Building Web Services with ArcGIS Server and Java

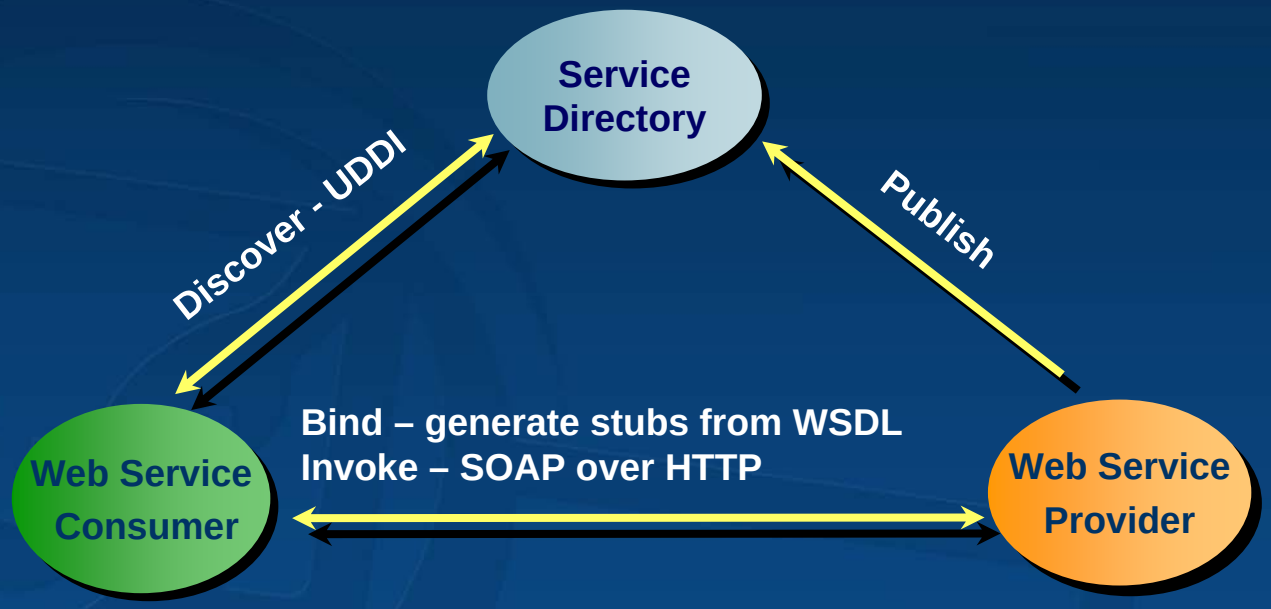
Eric Miller
Antony Jayaprakash
Jay Theodore
Steven Citron-Pousty

Overview

- Web Services Overview
- ArcGIS Server Web Services
- Creating and Consuming ArcGIS Server Web Services
- Application Web Services
- Creating and Consuming Application Web Services
- New at 9.2
- API references and Documentation

Web Services Overview

Web Service: a software component, described via WSDL, that is accessible via SOAP over HTTP.



- **ArcGIS Server Clients**
  - ArcMap
  - ArcCatalog
- **Custom Clients**
  - Web Applications
  - Desktop Applications

- **ArcGIS Server Web Services**
  - Catalog
  - Map Server
  - Geocode Server
- **Application Web Services**
  - Custom service

AGS Web Services @ 9.1

- Exposes Server Objects as web services
  - MapServer
  - GeocodeServer
- Provides a pseudo discovery service
  - Web Service Catalog
    - Also a web service
    - Hosts server object web services
  - Published using web service catalog template.
- Interoperability
  - Consumed by both Java and .NET
- Standard web application security
  - Basic, Digest, Forms based authentication.
  - HTTPS channel encryption.

AGS Web Services @ 9.1(cont)

- Designed for stateless consumption by ArcGIS clients; SOAP responses can be used by the AGSClient ArcObjects library.
  - ArcMap
  - ArcCatalog
  - Custom clients

- Implemented using JSP pages.
  - SOAP request is forwarded to server

    HttpSoapHandler.sendSoapResponse(null,
    String soapRequest, IRequestHandler requestHandler, String out);

  - Each server object type implements IRequestHandler which handles SOAP input by parsing it into native ArcObjects requests.

