Automating Geodatabase Creation with Geoprocessing

Russell Brennan - @russellGIS
Ian Wittenmyer - @ianWittenmyer
Twitter: #gdbcreate
Assumptions

• Geodatabase fundamentals

• Experience with geoprocessing

• Understanding of geodatabase design
Agenda

• Geodatabase creation options
• Geoprocessing review
• Schema creation in ModelBuilder
• Making model tools
• Using Python
• Making schema changes
• Enterprise considerations
Why are we here today?

- Use Geoprocessing for:
  - Creating Geodatabases
  - Modifying Geodatabases
- Best practices for:
  - Model builder
  - Custom tools
Geodatabase Creation Options
Schema Creation Options

Pros vs Cons

- ArcMap/ArcCatalog wizards
- Data Models (Solutions/Templates)
- UML
- Geoprocessing
ArcCatalog wizards

ArcCatalog – Templates – UML – Geoprocessing

• Pro:
  - Create any supported dataset type
  - Modify schema in place
  - Familiar environment

• Con:
  - Non-repeatable
  - No scheduling
  - No documentation
Templates

ArcCatalog – Templates – UML – Geoprocessing

• Pro:
  - Design work has been done by industry experts
  - Well documented
  - Map & Apps samples

• Cons:
  - Generic
    - Need to customize?
  - Not available for every use case/industry
UML
ArcCatalog – Templates – UML – Geoprocessing

• Pro:
  - Documentation
  - Visualization
  - May be required to share with other (non-GIS) departments

• Con:
  - Specialized skillset
    - understand both the language/concepts and the application
  - Does not support all data types
Geoprocessing
ArcCatalog – Templates – UML – Geoprocessing

• Pros:
  - GIS pro’s are comfortable with gp
  - One environment for creation, loading, analysis
  - Customizable/flexible

• Cons:
  - Limited diagramming
  - Difficult to share with non GIS
Geoprocessing Review
What is geoprocessing?

• Suite of tools
  - Over 800 tools and functions
  - Analysis
  - Data conversion
  - Dataset creation

• Framework
  - Link tools together (ModelBuilder)
  - Share/publish
  - Script and customize
ModelBuilder

- Create, edit and manage models
- Re-run workflows
- Visual programming language
Model Elements
What tools should I use?

- Look in Data Management toolset
- Many tools to create geodatabase objects
ModelBuilder Demo 1
Review
ModelBuilder Review

Discover tools
Renaming model elements
Creating basic schema
Pop-ups for gathering info
Running a model
Schema Creation
What is schema?
Getting started...

Violations (table related to inspections via InspectionID Field

- InspectionID, long
- ViolationType, long, subtype field
  - Subtypes:
    - 0 = Administration
    - 1 = General fire precautions
    - 2 = Fire Service features
    - 3 = Fire protection systems
    - 4 = Means of egress
    - 5 = Misc Violations ((includes: Building services and systems (Fuel fired appliances), Fire resistance rated construction, Interior finish, decorative materials and furnishings))

- Comments
- Enable attachments for photos, videos etc

Requirements

Prototype
Getting started...

Violations (table related to inspections via InspectionID Field

- InspectionID, long
- ViolationType, long, subtype field
  - Subtypes:
    - 0 = Administration
    - 1 = General fire precautions
    - 2 = Fire Service features
    - 3 = Fire protection systems
    - 4 = Means of egress
    - 5 = Misc Violations (includes: Building services and systems (Fuel fired appliances), Fire resistance rated construction, Interior finish, decorative materials and furnishings))
- Comments
- Enable attachments for photos, videos etc

Requirements

Prototype
Getting started…

Violations (table related to inspections via InspectionID Field

- InspectionID, long
- ViolationType, long, subtype field
  - Subtypes:
    - 0 = Administration
    - 1 = General fire precautions
    - 2 = Fire Service features
    - 3 = Fire protection systems
    - 4 = Means of egress
    - 5 = Misc Violations ((includes: Building services and
      systems (Fuel fired appliances), Fire resistance rated
      construction, Interior finish, decorative materials
      and furnishings))
- Comments
- Enable attachments for photos, videos etc

