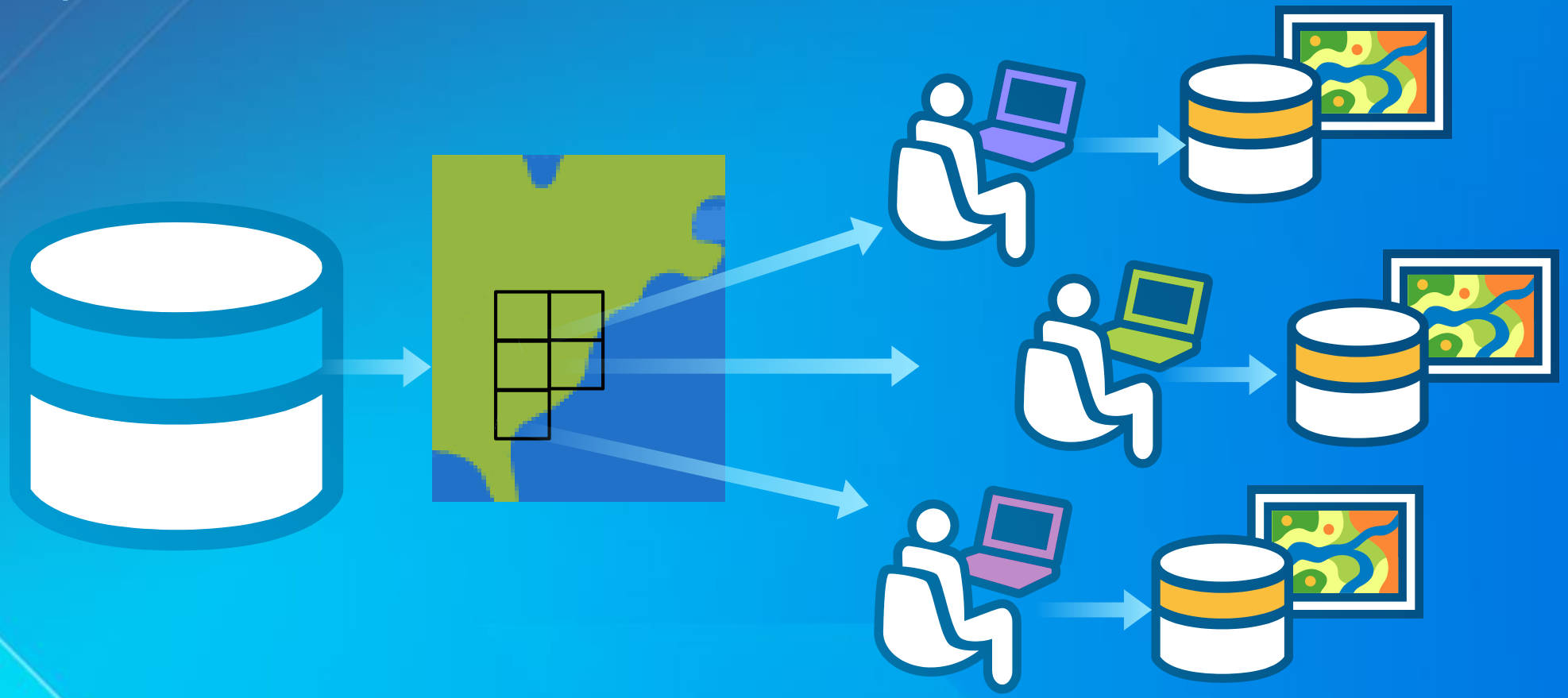
Esri Production Mapping: Map Automation and Advanced Cartography