ArcGIS Server (AGS) Web Services

Discovery and Request

Client Applications
- **ArcMap**
- **ArcCatalog**

1. SOAP request/response for URLs of hosted web services

2. SOAP request/response to/from MapServer Web Service

Catalog Web Service

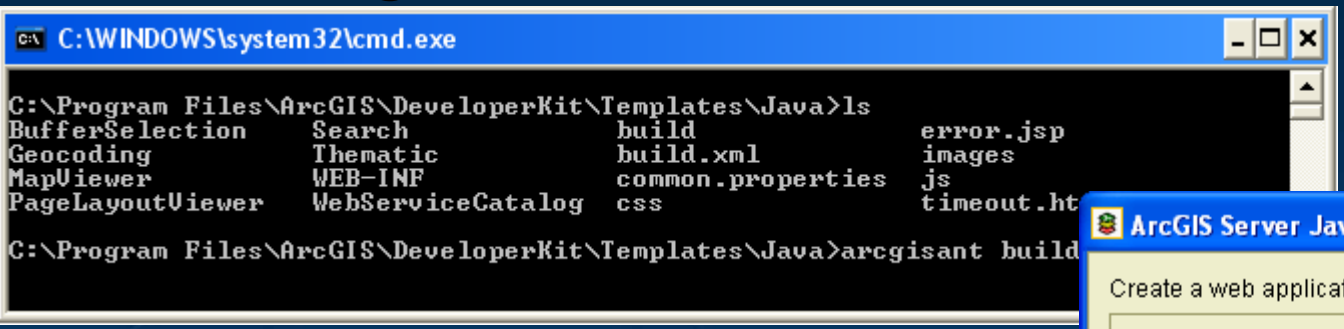
- USA_MapServer Web Service
- Conservation_MapServer Web Service
- Germany_MapServer Web Service
- Portland_GeocodeServer Web Service