Requirements

Prototype
Getting started…

Violations (table related to inspections via InspectionID Field

- **InspectionID**, long
- **ViolationType**, long, subtype field
  - Subtypes:
    - 0 = Administration
    - 1 = General fire precautions
    - 2 = Fire Service features
    - 3 = Fire protection systems
    - 4 = Means of egress
    - 5 = Misc Violations ((includes: Building services and systems (Fuel fired appliances), Fire resistance rated construction, Interior finish, decorative materials and furnishings))
- Comments
- Enable attachments for photos, videos etc

Requirements

Prototype

Deployment
Model Tools
Model tools

- Built in tools = good but limited
- Model tools
  - Implement custom behavior
  - Group of tasks
  - Run like system tools
Why create model tools?

- Reduce clutter, improve readability
  - Fewer tools

- Reduce data entry
  - Fewer parameters to change
Growing pains
Organizing your tools
Turning models into model tools

- ‘Model parameter’
  - Allow tools to consume other tools
- Can be any parameter within the tool
- Input and output (derived) parameters
Creating a model tool
Creating a model tool
Creating a model tool
Creating a model tool

Creating Feature Class Model

- Workspace
- Feature Class Name
- Create Feature Class
- Add Field
- Output Feature Class
- OutputFC

Empty Model

CreateFCModel
Creating a model tool
Creating a model tool
Creating a model tool

- Workspace
- Feature Class Name
- Create Feature Class
- Add Field
- Output Feature Class
- OutputFC

Workspace
C:/presentations/UC2013
Creating a model tool
Organizing your models

- Organize your models into logical groups
  - Feature datasets
  - Departments in your organization
  - Applications you are building
  - Domains*

- Makes schema modular
Model tool tip
Schema Creation with Model Tools
Schema Creation with Model Tools

Implement data model
Geodatabase behavior
Template Feature Class/Table
Model tools as sub models
Organization
Python
Script tools

• Get access to:
  - Programming logic
  - arcpy functions
  - Custom validation
Using Python for Geodatabase creation

- Improving user experience
- Focused functions
- Documentation
  - Reports
Using a table to run a tool

• Look for:
  - Redundant info
  - Tools running multiple times
• Use cursors to loop through a table to get parameters.
Reporting

- arcpy.List...
- arcpy.Describe
- Examine contents of geodatabase
- Get information that is not reported through UI
  - Not easily returned from UI
Python
Python

Table to fields

Reporting
Schema Changes
Data Modification
Schema Changes

- Over time your data model **will** change.
- Use additional models or scripts to push out updates.
- Deploy when appropriate.
Schema Changes

- Change documentation
- Schedule changes via simple scripts
- Report on dependencies
Domain Report Tool
# Name:    Domain Report
#
#---------------------------------------------------------------

```python
import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass" + "," + "FieldName" + "," + "FieldDomain" + 
"\n")  #write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('', 'feature') + ['']:  
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != ":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + 
"\n")
arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc,field.name,field.domain))
f.close()
```
```python
import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass " + "," + "FieldName " + "," + "FieldDomain " + "\n")  # write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('', 'feature') + ['']:  
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != "":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + "\n")
    arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc, field.name, field.domain))

f.close()
```
import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass" + "," + "FieldName" + "," + "FieldDomain" + "\n")  # write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('', 'feature') + [']:  # 
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != "":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + "\n")
arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc, field.name, field.domain))
f.close()
import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass" + "," + "FieldName" + "," + "FieldDomain" + "\n")  # write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('', 'feature') + ['', ']:
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != "":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + "\n")
arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc, field.name, field.domain))
f.close()}
```python
# Name: Domain Report
#
#-----------------------------------------------

import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass" + "," + "FieldName" + "," + "FieldDomain" + 
")  # write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb

fcs = []
for fds in arcpy.ListDatasets('', 'feature') + [""],
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string !="":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + 
")
arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc, field.name, field.domain))

f.close()
```
```python
# Name: Domain Report
#
#----------------------------------------

import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass" + "," + "FieldName" + "," + "FieldDomain" + "\n") #write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('', 'feature') + ['', '']:
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != ":"
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + "\n")
arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc, field.name, field.domain))
f.close()
```
```python
import arcpy, os

# get parameters from the tool
ingdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass\n" + "," + "," + "FieldDomain\n" + "," + "FieldDomain" + "," + "FieldDomain" + "," + "FieldDomain"
) #write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('"","feature") + ["":
    for fc in arcpy.ListFeatureClasses('"","',fcs):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != "":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + ","
            arcpy.AddMessage("FC=(0) FieldName=(1) Domain=(2)".format(fc,field.name,field.domain))
f.close()
```
import arcpy, os

