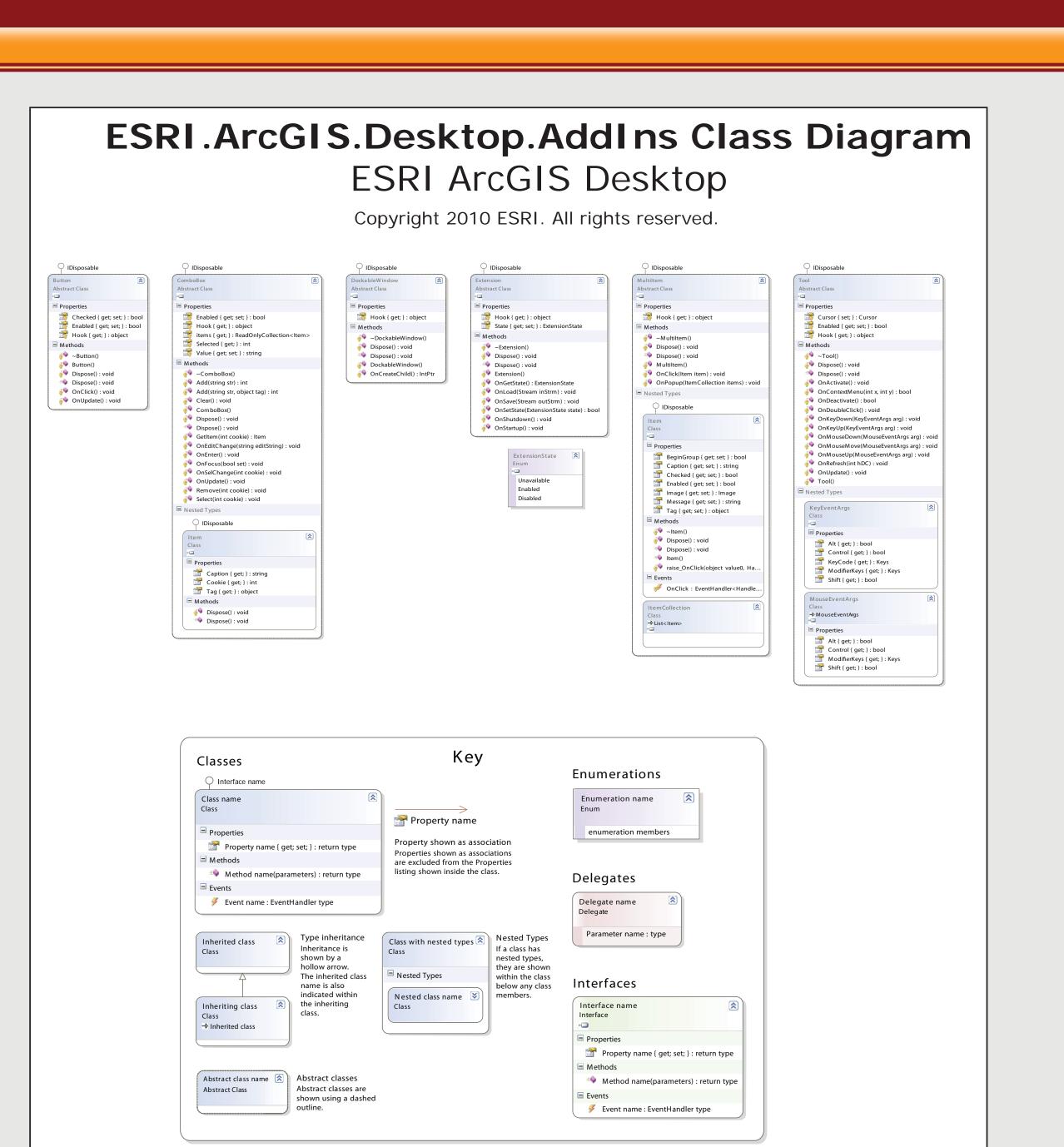
FIGURE 1.—Example class diagrams for the common ERSI Add-In classes.