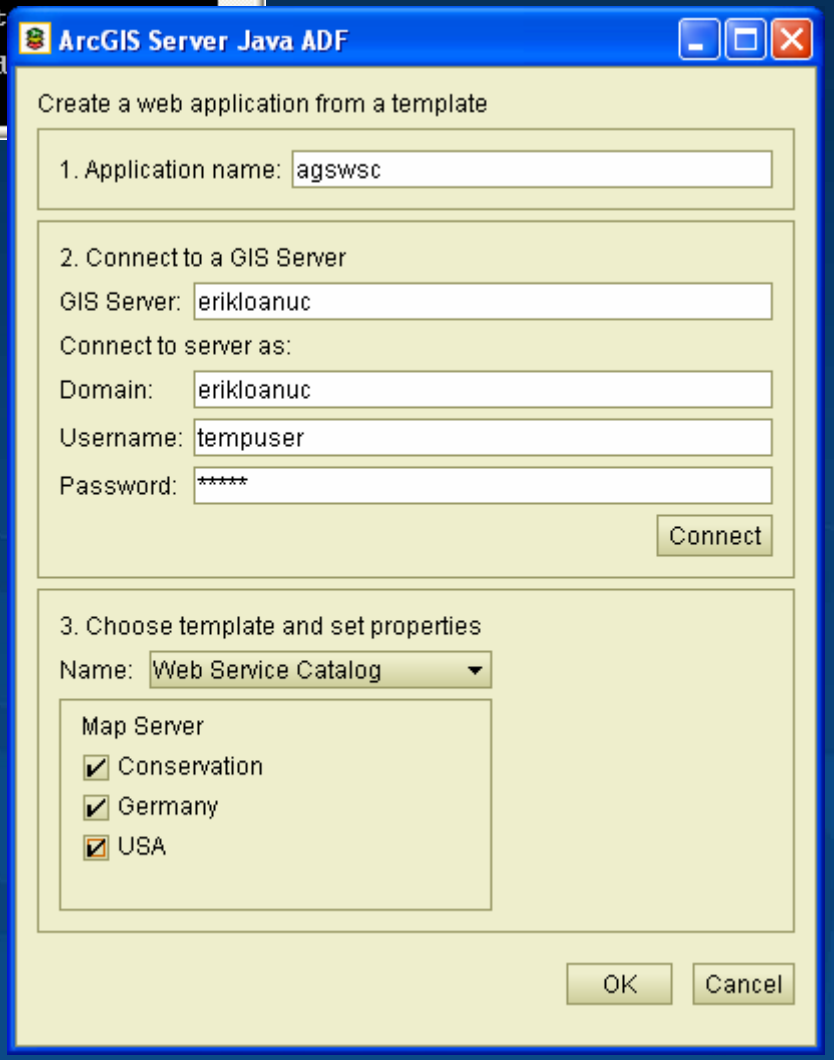
ArcGIS Server

3. DCOM Call with embedded SOAP request/response

4. Server Object Implements IRequestHandler

Creating AGS Web Services

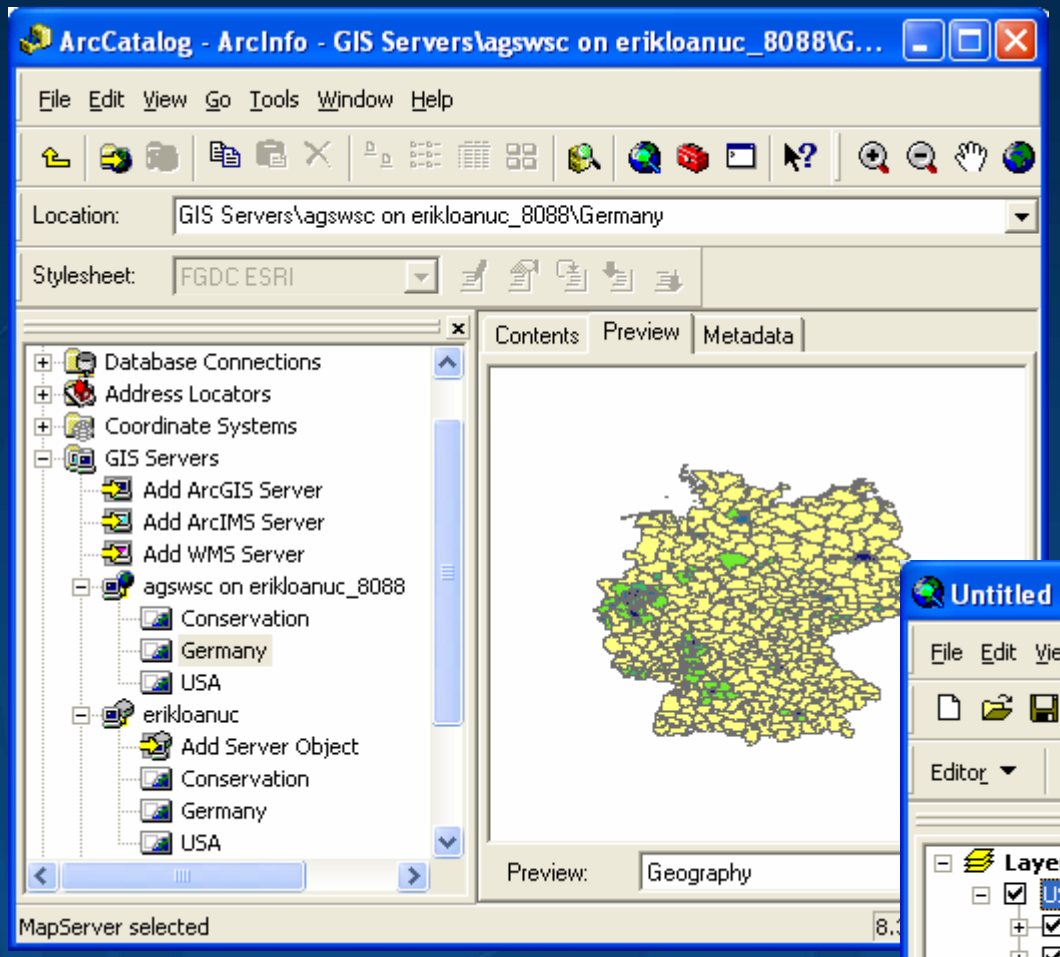


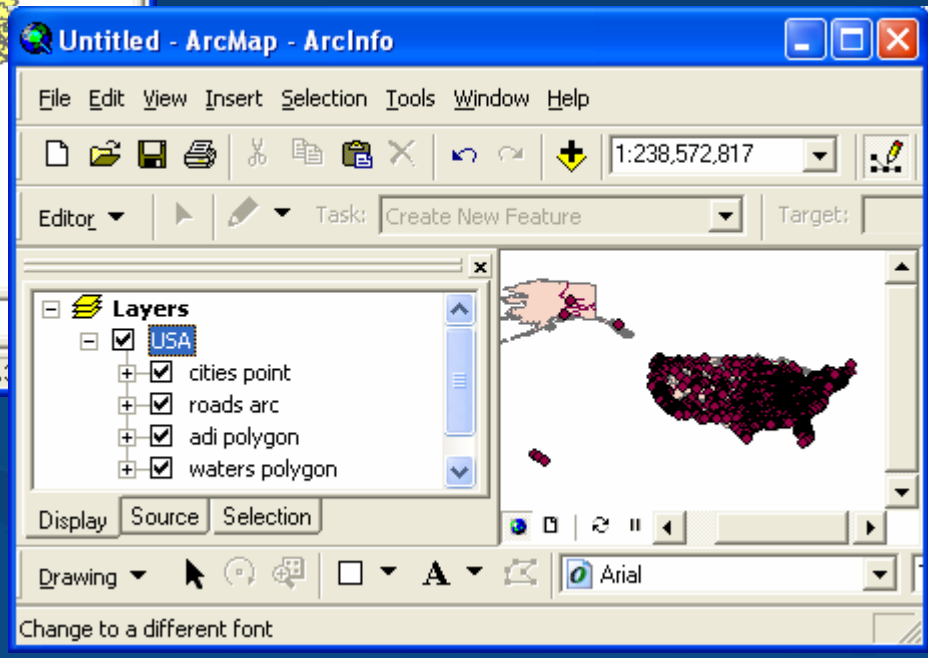


A simple matter of:

- Create a Server Object configuration.
- Start the Server object configuration.
- Create a web service catalog web service from template in Developer Kit.