# get parameters from the tool
inputdb = arcpy.GetParameterAsText(0)
outputDir = arcpy.GetParameterAsText(1)
outputName = arcpy.GetParameterAsText(2)

textFile = (outputDir + os.sep + outputName + ".csv")
f = open(textFile, "a")
f.write("FeatureClass" + "," + "FieldName" + "," + "FieldDomain" + "\n")  # write headers

# get a list all Feature Classes in a geodatabase, including inside Feature Datasets
arcpy.env.workspace = gdb
fcs = []
for fds in arcpy.ListDatasets('', 'feature') + ['', '):
    for fc in arcpy.ListFeatureClasses('', '', fds):
        fcs.append(os.path.join(fds, fc))

# based on the list of feature classes collected above, this will create a list of fields
# and the write out the feature class, field name and domain name.
for fc in fcs:
    fields = arcpy.ListFields(fc)
    for field in fields:
        # check to see if there is a value in the domain property, basically if string != "":
        if field.domain:
            f.write(fc + "," + field.name + "," + field.domain + "\n")
f.close()
Completed script DomainReport...
Succeeded at Thu Jul 04 14:29:07 2013 (Elapsed Time: 26.21 seconds)
<table>
<thead>
<tr>
<th>FeatureClass</th>
<th>FieldName</th>
<th>FieldDomain</th>
</tr>
</thead>
<tbody>
<tr>
<td>larbase/feature_linked_parcel_ann_1</td>
<td>HorizontalAlignment</td>
<td></td>
</tr>
<tr>
<td>geometicnetwork/prime_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/Secondary_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/bridging_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/circuit_bkr_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/circuit_bkr_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/StreetLights_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/StreetLights_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/StreetLights_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>GeometricNetwork/meters_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/switch_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/switch_1</td>
<td>Status</td>
<td>SwitchStatus</td>
</tr>
<tr>
<td>GeometricNetwork/transformers_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/Tapwire_1</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/Electric_Net_1_Junctions</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>GeometricNetwork/Redline_Line_1</td>
<td>INSPECTOR</td>
<td>Employee</td>
</tr>
<tr>
<td>GeometricNetwork/Redline_Line_1</td>
<td>COMMENTS</td>
<td>StreetlightStatus</td>
</tr>
<tr>
<td>GeometricNetwork/WorkRequest_1</td>
<td>Work_Type</td>
<td>WorkType</td>
</tr>
<tr>
<td>GeometricNetwork/WorkRequest_1</td>
<td>Employee</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology/Parcels_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology/Parcels_1</td>
<td>Comments</td>
<td>WeedAbatement</td>
</tr>
<tr>
<td>Topology/selected_parcel_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology/selected_parcel_1</td>
<td>Comments</td>
<td>WeedAbatement</td>
</tr>
<tr>
<td>Topology/address_loc_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology/address_loc_1</td>
<td>Comments</td>
<td>WeedAbatement</td>
</tr>
<tr>
<td>RelationshipClassAttributed_bapOriginFC</td>
<td>LandUse</td>
<td>LandUse</td>
</tr>
<tr>
<td>RelationshipClassSimple_bapSimpleOriginFC</td>
<td>LandUse</td>
<td>LandUse</td>
</tr>
<tr>
<td>RelationshipClassComposite_bapCompositeOriginFC</td>
<td>LandUse</td>
<td>LandUse</td>
</tr>
<tr>