Amber Bethell

Joe Sheffield

Traditional Cartography

With ArcMap

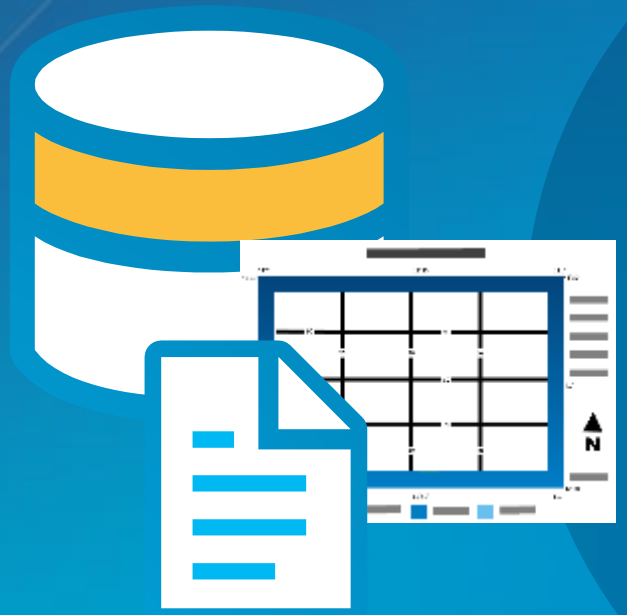


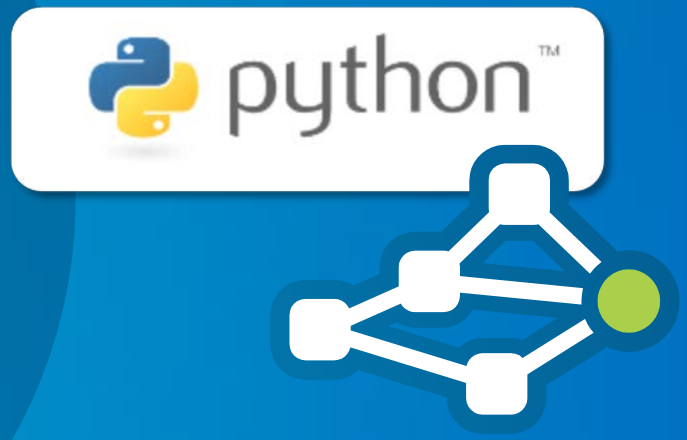
What you really want

Map Automation

Production Mapping
Cartographic Rules

Map Automation

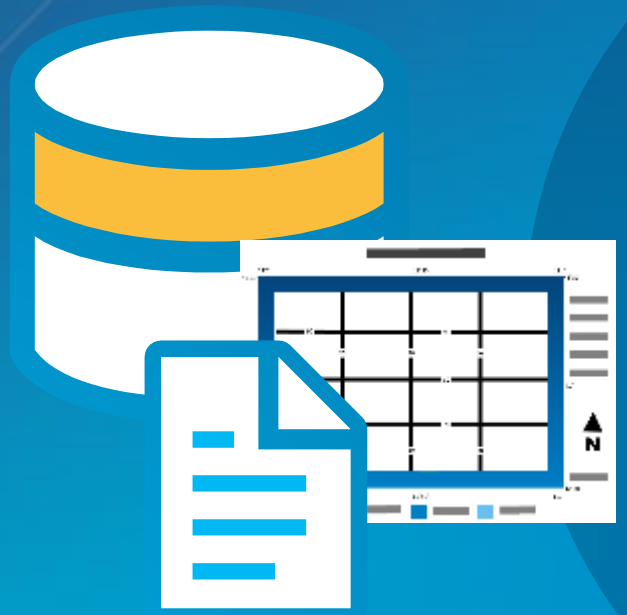




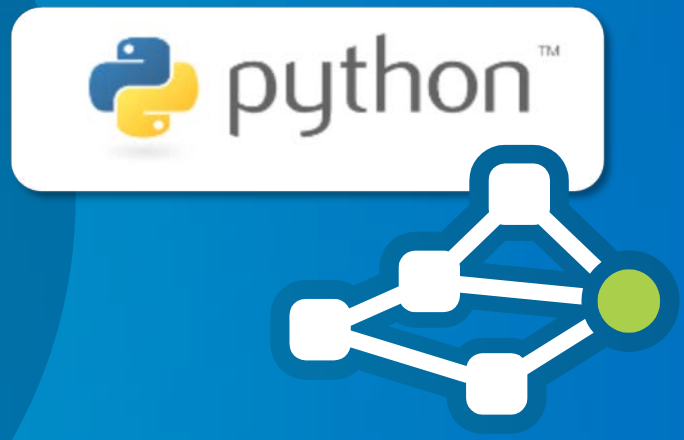