Consuming AGS Web Services

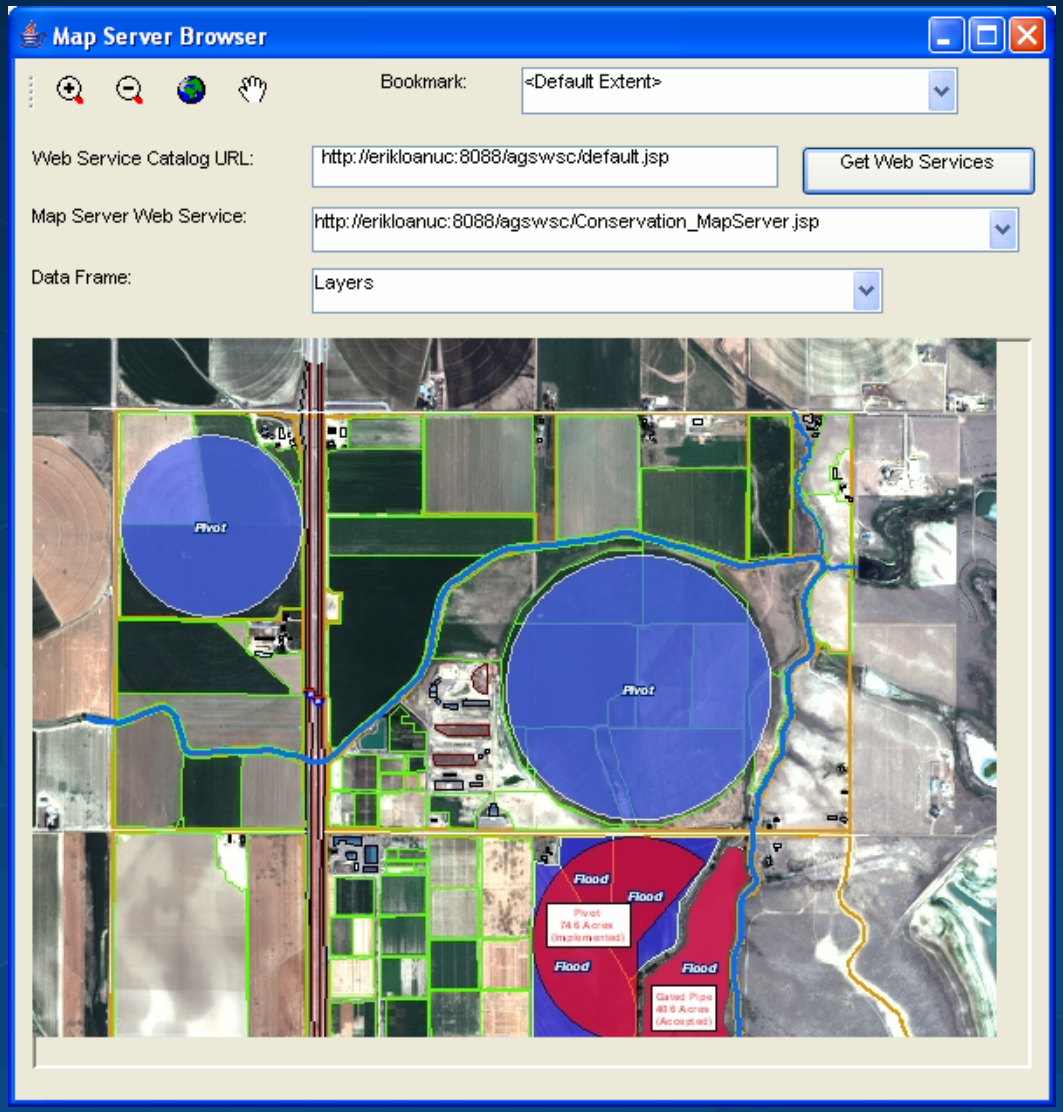




ArcGIS Desktop Clients

Consuming AGS Web Services (cont)

Custom Clients



Consuming AGS Web Services (cont)

Custom Client Implementation:

Get MapServer Web Service URLs from ServiceCatalog.

```
// get all MapServer web services and add their URLs
// to the MapServer JComboBox
URL url = new URL(urlTextField.getText());
ServiceCatalogBindingStub servicecatalog = (ServiceCatalogBindingStub)
        new _defaultLocator().getServiceCatalogPort(url);
ArrayOfServiceDescription sdArray =
        servicecatalog.getServiceDescriptions();
ServiceDescription sd[] = sdArray.getServiceDescription();

// add MapServers to the JComboBox
for (int i = 0; i < sd.length; i++) {
        if ( sd[i].getType().equals("MapServer") ) {
                mapserverComboBox.addItem(sd[i].getUrl());
        }
}
```

Consuming AGS Web Services (cont)

Custom Client Implementation:

Get data frames from MapServer Web Service.

```
// Connect to MapServer web service
MapServerBindingStub mapserver =
        (MapServerBindingStub) new MapLocator().getMapServerPort(url);
dfComboBox.removeAllItems();
int numMaps = mapserver.getMapCount();
for (int i = 0; i < numMaps; i++) {
        dfComboBox.addItem(mapserver.getMapName(i));
}
```

Consuming AGS Web Services (cont)

Custom Client Implementation:

Get map document bookmarks from MapServer Web Service.

```
MapServerBindingStub mapserver = (MapServerBindingStub)
        new MapLocator().getMapServerPort(url);
String name =
        mapserver.getMapName(dfComboBox.getSelectedIndex());
MapServerInfo mapInfo = mapserver.getServerInfo(name);
ArrayOfMapServerBookmark bookmarkArray =
        mapInfo.getBookmarks();
MapServerBookmark[] bookmarks =
        bookmarkArray.getMapServerBookmark();
if ( bookmarks != null ) {
        for (int i = 0; i < bookmarks.length; i++) {
            bookmarkComboBox.addItem(bookmarks[i].getName());
        }
}
```

Consuming AGS Web Services (cont)

Custom Client Implementation:

Draw map at extent of map document bookmark.

```
MapServerInfo mapInfo = mapserver.getServerInfo(name);
MapDescription mapDesc = mapDescription;
ArrayOfMapServerBookmark bookmarkArray = mapInfo.getBookmarks();
MapServerBookmark[] bookmarks =
        bookmarkArray.getMapServerBookmark();
if ( bookmarks != null ) {
        for (int i = 0; i < bookmarks.length; i++) {
                if (bookmarks[i].getName().equals(bookmarkComboBox.
                        getSelectedItem().toString())) {
                mapDesc.setMapArea(bookmarks[i]);
                }
        }
}
mapDescription = mapDesc;
drawMap(mapDescription);
```

Consuming AGS Web Services (cont)

Custom Client Implementation:

Create an image by calling ExportMapImage().

```
ImageType it = new ImageType();
it.setImageFormat(EsriImageFormat.esriImageJPG);
it.setImageReturnType(EsriImageReturnType.esriImageReturnMimeData);
ImageDisplay idisp = new ImageDisplay();
idisp.setImageHeight(400);
idisp.setImageWidth(552);
idisp.setImageDPI(150);
ImageDescription iDesc = new ImageDescription();
iDesc.setImageDisplay(idisp);
iDesc.setImageType(it);
MapImage mi = mapserver.exportMapImage(mapDesc, iDesc);
byte[] data = mi.getImageData();
mapimage = java.awt.Toolkit.getDefaultToolkit().createImage(data);
```

Application Web Services

- Software components that accept and return simple non-ArcObjects Java types.
- Use the ArcObjects Java Server API to solve your business logic, but accept and return only simple Java types as input and output.
- Example:
  - Locate ideal locations to build a school by building farthest from toxic waste sites
    - [in] double x, [in] double y
    - [out] double distance

Creating Application Web Services

- Development plan

Step 1: Create core functionality

Step 2: Create a SOAP service wrapper

Step 3: Create WSDL service description

Step 4: Deploy service

Step 5: Register service via UDDI (optional)

Consuming Application Web Services

- Development plan

Step 1: Discover services (UDDI)

Step 2: Retrieve service description file (WSDL)

Step 3: Generate stubs and add to classpath

Step 4: Work with proxy stubs

New at 9.2 - AGS Web Services

- Exposes Server Objects and Extensions as web services
  - MapServer (KML, WMS, NA, Mobile), GlobeServer, GeocodeServer, GeoDataServer
- Capabilities can be restricted or enabled
  - MapServer – map, query, data
  - GeocodeServer – geocode, reversegeocode
  - GeoDataServer – query, extraction, replication
  - GlobeServer – globe, animation, query
- Provides a pseduo discovery service
  - Web Service Catalog
    - Also a web service
    - Hosts server object SOAP web services.
    - Hosts server object binary endpoints.
  - Published using standalone publisher (under development)

New at 9.2 - AGS Web Services (cont)

- Designed for stateless consumption by ArcGIS clients; SOAP responses can be used by the AGSClient ArcObjects library.
  - ArcMap
  - ArcGlobe
  - ArcExplorer 2
  - ADF Clients
  - ArcCatalog
  - Custom clients