<td>RelationshipClassComposite_bapACrogInFCC</td>
<td>LandUse</td>
<td>LandUse</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSAnno</td>
<td>Status</td>
<td>AnnotationStatus</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSAnno</td>
<td>Bold</td>
<td>BooleanSymbolValue</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSAnno</td>
<td>Italic</td>
<td>BooleanSymbolValue</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSAnno</td>
<td>Underline</td>
<td>BooleanSymbolValue</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSAnno</td>
<td>VerticalAlignment</td>
<td>VerticalAlignment</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSAnno</td>
<td>HorizontalAlignment</td>
<td>HorizontalAlignment</td>
</tr>
<tr>
<td>cttNDSFS/cttCEFON_NG</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>cttNDSFS/cttNDSFS_Net_Junctions</td>
<td>Enabled</td>
<td>EnabledDomain</td>
</tr>
<tr>
<td>Representations</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Representations</td>
<td>Comments</td>
<td>WeedAbatement</td>
</tr>
<tr>
<td>Representations</td>
<td>RuleID</td>
<td>Representations_Rep_Rules</td>
</tr>
<tr>
<td>Representations</td>
<td>RuleID_1</td>
<td>Representations_Rep1_Rules</td>
</tr>
<tr>
<td>FeatureClass</td>
<td>FieldName</td>
<td>FieldDomain</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Annotation\Notes_Arco</td>
<td>UserName</td>
<td>Employee</td>
</tr>
<tr>
<td>Annotation\Notes_line</td>
<td>UserName</td>
<td>Employee</td>
</tr>
<tr>
<td>Annotation\Notes_Points</td>
<td>UserName</td>
<td>Employee</td>
</tr>
<tr>
<td>Annotation\Notes_Poly</td>
<td>UserName</td>
<td>Employee</td>
</tr>
<tr>
<td>landbase\Parcels</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>landbase\Potholes</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>landbase\address_loc_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>landbase\selected_parcles_1_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>GeometricNetwork\StreetLights_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>GeometricNetwork\Redline_Line_1</td>
<td>tNSPECTOR</td>
<td>Employee</td>
</tr>
<tr>
<td>GeometricNetwork\WorkRequest_1</td>
<td>Employee</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology\Parcels_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology\selected_parcles_1_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Topology\address_loc_1_1</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
<tr>
<td>Representations</td>
<td>Inspector</td>
<td>Employee</td>
</tr>
</tbody>
</table>
Scheduling

- Important to reduce downtime.
- Simple to schedule using OS scheduler
Modifying Geodatabase
Modifying Geodatabase

Inline variables
Scripting models
Schedule changes
Enterprise Geodatabases
Considerations for enterprise

- Data ownership
- Privileges
- Managing Versions
- Registering data as versioned
- Replicas
Limitations

- Create Network Datasets
- Parcel Fabrics
- Create Annotation
- Create Schematic Dataset
Final Thoughts

- Geoprocessing creates full featured geodatabases
- Use model tools and python
- Make schema changes easily
- Use Python to report
Other Sessions

Geodatabase Sessions

• www.esriurl.com/uc13geodata

Model Builder Sessions

• www.esriurl.com/uc13mb

Python Sessions

• www.esriurl.com/uc13python
Thank you...

Please fill out the session evaluation

Tuesday ID: 1266
Thursday ID: 1452

Online – www.esri.com/ucsessionssurveys
Paper – pick up and put in drop box