Production Mapping
Cartographic Rules

Geoprocessing
& Python

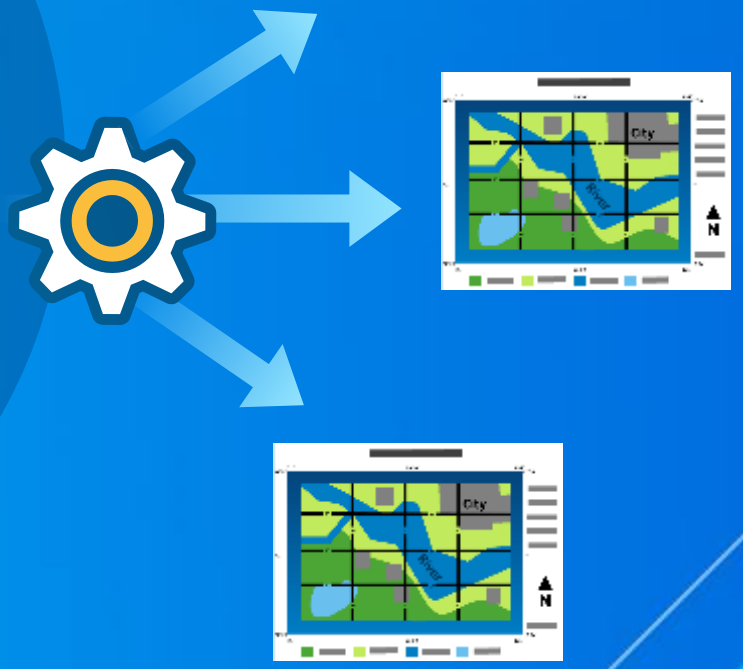
Map Automation



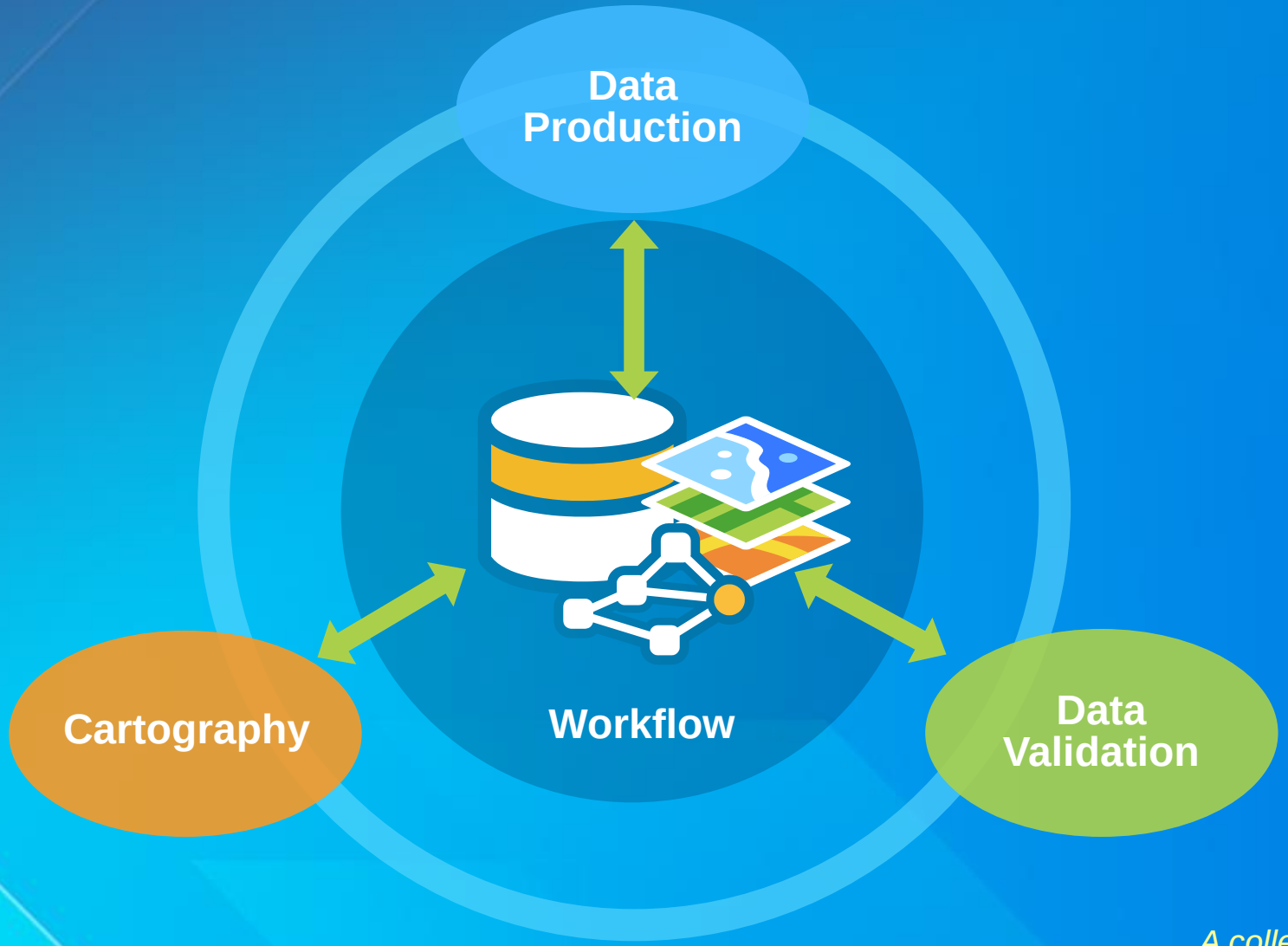
Production Mapping
Cartographic Rules



Geoprocessing
& Python

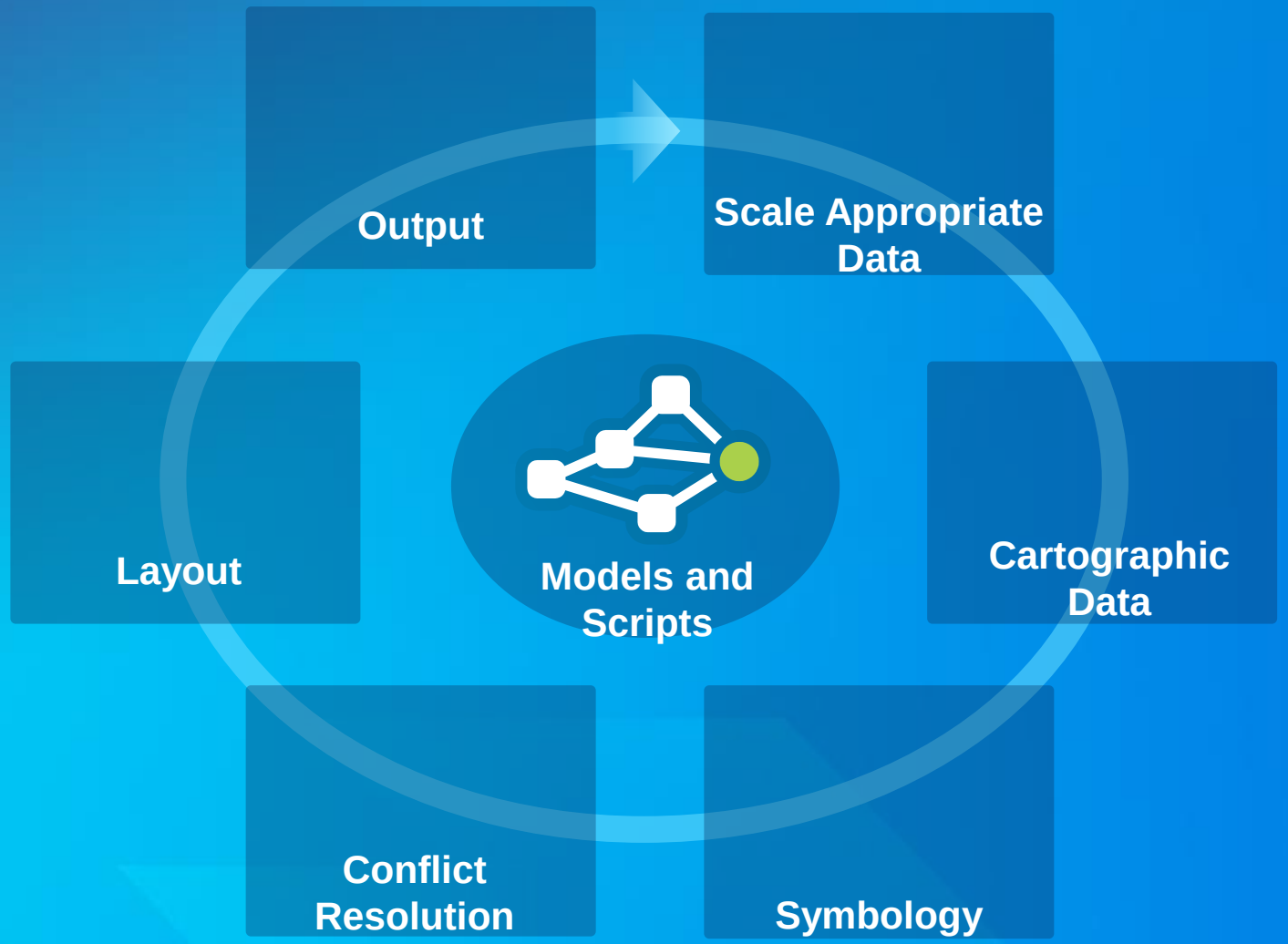


Esri Production Mapping

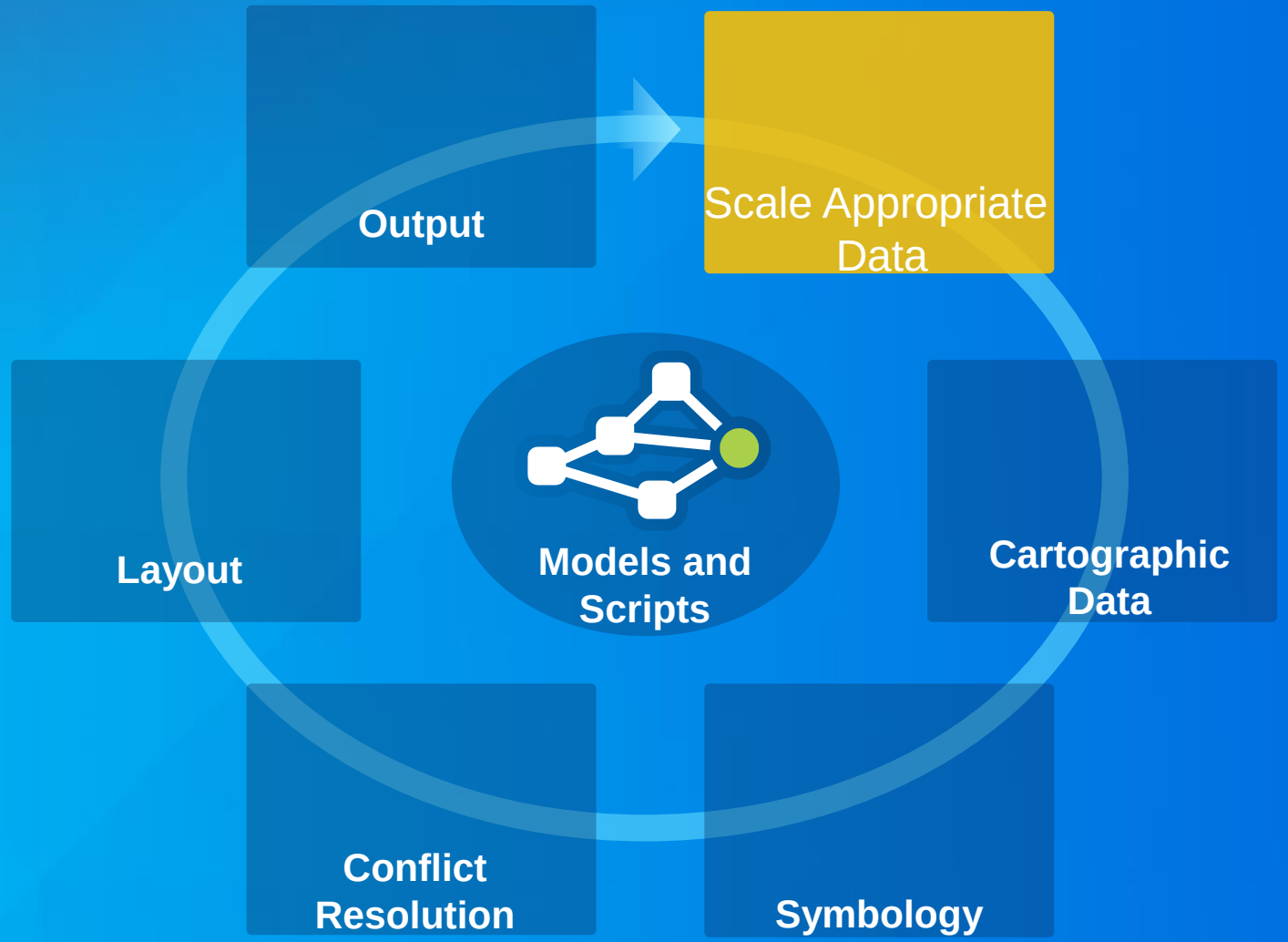