- Implemented using a Java Servlet
  - SOAP request is forwarded to server

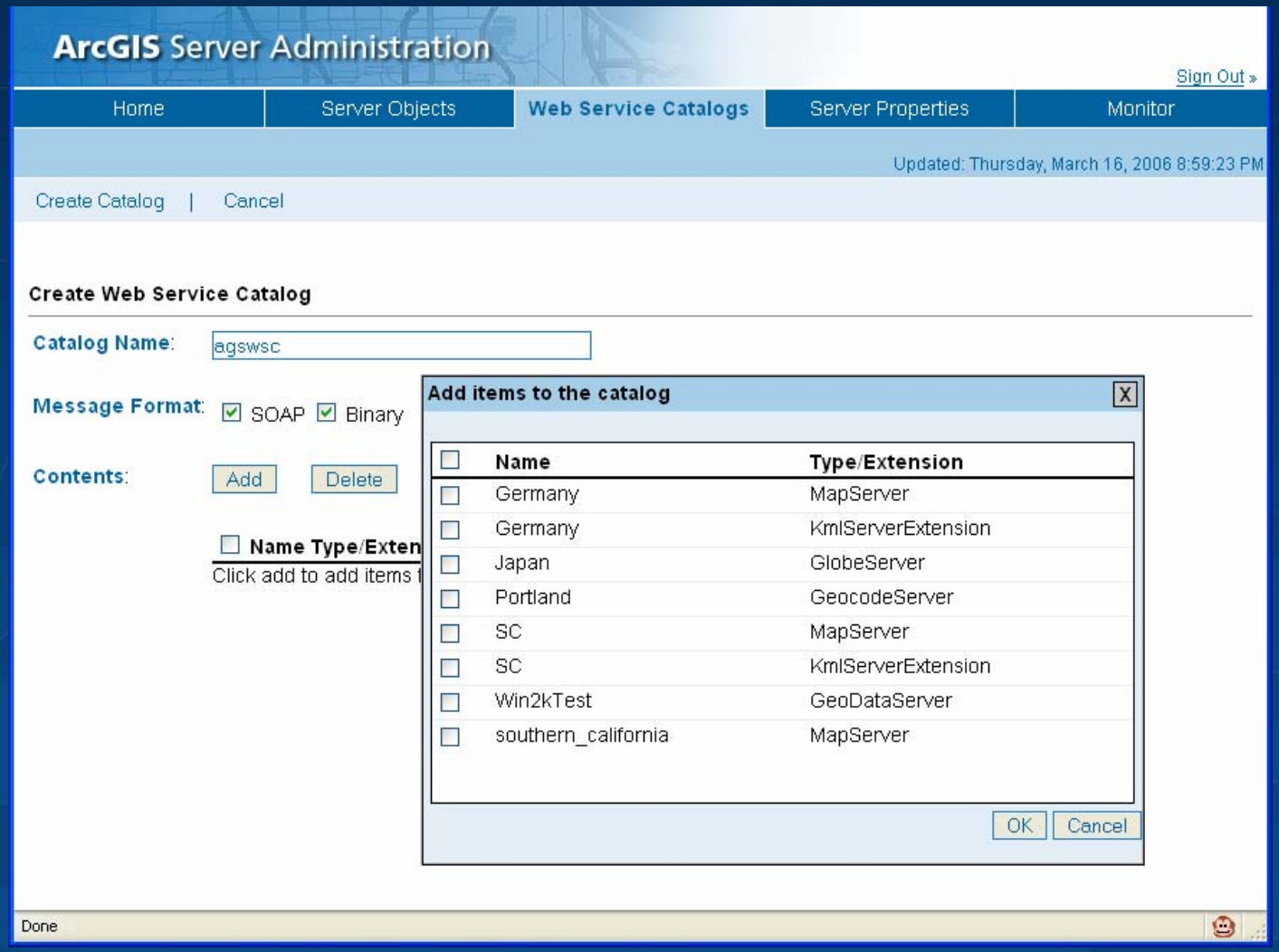
    HttpSoapHandler.sendSoapResponse(null,
    String soapRequest, IRequestHandler requestHandler, String out);

  - Each server object type implements IRequestHandler which handles SOAP input (IRequestHandler.HandleStringRequest()) by parsing it into native ArcObjects requests.

  - Binary requests are also forwarded to the server and handled by IRequestHandler.HandleBinaryRequest().

New at 9.2 - AGS Web Services (cont)

