Moving Desktop Applications to ArcGIS Server

Kelly Hutchins
Jian Huang
Schedule

• 75 minute session
  – 60 – 65 minute lecture
  – 10 – 15 minutes Q & A following the lecture

• Cell phones and pagers

• Please complete the session survey – we take your feedback very seriously!
Introductions

• Who are we?

• Who are you?
  – Development experience with ArcObjects
  – Development experience with .NET and ASP.NET
  – Basic understanding of ArcGIS Server
Questions

• How many:
  – Want to migrate Desktop (VBA, VB, VC++, .NET) applications?
  – Want to migrate Engine (VB, VC++, .NET) applications?
  – Already have ArcGIS Server development experience?
Topics

• ArcGIS Desktop customizations

• How to migrate customizations to Server

• Best practices for migrating
Types of ArcObjects applications and extensions

• **ArcGIS Desktop – ArcObjects Extensions**
  – ArcMap, ArcCatalog, ArcScene and ArcGlobe
  – Commands, tools, toolbars, windows, extensions
  – Mostly COM implementations (VB 6, VC++, .NET)

• **ArcGIS Engine – ArcObjects Applications**
  – Custom standalone forms-based and utility applications
  – Commands, tools, extensions (VB 6, VC++, Java, .NET)
  – COM or pure .NET implementations

• **ArcObjects Components**
  – Utility components and DLLs
Migrating desktop applications: The challenge

1. ArcGIS Server is a multi-tiered, multi-API system
2. ArcObjects code runs remotely on the GIS Server
3. User interface is a web browser
Migration options

1. Migrate directly to ASP.NET
2. Build Utility Object or Server Object Extension
3. Geoprocessing
4. Use the ArcGIS Server Web ADF
   • Common data-source api
   • Data-source specific API
5. JavaScript API
Which ArcObjects components can be used?

- Most of the Engine libraries
- UI libraries not available
- Capabilities
  - Display
  - Symbolization
  - Analysis
  - Query
  - Data access
  - Editing
  - Output

Accessing ArcObjects remotely

1. Web app communicates through ArcObjects proxies
2. Accesses a Server Object through the SOM
3. Works with ArcObjects in a Server Context (ArcSOC.exe)
Migrating ArcObjects code to ASP.NET

- Move code from .NET windows application to ASP.NET
- Create ArcObjects in a server context

```vba
Dim pPoint as IPoint = New Point

Dim pPoint as IPoint = serverContext.CreateObject("esriGeometry.Point")
```

- Use Library Locator to determine CLSID
Migrating ArcObjects code directly to ASP.NET

1. Build an ASP.NET web application

2. Copy/paste your ArcObjects code

3. Replace “New” with “CreateObject”

4. Manage objects in the server context
Migrating ArcObjects code to a Utility Object

• Build a custom COM component to reduce the number of fine-grained ArcObjects calls

• Moves code to the GIS Server (SOC)

• When to use
  – Need to access a large ArcObjects code base
  – Share code with other applications
  – Optimize performance
Creating a COM utility object

• General steps
  1. Create a new project (VB6, .NET, VC++)
  2. Reference the ESRI COM libraries or .NET assemblies
  3. Define a COM class
  4. Define a public interface or public members
  5. Migrate existing ArcObjects code
  6. Use CreateObject to create and access objects

• Required to register COM component on all SOC machines
Migrating to a Server Object Extension

- Extends the capabilities of the Server Object
- Initialized once during server object startup
- Configurable in ArcCatalog through custom property page.
COM utility object versus server object extension

• COM utility objects
  – Initialized when required by the GIS server
  • May happen several times
  – Not registered with a specific server object instance
  • Created “ad-hoc” using the server context

• Server object extensions
  – Initialized once during server object startup
  – Can benefit from caching logic
  – Registered with specific server objects (Map Services)
  – Configurable in ArcCatalog through custom property page
Implementing a Server Object Extension (SOE)

1. Build a utility object
2. Build a SOE object (IServerObjectExtension)
3. (Optional) Build a custom property page for ArcCatalog
4. Access SOE from web application
Migrating – Geoprocessing Service

- Replace ArcObjects code with a Geoprocessing Service
- Build a model and return results to a web application