A collection of ArcMap Extensions

Cartographic Workflow



Scale Appropriate Data



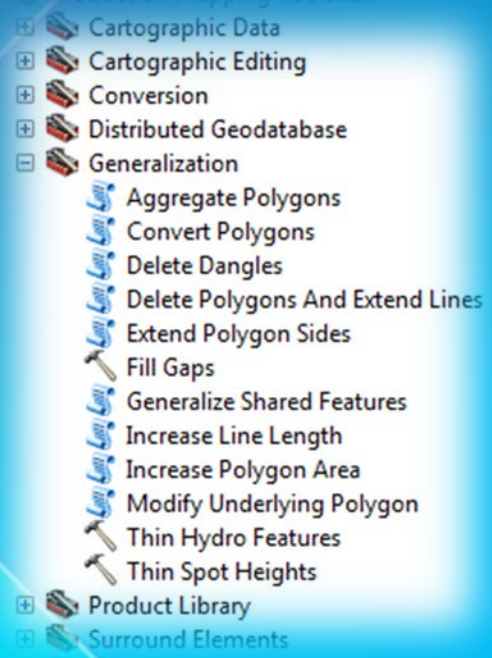
Generalization

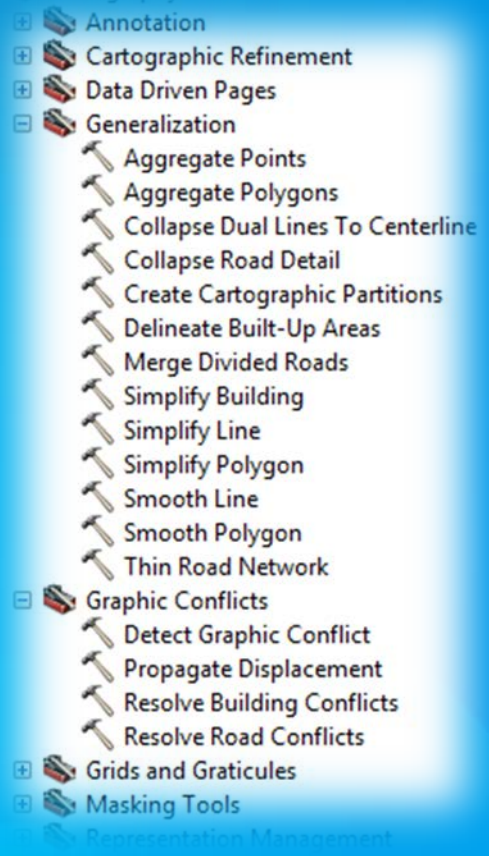
- Data is collected and maintained in high detail
- When drawn at a smaller scale:
  - it can be too detailed
  - symbols can conflict