Publishing ArcGIS Server Web Services

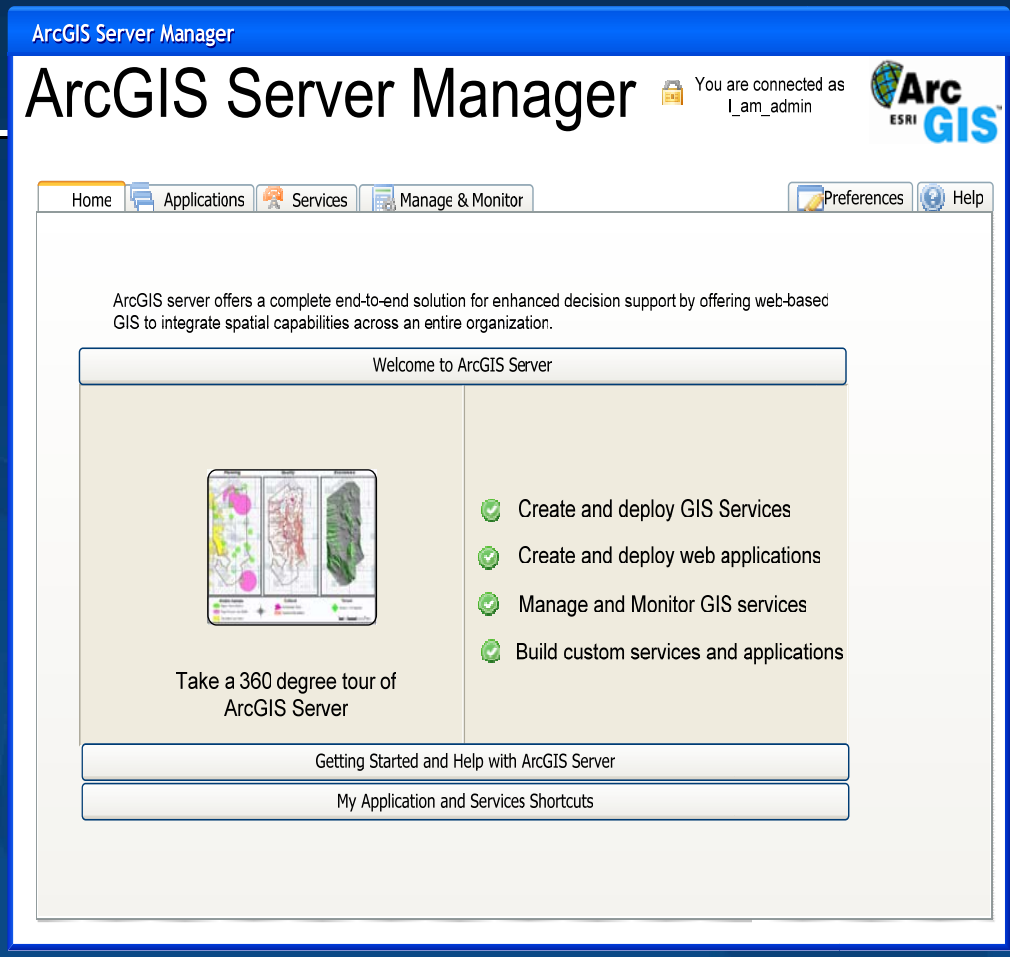


With ArcGIS Server Manager You Can:

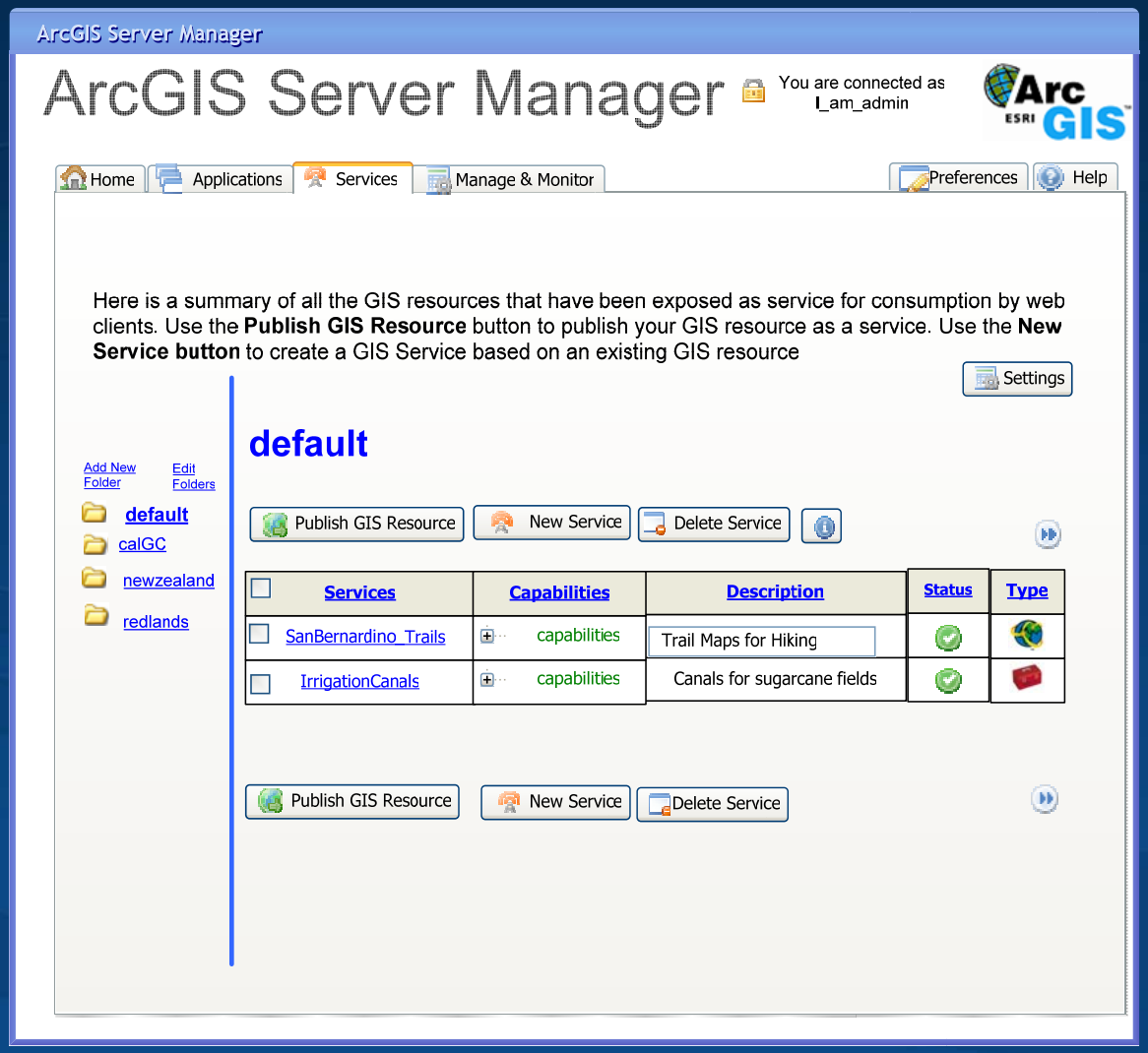
- Create and deploy GIS Services
- Create and deploy Web Applications
- Create and deploy Mobile Applications
- Manage and Monitor GIS Services
- Build custom services and applications