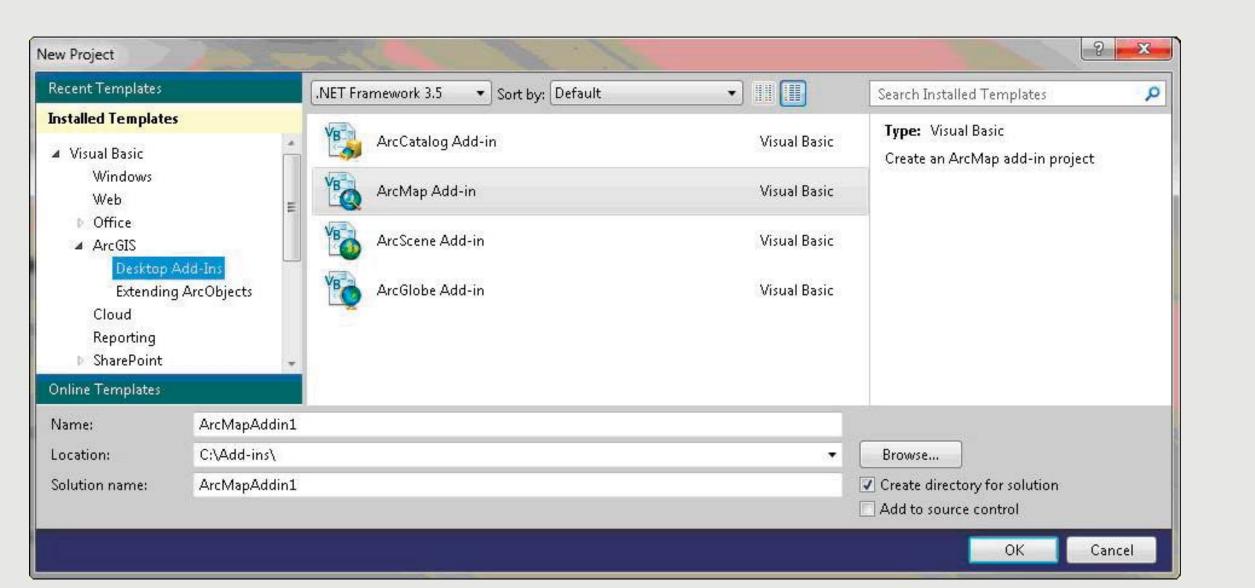FIGURE 2.—The New Project dialog showing the ArcMap Add-In templates.

FIGURE 3.—An XML configuration file, Config.esriaddinx, in Microsoft Visual Studio.

FIGURE 4.—The ArcMap Customize dialog showing Add-In commands.