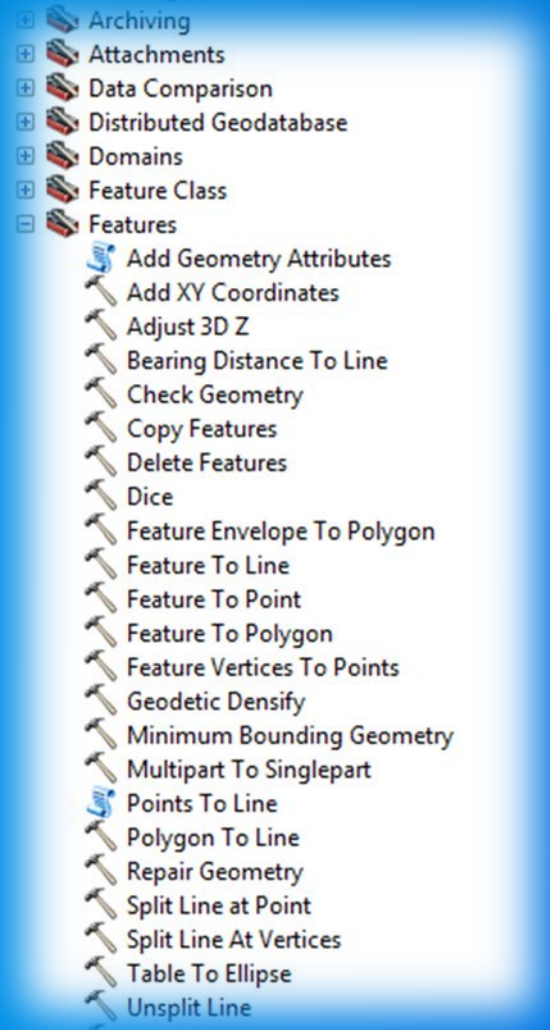
Generalization

- Clarifies the display of feature geometry at smaller scales

Generalization Tools



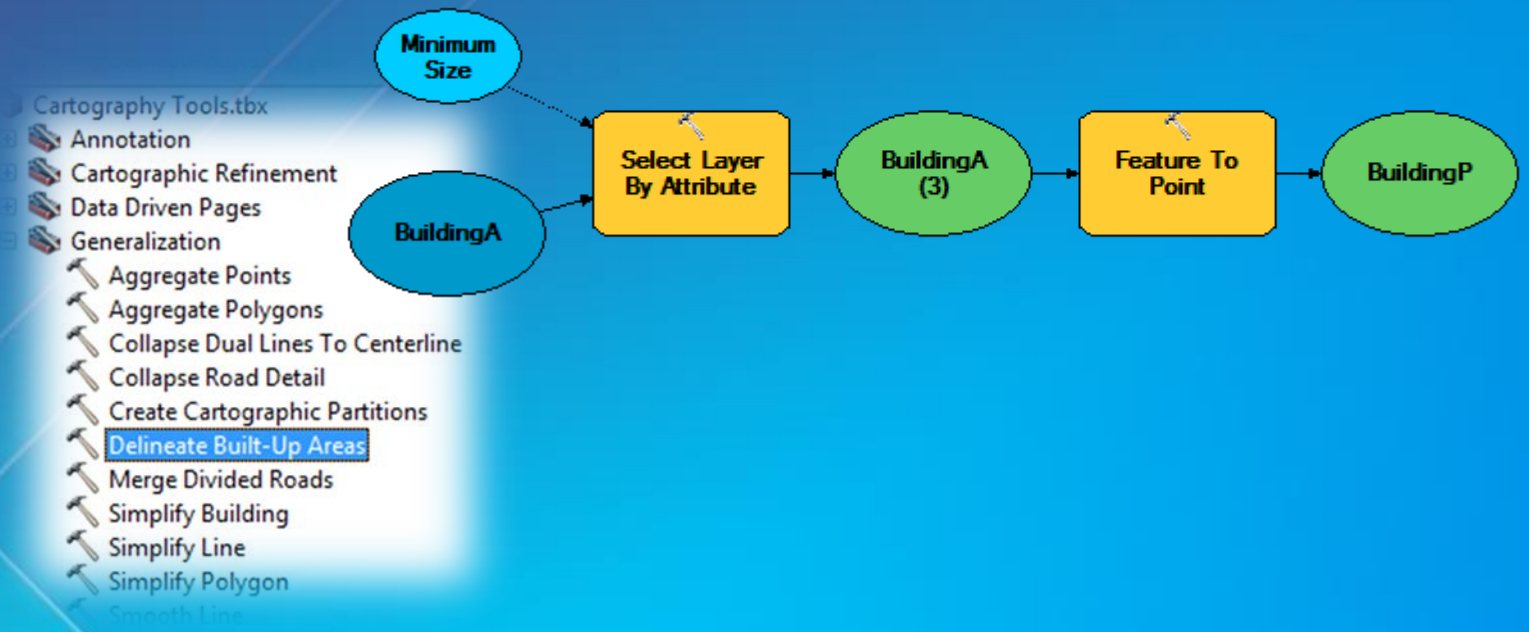




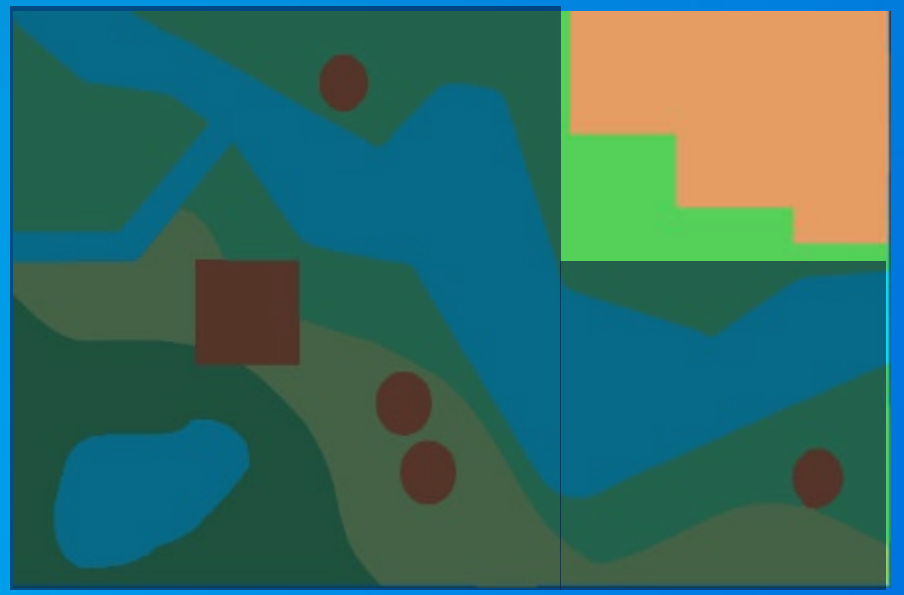
Feature Generalization

- Features assessed individually without regard to symbology or spatial relationships

