Working with ArcGIS Server Web Services
Using the SOAP API

Sud Menon
Julio Andrade
Rex Hansen
Introductions

• **Who are we?**
  – Sud Menon
  – Julio Andrade
  – Rex Hansen

• **Who are you?**

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

• Brief overview of ArcGIS Server Web Services

• 9.3 SOAP API Walkthrough:
  – Functional capabilities of each GIS web service
  – The SOAP API for the service
  – Sample applications that consume the service

• SOAP API : Under the Hood
  – Implementation details of interest to you
ArcGIS Server Web Services - Overview

- ArcGIS Server has a rich set of GIS Web Services

- You can work with Web Services using the following api’s / interfaces:
  - SOAP
  - REST
  - OGC
ArcGIS Server – Publishing GIS Web Services

• Author GIS resources (e.g., maps, locators) using desktop.

• Publish GIS resources to create GIS Services.

• A GIS Service may have multiple capabilities
  – Eg Mapping, Network Analysis

• Each capability is exposed to consumers as an independent GIS Web Service accessible over HTTP via SOAP or REST
ArcGIS Server : SOAP Web Service URLs

- A standard scheme for all SOAP web service URLs:
  
  - http://<hostname>/<arcgis_instance>/services/<folder>/
    <servicename>/<servicetype>

- http://gis.mybiz.com/arcgis/services/usa/mapserver

- http://gis.mybiz.com/arcgis/services/northernca/SanFranciscoLocator/geocodeserver

- To get the wsdl:
  
ArcGIS Server SOAP Web Services -
Out of the Box Clients

- ArcGIS Explorer and ArcGIS Desktop are out of the box Desktop Clients
- You can use Manager wizards to build Web Mapping Applications that consume GIS Web Services
  - no programming required
Options for Developing Custom Client Applications that use SOAP Web Services

• **Work with the Web ADF object model (for web apps)**
  – Use the ArcGIS Server Internet data source

• **Work with ArcGIS Engine (for desktop apps)**
  – Use the ArcEngine object model and controls
  – Use MapServiceLayers, WebLocators and built-in ArcObjects proxies for each web service

• **Work directly with the SOAP API**
  – *The focus of this presentation*
  – You can use pre-generated proxies that ship with the .Net and Java Web ADFs, eg ESRI.ArcGIS.ADF.ArcGISServer
  – You can generate the proxies yourself using your favourite wsdl tool
Working with a SOAP service

• **You work with Proxies and Value objects**
  – Generated from a WSDL using a SOAP toolkit

![Diagram showing the interaction between Client Application, SOAP Toolkit, End Point, and Server with arrows indicating flow of operations like Get WSDL, Generate, and service.](image-url)
SOAP Web Services - Advantages

• easily accessible from .Net, Java, C++

• well defined interface contract

• lets you take advantage of compile time type checking

• lends itself to standard OO programming

• fully integrated into IDE’s
  – Class documentation, Full intellisense
The ArcGIS Server SOAP API

- Coarse Grained
- Stateless

- Described by a *wsdl* - the calling contract for the service

- Methods take serializable Value objects (simple structures) as parameters

- Value Objects are based on existing well known ArcObjects types
  - Eg Color, Symbol, Graphic Element, Geometries, DataElements

- Simple
- Powerful
SOAP Web Services in 9.3

- **Service Catalog Service** – discover the services on this server
- **Map Service** – draw maps, query data
- **Geocoding Service** – address to location, location to address
- **Geometry Service** – projection, densification, buffering
- **Network Analysis Service** – routing and network problem solving
- **Image Service** – access to large image repositories
- **Geoprocessing Service** – synchronous and asynchronous spatial analysis
- **Geodata Service** – data discovery, query, extraction and replication
  - Mobile Service (accessed using the Mobile ADF client)
  - Globe Service (accessed using the ArcEngine client)
Discovering Services – The Service Catalog Web Service

Every ArcGIS Server has a Service Catalog Web Service:

http://<web server hostname>/<arcgis instance>/services?wsdl

- What Services do you have?