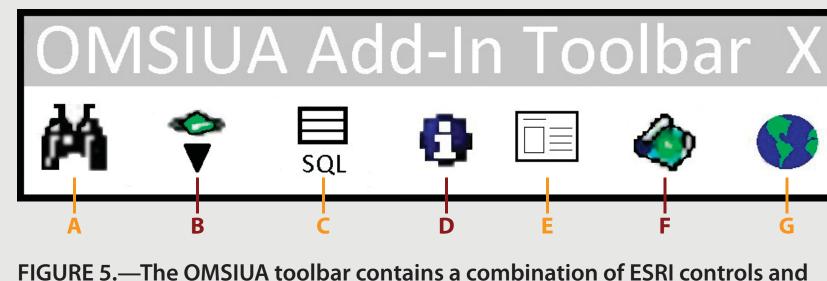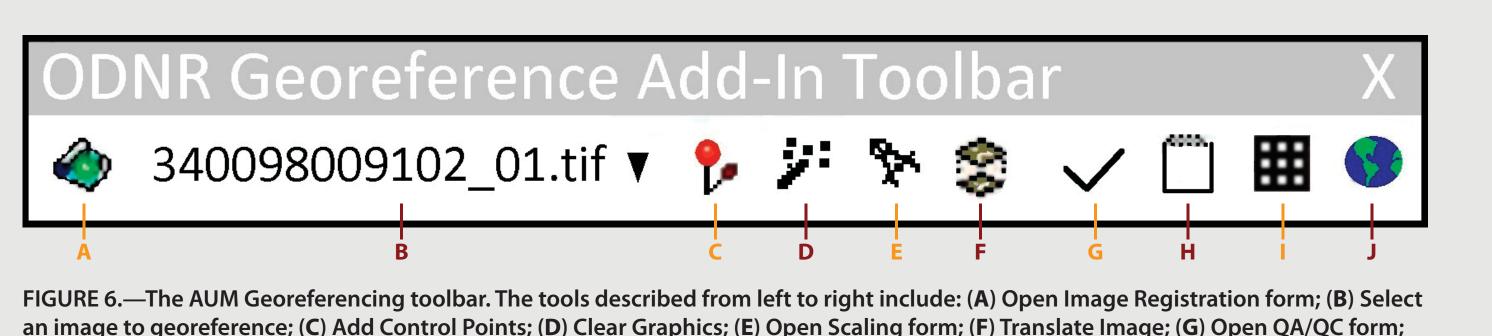FIGURE 5.—The OMSIUA toolbar contains a combination of ESRI controls and custom controls. The tools described from left to right include: (A) the native ERSI Find tool; (B) Select AUM Location tool, which loads all known geologic maps and records into the ArcMap; (C) a tool to open the Underground Mine Information database in MS-Access; (D) the Underground Mine Information form, which allows the selected mine attribute information to be loaded into a form for display; (E) the native ESRI Hyperlink tool; (F) the Export PDF tool; and (G) a tool to reset and zoom to the entire state. See McDonald (2012) for further explanation of the tools.

FIGURE 6.—The AUM Georeferencing toolbar. The tools described from left to right include: (A) Open Image Registration form; (B) Select an image to georeference; (C) Add Control Points; (D) Clear Graphics; (E) Open Scaling form; (F) Open QA/QC form; (H) Open Add Comment form; (I) Quarter-Quad Tool; and (J) ESRI Georeference Toolbar. See Hoeffler and Gordon (2009) for further explanation of the tools.

FIGURE 7.—Adding an existing item to Visual Studio.

FIGURE 8.—Setting the debug start action in project properties.
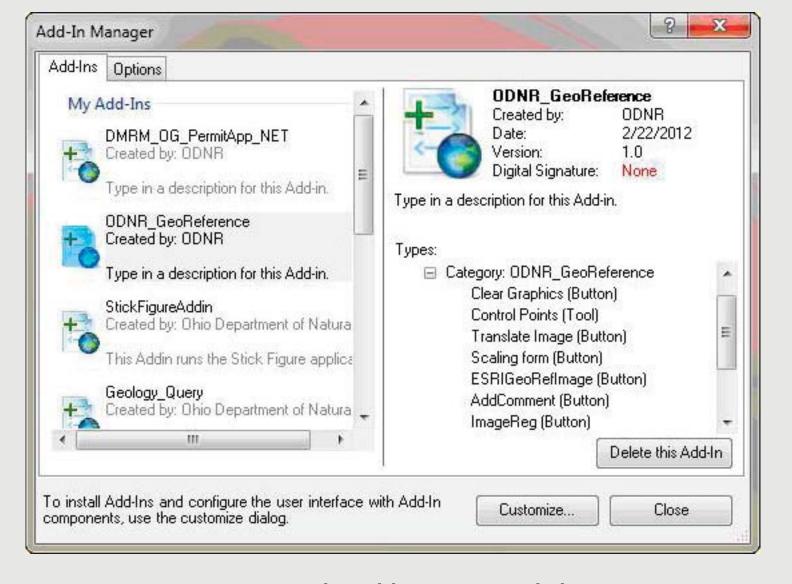
FIGURE 9.—The Add-In Manager dialog.