- When to use
  - Large processing tasks that can be modeled
  - Optimize ArcObjects tasks
Migrating – Geoprocessing Service

1. Build a Geoprocessing model

2. Publish the model through a Map and Geoprocessing service

3. Consume as a task in the Web Server application
Introducing the Web ADF

Replace ArcObjects functionality with the ADF

1. Web controls API
2. Common Datasource API
3. Datasource Specific API

Web ADF – Development Paths

Developer Paths

Web Controls

Generic

Common Data Source API

Implementations

Data Source Specific APIs
Migrating the User Interface – Web Controls

- Visual Studio Map template

- WebControls
  - Pure ASP.NET controls
  - AJAX enabled

- Functionality
  - Visualization
  - User interaction

Migrating the UI - Commands and Tools

1. Start with the Map template
2. Add new item to Toolbar
3. Specify client-side action
4. Create a new class and implement interface

<table>
<thead>
<tr>
<th>Tool</th>
<th>IIMapServerToolAction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>IMapServerCommandAction</td>
</tr>
<tr>
<td>DropDownBox</td>
<td>IMapServerDropDownBoxAction</td>
</tr>
</tbody>
</table>

5. Define server-side action
Migrating the UI – Task Framework

• Tasks: Objects that encapsulate business logic

• Out-of-the-box tasks
  – Search Attributes
  – Find Address
  – Editor Task
  – Print Task

• Container for displaying task results
ArcGIS Server Web ADF – Development Paths

Developer Paths

I

II

III

IV

Web Controls

Generic

Common Data Source API

Implementations

Data Source Specific APIs
The Common Datasource API

- ArcGIS Server supports multiple data sources
  - ArcGIS Server, ArcIMS, ArcWeb services, OGC, Graphics

- Common Datasource API
  - .NET classes for the Web ADF
  - Access and interact with all data sources the same way!

- Provides different functionalities - query, find, identity…
The Common Datasource API

Resource managers

IGISDataSource

IGISResource

IMapResource

IGeocodeResource

IGeoprocessingResource

IGISFunctionality

IGeocodeFunctionality

IGeoprocessingFunctionality

Web ADF controls

IMapFunctionality

IMapTocFunctionality

IScalebarFunctionality

IQueryFunctionality

ITileFunctionality

Common data source API

Data source implementations

ArcGIS Server

ArcIMS

Web ADF controls

ArcGIS Server

ArcIMS
Using the Common Datasource API

• Where possible, replace ArcObjects code

• Examples – Query, Identify, Find

• Steps
  1. Identify the ArcObjects code to be replaced
  2. Implement the Common API
  3. Reuse code to access other data sources

– Review SDK, code samples and OMDs…
ArcGIS Server Web ADF – Development Paths

Developer Paths

I

II

III

IV

Web Controls

Generic

Common Data Source API

Implementations

Data Source Specific APIs
Data source-specific APIs

• Composed of data sources that have functionality beyond Common data source API
  – ArcGIS Server
  – ArcIMS
  – ArcWeb Services
Programming with data source-specific APIs

• Each data source exposes a different set of functionality
  – ArcGIS Server
    • SOAP, ArcObjects
      – ArcIMS – AXL
      – ArcWeb Services – SOAP API
  • What does this mean?
    – Many other data source-specific classes available
    – More business/GIS logic
    – Different APIs use different communication protocols
    – Requires different programming patterns for each data source
The ArcGIS Server SOAP API

- Exposes a sub-set of ArObjects functionality
- Composed of a number of value and proxy objects
- Objects work with both Internet and local resources
- Value objects
  - Geometry, symbology, query filters, spatial reference…
- Proxy objects
  - Emulate functionality provided by coarse-grained server objects (MapServer, GeocodeServer)
- See ESRI.ArcGIS.ADF.ArcGISServer
What are Value and Proxy objects?