ServiceCatalog

GetServiceDescriptions

ServiceDescription

Name, Type, URL

WSDL

Method

Response

Value Object

Value Object
Map Service

• Based on a Map Document (.mxd) authored using ArcMap

• Can be used to **dynamically generate map images** in a variety of image formats

• Can be used to **overlay additional information** on top of the map
  – Graphics
  – Selection IDs and buffer distances
Map Service

• Can be used to identify features

• Can be used to query features
  – Results returned as Geodatabase record sets
  – Easily marshaled by value
  – Can be converted to ADO.NET datasets
  – Can return densified geometries (new at 9.3)
  – Can return geometries as KML (new at 9.3)
Map Service

- Maps can be pre-rendered and cached for performance resulting in a tiled map service that has a fixed set of scales

- Methods to discover if
  - A map is a tiled fixed scale map
  - If the map tiles correspond to single fused images or if there are separate tiles for each layer

- Methods to get the tiling scheme
- Methods to fetch tiles
- Tiles can also be obtained directly via HTTP Get.
Working with the Map Service

// Drawing the map - generating map images
MapImage mi = ms.ExportMapImage (mapDescription, imageDescription);

// Querying the map
Recordset rs = ms.QueryFeatureData(mapName, layerID, queryFilter);

FidSet fs = ms.QueryFeatureIDs(mapName, layerID, queryFilter);

MapServerFindResult[] results = ms.Find(mapDescription, mapImageDisplay, searchString,
    expression, searchFields, findOption, layerIDs);
Map Service Object Model - Drawing

MapServer

MapServerInfo ----> MapLayerInfo

MapDescriptor

LayerDescriptor

ImageDescriptor

ImageDisplay

ImageType
Map Service Object Model – Querying Features

MapServer

Recordset → Record

IdentifyResults → IdentifyResult

FindResults → FindResult
Map Service Object Model – Adding Custom Graphics

- MapServer
- GraphicElement
  - Geometry
  - Symbol
  - MarkerElement
  - LineElement
  - PolygonElement
  - PointN
  - LineN
  - PolygonN
  - SimpleMarkerSymbol
  - SimpleLineSymbol
  - SimpleFillSymbol
## Map Server - Key Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MapCount</td>
<td>No of maps in the service</td>
</tr>
<tr>
<td>MapName</td>
<td>Name of a map, by index</td>
</tr>
<tr>
<td>GetServerInfo</td>
<td>Retrieves Information about a specified map</td>
</tr>
<tr>
<td>ExportMapImage</td>
<td>Generates an image based on a specified map description</td>
</tr>
<tr>
<td>GetLegendInfo</td>
<td>Returns legend information for specified layers</td>
</tr>
<tr>
<td>Identify</td>
<td>Returns a collection of Identify Results</td>
</tr>
<tr>
<td>Find</td>
<td>Returns a collection of Find Results</td>
</tr>
<tr>
<td>QueryFeatureData</td>
<td>Returns a recordset of features based on a query filter</td>
</tr>
<tr>
<td>FromMapPoints</td>
<td>Converts from map to screen/image coordinates</td>
</tr>
<tr>
<td>ToMapPoints</td>
<td>Converts from screen/image to map coordinates</td>
</tr>
</tbody>
</table>
MapDescription value object:

Properties used to control the drawing of the map

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the map</td>
</tr>
<tr>
<td>LayerDescriptions</td>
<td>A collection of layer descriptions</td>
</tr>
<tr>
<td>CustomGraphics</td>
<td>CustomGraphics to be drawn on the map</td>
</tr>
<tr>
<td>SpatialReference</td>
<td>The spatial reference in which to draw the map</td>
</tr>
<tr>
<td>TransparentColor</td>
<td>The color that is transparent</td>
</tr>
</tbody>
</table>
LayerDescription value object:

Properties used to control the drawing of a layer

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LayerID</td>
<td>The ID for the layer</td>
</tr>
<tr>
<td>DefinitionExpression</td>
<td>The definition expression for the layer</td>
</tr>
<tr>
<td>Visible</td>
<td>Is the Layer Visible</td>
</tr>
<tr>
<td>ShowLabels</td>
<td>Should labels be drawn</td>
</tr>
<tr>
<td>SelectionFeatures</td>
<td>Selected feature ids</td>
</tr>
<tr>
<td>SelectionColor</td>
<td>Color to be used for selected features</td>
</tr>
</tbody>
</table>
Map Service demo

GeoDB -> MXD -> Map Svc -> GIS Server -> Web Server -> client
Cached Map Services

- Tiles pre-rendered at fixed scales
- Rapid display of static base maps
- Richer symbols and more information
Map Service Object Model – Tiled Map Services