Find the tool and add to a model

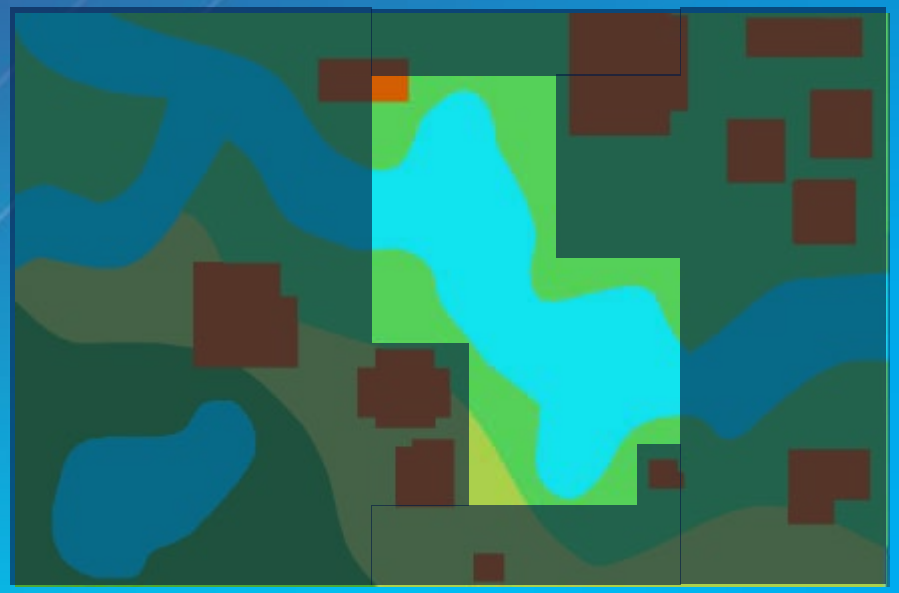


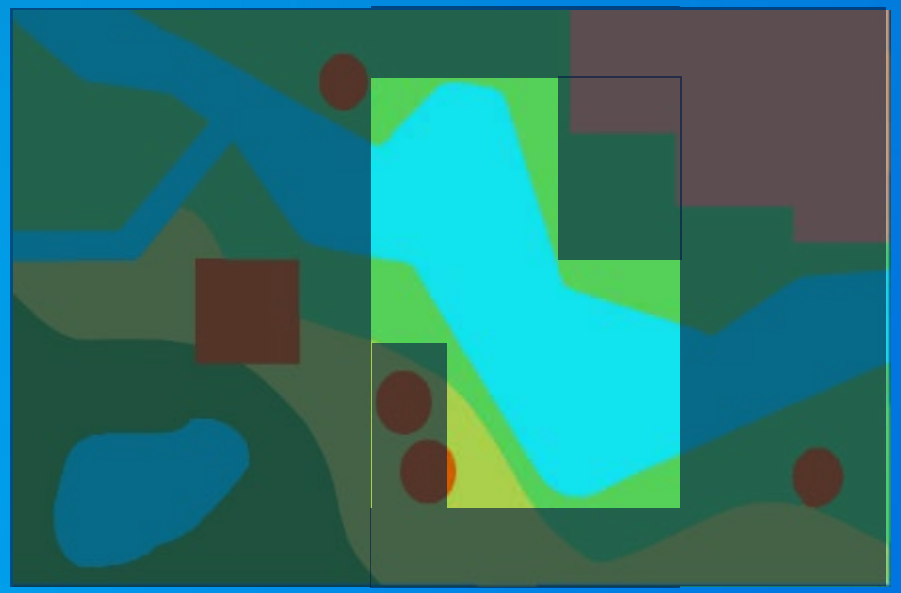
Contextual Generalization



- **Features from multiple layers assessed simultaneously**
  - **Maintain pattern, density, and spatial relationships**