- **Value objects**: native .NET classes used to program against ArcGIS Server data sources
- **Proxy objects**: use proxies to communicate with server end points
- **Proxy objects perform two main tasks:**
  - Serializing Value objects to SOAP that is sent to resource
  - Deserializing SOAP responses to Value objects for client
Publish a resource as a service

<table>
<thead>
<tr>
<th>Service</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geocoding</td>
<td>Address locator</td>
</tr>
<tr>
<td>Geodata</td>
<td>Query, Extraction and Replication</td>
</tr>
<tr>
<td>Geometry</td>
<td>Perform geometric calculations such as project and densify</td>
</tr>
<tr>
<td>Geoprocessing</td>
<td>Access to geoprocessing models</td>
</tr>
<tr>
<td>Network Analysis</td>
<td>Solves transportation NA problems</td>
</tr>
<tr>
<td>Mapping</td>
<td>Map viewing and access to contents of map</td>
</tr>
</tbody>
</table>

http://localhost/ArcGIS/SDK/SOAP/
Migrating graphics - Web ADF graphics data source

• Draws on top of layers in map
• Used to perform tasks such as:
  – Highlighting features (select)
  – Labeling text
  – Displaying buffers
  – Geocoding
  – Displaying dynamic data (GPS)
• Provides alternative to drawing graphics on server
  – Server-tier graphics are part of map image
• Rendering occurs independently from map
  – Does not require map redraw
Migrating graphics

• General steps
  1. Create a new graphics layer
  2. Add to graphics dataset
  3. Create new geometry
  4. Define symbols and rendering
  5. Create graphic elements
  6. Add to graphics layer
  7. Redraw graphics resource
  8. Render on client or server
JavaScript API

- Browser based API
- Lightweight
- Embed maps and tasks into Web applications
  - Display an interactive map
  - Execute a geoprocessing model
  - Display your data on an ArcGIS Online base map
  - Search for features or attributes
  - Geocode an addresses and display the results
## ArcGIS Desktop vs ArcGIS Server Comparison

<table>
<thead>
<tr>
<th>ArcGIS Desktop/Engine</th>
<th>ArcGIS Server Web ADF</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Commands, tools and toolbar</td>
<td>• Buttons, tools, toolbar and tasks</td>
</tr>
<tr>
<td>• TOC</td>
<td>• TOC</td>
</tr>
<tr>
<td>• DockableWindow</td>
<td>• FloatingPanel, Treeviewplus</td>
</tr>
<tr>
<td>• Map and PageLayout</td>
<td>• Map and PageLayout</td>
</tr>
<tr>
<td>• Extension</td>
<td>• SOE</td>
</tr>
<tr>
<td>• ArcObjects</td>
<td>• Common API</td>
</tr>
<tr>
<td>• Graphics</td>
<td>• Datasource Specific API</td>
</tr>
<tr>
<td></td>
<td>• Graphics API</td>
</tr>
</tbody>
</table>
Best Practices

• Leverage the Web ADF API as much as possible
  – GIS services and capabilities
    • Geoprocessing, Geodata, Network Analysis
      – Task framework
    • Print, Editor, Search, Query
  • Minimize fine-grained calls to remote ArcObjects
    – COM Utility object
    – Server Object Extension (SOE)

Summary

Migrating Desktop applications to Server

• Migrating pure ArcObjects code
  • ASP.NET, DCOM and Utility objects
  • SOE and Geoprocessing

• Migrating the User Interface
  • ArcGIS Server Web controls
  • ArcGIS Server Web ADF framework (Commands and Tools)

• Replacing ArcObjects
  • Common Datasouce API (Query, Identify, Find…)
  • Datasource Specific API (ArcGIS Server SOAP API and Graphics)
Presentation materials

• PowerPoint presentation and code are posted on the conference web site

• EDN – downloads and videos
Further questions?

• TECH-TALK AREAS
  – **What:** Further opportunity to discuss questions and concerns with presenters and subject matter experts
  – **Where:**
  – **When:** during the next 30 minutes

• ESRI Showcase

• Meet the teams

• ESRI Developers Network (EDN) website
The end – Thank you!
In Conclusion...

- Other recommended sessions and meetings

- All sessions are recorded and will be available on EDN
  - Slides and code will also be available

- Please fill out session surveys!

- Still have questions?
  1. Tech talk, Demo Theatres, Meet the Team
  2. “Ask a Developer” link on web page
     - [www.esri.com/devsummit/techquestions](http://www.esri.com/devsummit/techquestions)