MapServer

TileCacheInfo

LODInfo

SpatialReference
### Map Service - Methods for working with tiled map services

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsFixedScaleMap</td>
<td>Indicates that the map has a fixed set of LODs</td>
</tr>
<tr>
<td>HasSingleFusedMapCache</td>
<td>Indicates that the map has a single cache that prefuses all layers</td>
</tr>
<tr>
<td>GetTileCacheInfo</td>
<td>Gets the tiling scheme for the map</td>
</tr>
<tr>
<td>GetMapTile</td>
<td>Gets the specified tile for the map given tile level, row and column coordinates</td>
</tr>
<tr>
<td>GetVirtualCacheDirectory</td>
<td>Gets the virtual cache directory for the map, if present</td>
</tr>
</tbody>
</table>
Tile Cache Info Object:

Properties that describe a tiling scheme

<table>
<thead>
<tr>
<th>LODInfos</th>
<th>The levels of detail in the cache {id, scale, resolution}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>The origin for the tiling grid</td>
</tr>
<tr>
<td>TileCols</td>
<td>The number of columns in a tile</td>
</tr>
<tr>
<td>TileRows</td>
<td>The number of rows in a tile</td>
</tr>
<tr>
<td>DPI</td>
<td>Resolution in dots per inch</td>
</tr>
<tr>
<td>SpatialReference</td>
<td>The spatial reference of the cache</td>
</tr>
</tbody>
</table>

\[
tileWidth = \text{tileCols} \times \text{LODResolution}
\]

\[
tileColumn = \text{floor} \left( \frac{x - xorig}{tileWidth} \right)
\]

\[
tileHeight = \text{tileRows} \times \text{LODResolution}
\]

\[
tileRow = \text{floor} \left( \frac{y - yorig}{tileHeigh} \right)
\]
Geometry Service

- A computational service, not bound to any data on the server. New at 9.3

- An ArcGIS Server can host a Geometry Service
  - Number of instances (service threads) can be increased as needed

- Provides basic geometric operations for use by web service clients

- You can also support geometric clients by publishing GP tools like buffer, project and intersect
Geometry Service – Object Model

- GeometryServer
- GeometryArray
- Geometry
- SpatialReference
- Unit
- Transformation
## Geometry Service - Key Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffer</td>
<td>Buffers each geometry in an array of geometries by distances specified in an array of distances</td>
</tr>
<tr>
<td>Project</td>
<td>Projects an array of geometries from their current spatial reference to the specified spatial reference</td>
</tr>
<tr>
<td>Simplify</td>
<td>Generates topologically correct geometry</td>
</tr>
<tr>
<td>Densify</td>
<td>Converts non linear geometry to linear geometry</td>
</tr>
<tr>
<td>Relation</td>
<td>Determines the pairs of geometries in the input that participate in the specified topologic relationship</td>
</tr>
</tbody>
</table>
Geocode Service

• Integrated application using Map, Geocode and Geometry Services
Demo

- Based on a Locator stored in the file system or in a Geodatabase

- Methods to perform single address and batch geocoding

- Methods to perform reverse geocoding

- Support for different address styles
Network Analysis Service

• Based on publishing a map containing network analysis layers

• A network analysis layer represents a specific analysis (Routing, Service Area, Closest Facility) to be performed against a specific network dataset (for eg. LA Streets)

• Methods to :
  – Discover the parameters of the analysis solver
  – Solve
  – Get Results (paths, directions etc)

• Callers supply input parameters (for eg stops, barriers) using basic web service types such as PropertySets and RecordSets.
Network Analysis Service – Object Model

- SolverParams
- NAServer
- SolverResults
- RouteParams
- ClosestFacilityParams
- ServiceAreaParams
- RouteResults
- ClosestFacilityResults
- ServiceAreaResults
- NAServerLocations
- NAServerPropertySets
NetworkAnalysis Server - Key Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetNALayerNames</td>
<td>Array of Network Analysis Layers of a particular type</td>
</tr>
<tr>
<td>GetNetworkDescription</td>
<td>Returns the Network Dataset Data Element for the specified layer</td>
</tr>
<tr>
<td>GetSolverParameters</td>
<td>Default Solver Parameters for the specified Network Analysis Layer</td>
</tr>
<tr>
<td>Solve</td>
<td>Perform Network Analysis based on the specified solver parameters</td>
</tr>
</tbody>
</table>

• Finding a Route:
  – Get Route Network Analysis Layer
  – Get Default Route Solver Parameters
  – Specify Desired Route Solver Parameters
    • Barriers, Stops, FindBestSequence, UseTimeWindows, ...
  – Solve
  – Consume Results
Geocoding and Network Analysis Service demo - routing
Image Service