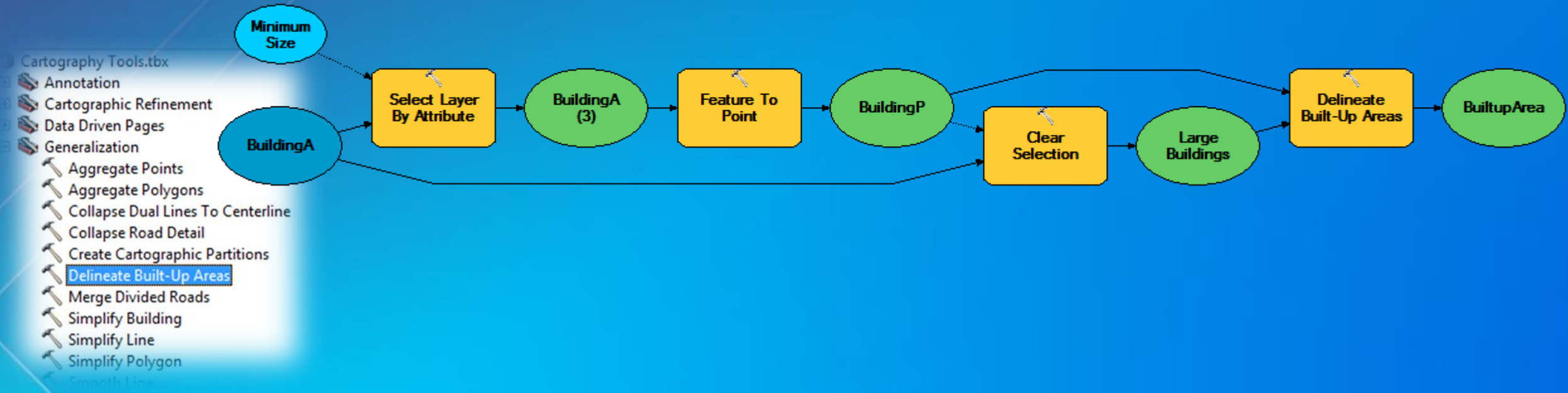
Contextual Generalization



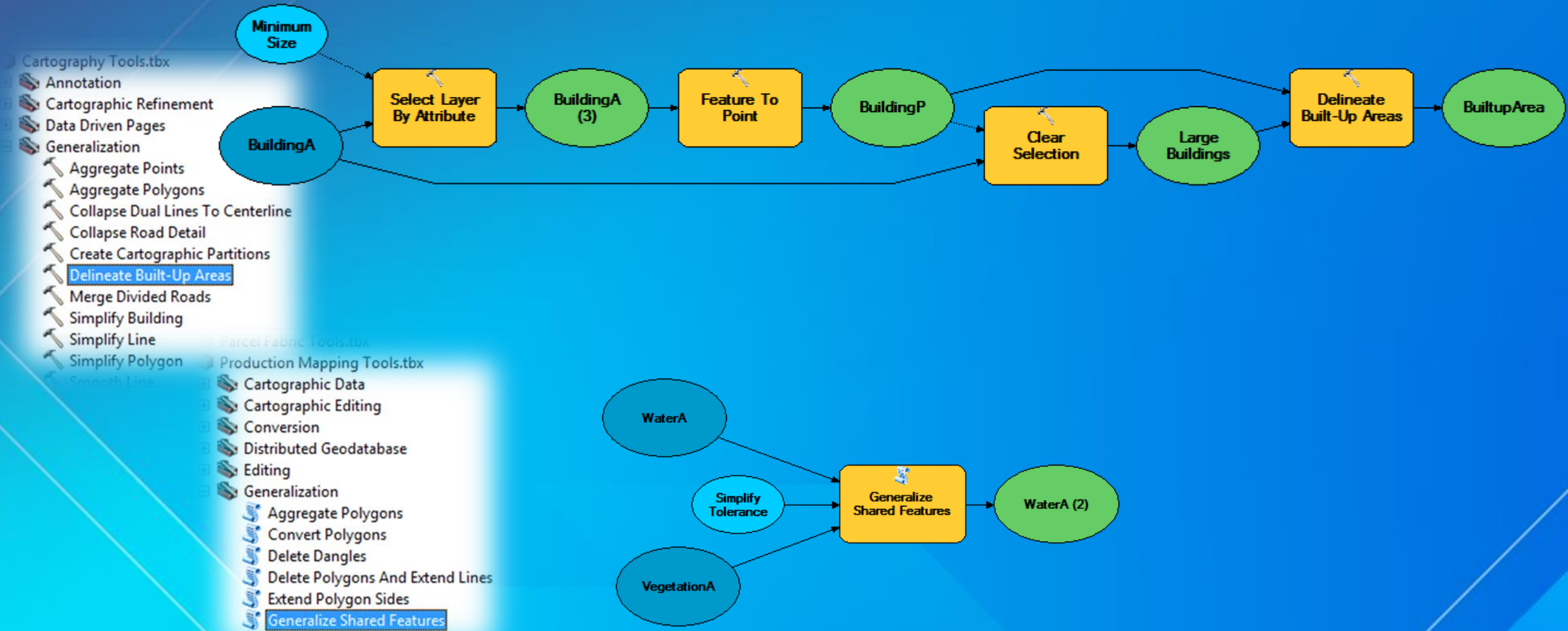


- **Features from multiple layers assessed simultaneously**