ArcGIS Server Manager: Complete Solution

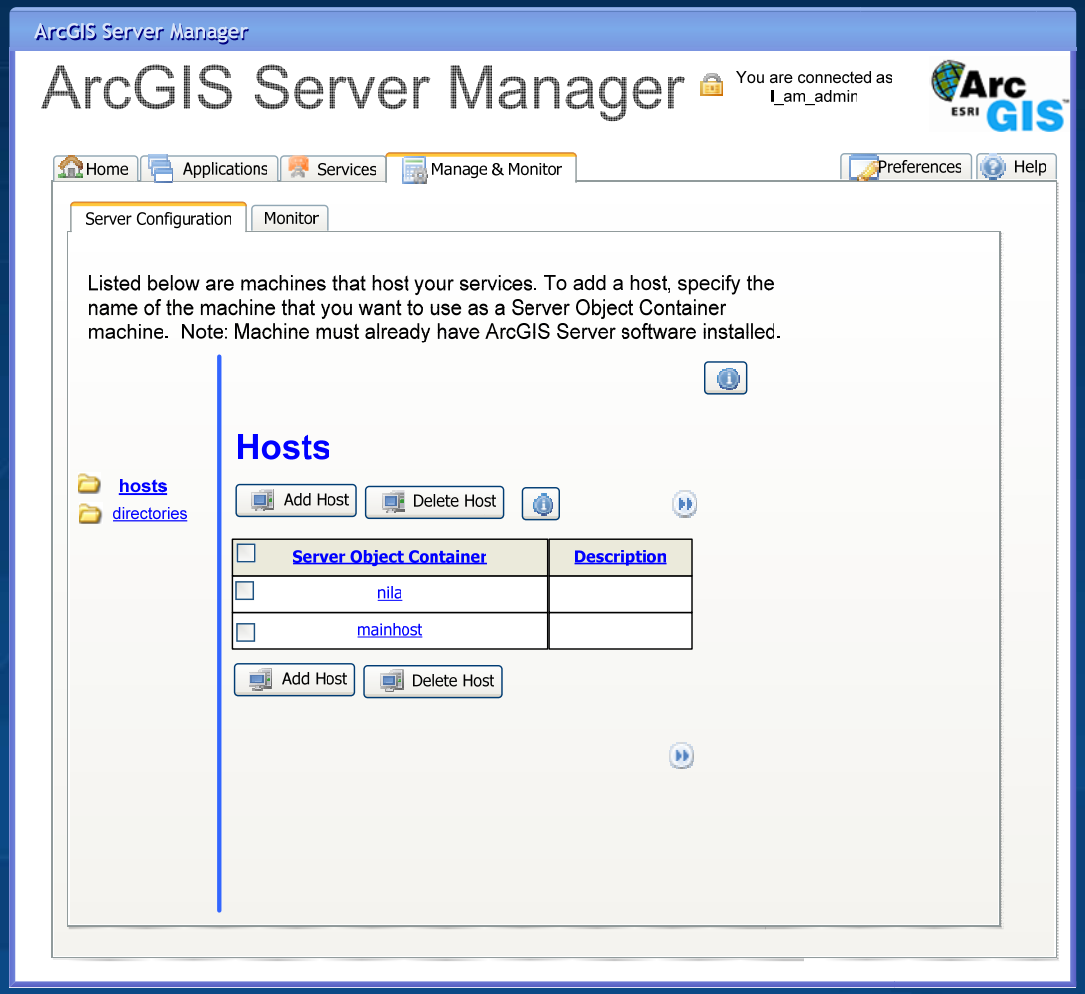
- Single consolidated view of ArcGIS Server
- Zero programming web-application builder
- Simplified publishing model just one-click away
- Centralized management and monitoring



Simplified Publishing
…..Just One-Click Away



Integrated Server Administration …Point and Click Ease



Features

- Easy to use Web Application Builder – zero programming
- Provides extensive support for Mapping, KML, OGC WMS, Network Analysis, Caching…
- Provides flexible deployment to J2EE environments
- Easy to customize applications
- Integrate GIS Services into Web Applications
- Access full power of ArcGIS Server within your applications
- Easy to use out-of-the box applications and tools

API References and Documentation

- JavaDoc generated from WSDL-generated stub classes.
- Match descriptions to similarly named classes in Server API.
- Developer scenarios
- Samples

Session Evaluations Reminder

Session Attendees:

Please turn in your session evaluations.

. . . Thank you