• **New at 9.3**

• **Allows you to serve imagery and raster data:**
  – Raster Datasets
    • eg large mosaics in SDE
  – Raster Layers
    • With rendering set up on the server
  – Image Service Definitions
    • Catalogs of file based imagery that can be mosaiced and processed on the fly
    • Uses a back end ArcGIS Image Server

• **Clients can control**
  – Returned image format, mime or url
  – band selection, compression, projection, interpolation
Image Service – Object Model

ImageServer

ImageServiceInfo

GeoImage

Description

ImageType

ImageResult
## Image Server - Key Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExportImage</td>
<td>Returns the requested image using a well known image format</td>
</tr>
<tr>
<td>GetImage</td>
<td>Returns the requested image as a byte stream</td>
</tr>
<tr>
<td>GetServiceInfo</td>
<td>Returns the properties of the image service - numBands, pixelType, statistics etc</td>
</tr>
</tbody>
</table>
Image Service demo - routing

- RDS
- ImageSvc
- GIS Server
- Web Server
- WebApp
ArcGIS Server 9.3 - Geoprocessing Service

- Allows organizations to centralize both data and processing on the server

- Allows organizations with spatial data to expose spatial analysis functionality against that data as easy to use tasks that can be invoked by less experienced users

- Allows GIS Analysts to easily author and publish geoprocessing models
ArcGIS Server 9.3 - Geoprocessing Service

• You publish either a geoprocessing Toolbox or a Map containing Tool Layers

• The service makes the tools available to the consumer

• A GP Service may be associated with a map service that can be used to render results created on the server.

• Clients can also fetch vector and raster results by value.
Geoprocessing Service – 9.3 Enhancements

**Improved Performance**
- Reduced framework overhead for fast synchronous services
- Optimized transfer of large recordsets to and from server
- Tools have improved performance against SDE
- Faster execution of Python Script Tools

**Improved Logging**

**More output formats (KML)**

**Improved Publishing**
- Better error reporting
- Better detection of models that do not confirm to server publishing rules (incompatible types)

**Improved documentation**
- Included with both Server and Desktop
Geoprocessing Service - Parameters

• Supported parameter types
  – String
  – Long
  – Double
  – Boolean
  – Date
  – Linear unit
  – Feature Set
  – Record Set
  – Raster
Geoprocessing Service Properties

• Execution Mode:

  – Synchronous
    • Results calculated and returned immediately by value
    • Appropriate for fast tools with small results

  – Asynchronous
    • Allows client to submit a job and come back later to fetch or draw results
    • Results are saved on the server
    • large results are best left on the server
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetToolNames</td>
<td>Return the names of the tools in the service.</td>
</tr>
<tr>
<td>GetToolInfo</td>
<td>Returns the signature and default parameter values for a tool.</td>
</tr>
<tr>
<td>GetExecutionType</td>
<td>Return the type of execution: synchronous or asynchronous.</td>
</tr>
<tr>
<td>Execute</td>
<td>Execute a synchronous geoprocessing tool.</td>
</tr>
<tr>
<td>SubmitJob</td>
<td>Execute an asynchronous geoprocessing tool.</td>
</tr>
<tr>
<td>GetJobStatus</td>
<td>Return the current status of a geoprocessing job.</td>
</tr>
<tr>
<td>GetJobMessages</td>
<td>Return verbose messages representing the current state of a submitted geoprocessing job.</td>
</tr>
<tr>
<td>GetJobResults</td>
<td>Return the results of a geoprocessing job that has completed successfully.</td>
</tr>
</tbody>
</table>
Geoprocessing Demo – interrogate a service, use a clip tool in the service
Geodata Services and Replication

• **Geodata Services** allow you to publish a geodatabase so that it can be accessed remotely over the Web

• **ArcGIS Desktop** can be used as a client with both local and remote geodatabases in order to
  – Extract data
  – Create replicas
  – Synchronize replicas
Working with Geodata Services in ArcGIS Desktop

- All the distributed database tools work with local geodatabase connections as well as with geodata services ("remote geodatabase connections")

- GIS Analysts can also use GP scripting to setup automated synchronization jobs that automatically synchronize geodatabases at periodic intervals
**Geodata Service**

- **Methods for**
  - browsing datasets
  - exporting datasets to XML
  - creating check-out, 1 way and 2 way replicas that can be downloaded and edited
  - exporting and importing replica changes.
Geodata Service – Object Model

- GeodataServer
- GPReplicaDescription
- GPReplicaDataset
- GDSExportOptions
- GDSData
Geodata Service demo – data extraction
Geodata Service - Creating a replica

//create array of datasets
GPReplicaDataset[] repDSs = new GPReplicaDataset[2];

//fill it up w/datasets to replicate
GPReplicaDataset repDS = new GPReplicaDataset();
repDS.DatasetName = "Parcels";
repDS.DatasetType = esriDatasetType.esriDTFeatureClass;
//selection IDs, UseGeometry, DefQuery...
repDSs[0] = repDS;

repDS = new GPReplicaDataset();
repDS.DatasetName = "Buildings";
repDS.DatasetType = esriDatasetType.esriDTFeatureClass;
repDSs[1] = repDS;

//setup a replicaDescription
GPReplicaDescription repDesc = new GPReplicaDescription();
repDesc.GPReplicaDatasets = repDSs;
repDesc.QueryGeometry = myEnvelope;
repDesc.SingleGeneration = true; //ie checkOut ...
Geodata Service - Creating a replica

//extra options
GPReplicaOptions repOpts = new GPReplicaOptions();
repOpts.AccessType = esriReplicaAccessType.esriReplicaBothReadWrite;

//how to format the data
GDSExportOptions expOpts = new GDSExportOptions();
expOpts.ExportFormat = esriGDSExportFormat.esriGDSExportFormatFileGDB;

//how to transfer the data
esriGDSTransportType transpType =
    esriGDSTransportType.esriGDSTransportTypeUrl;

//create the replica
GDSData data =
    m_gds.CreateReplica(sdeVersion, repName, repDesc, repOpts, expOpts, transpType);
DownloadFile(data.URL, outFGdb);

//m_gds.ExtractData with similar signature will extract the data (no replica created)
//export from source gds
GDSData changes
   = src_gds.ExportReplicaDataChanges(repName, expOpts, transpType,
   esriExportGenerationsOption.esriExportGenerationsAll, switchRoleYes);

//import into destination gds
dest_gds.ImportReplicaDataChanges(
   esriGDSReplicaImportSource.esriGDSReplicaImportSourceDeltaFileGDB,
   esriReplicaRecPolicyType.esriResolveConflictsInFavorOfDatabaseChanges,
   colLevelYes, changes);

//repeat in the opposite direction
Developing Desktop and ArcEngine Clients to the GeodataService

- Use the GeoDataServer and ReplicationAgent proxy classes that are part of the GeodatabaseDistributed ArcObjects assembly

- ReplicationAgent gives you single call synchronization
ArcGIS Server Web Services – Under the Hood

- SOAP requests received by the web server are handled by a custom HTTP handler [.Net] / Servlet [Java] and forwarded to the corresponding server object in the GIS Server for processing.

- ArcGIS includes its own SOAP stack and XML serialization framework for ArcObjects

- All parameter objects implement IXMLSerialize

- SOAP requests [the SOAP body] is deserialized and processed by the server object

- Results are serialized into the corresponding SOAP response
POST(url, payload)

http://myServer/ArcGIS/services/USA/MapServer

<soap:Envelope xmlns:soap="http://..."
xmlns:xsi="http://..."
xmlns:xs="http://..."
xmlns:tns="http://ArcGIS/9.2">
<soap:Body>
<tns:GetDefaultMapName/>
</soap:Body>
</soap:Envelope>

getContext(soName, soType)

getServerObject(soName, soType, so)

handleStringReq(req, resp)

resp is returned

releaseContext()
Working with SOAP requests over the LAN

- Web ADF includes both a Web proxy as well as a LAN proxy
- Web service proxy generated from WSDL - MapServerProxy
  - Used for SOAP / HTTP
- LAN proxy is also included - MapServerDCOMProxy
  - Used for SOAP / LAN
  - Uses the IRequestHandler interface to submit SOAP requests directly to the server object
Summary

• ArcGIS Server has a rich set of GIS Web Services

• You can work with services using the SOAP API

• The SOAP API is
  – easily accessible from .Net and Java
  – has a well described contract
  – is fully integrated into IDE’s
  – lets you take advantage of compile time type checking
In Conclusion...

• Other recommended sessions and meetings
  – Using the ArcGIS Server REST API

• 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