  - **Maintain pattern, density, and spatial relationships**

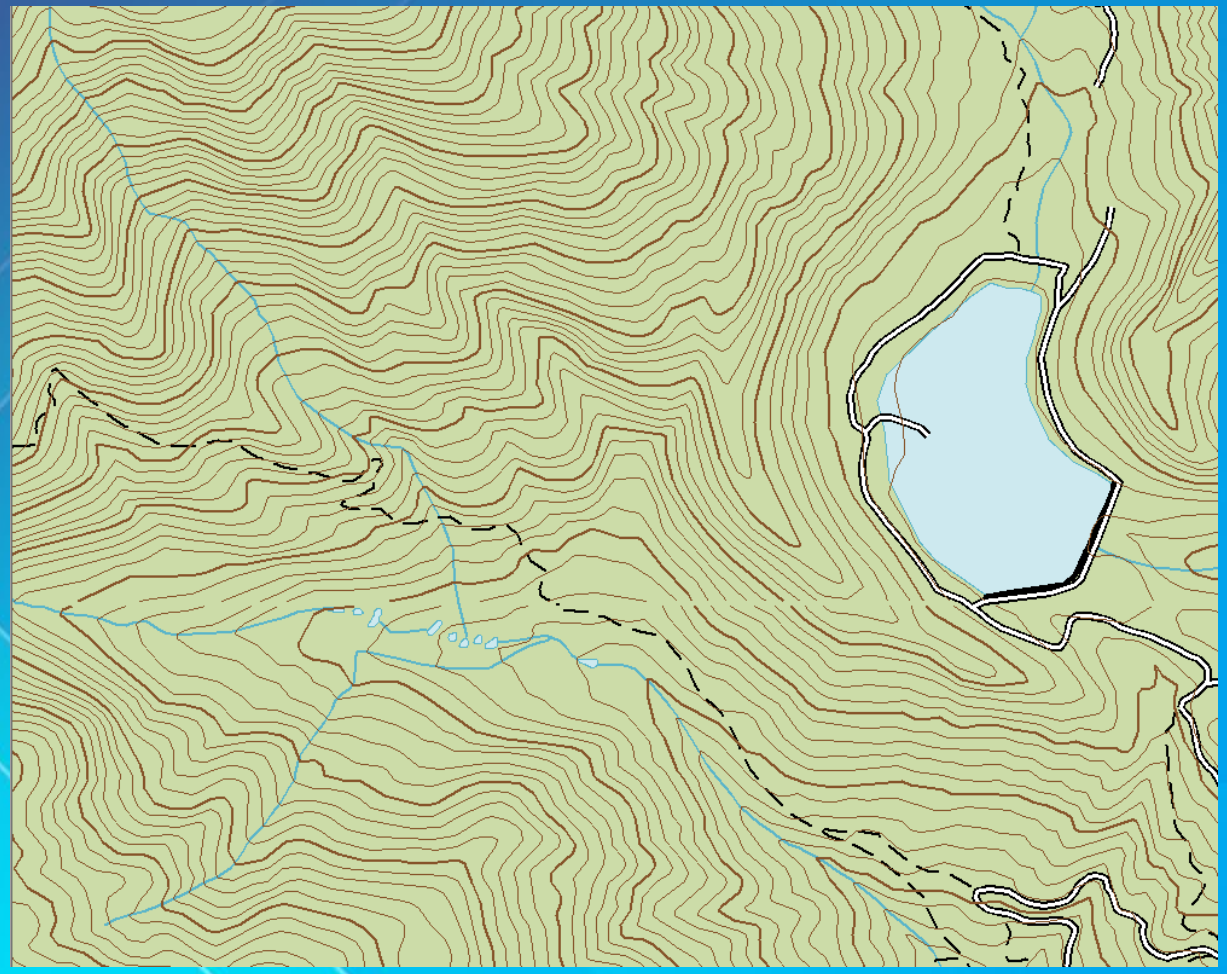
Find the tool and add to a model

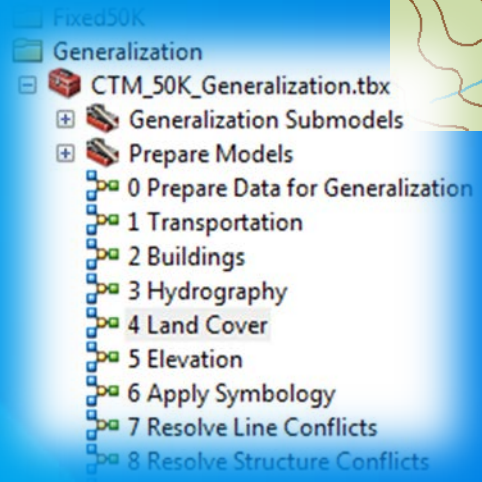


Find the tool and add to a model



Run the models

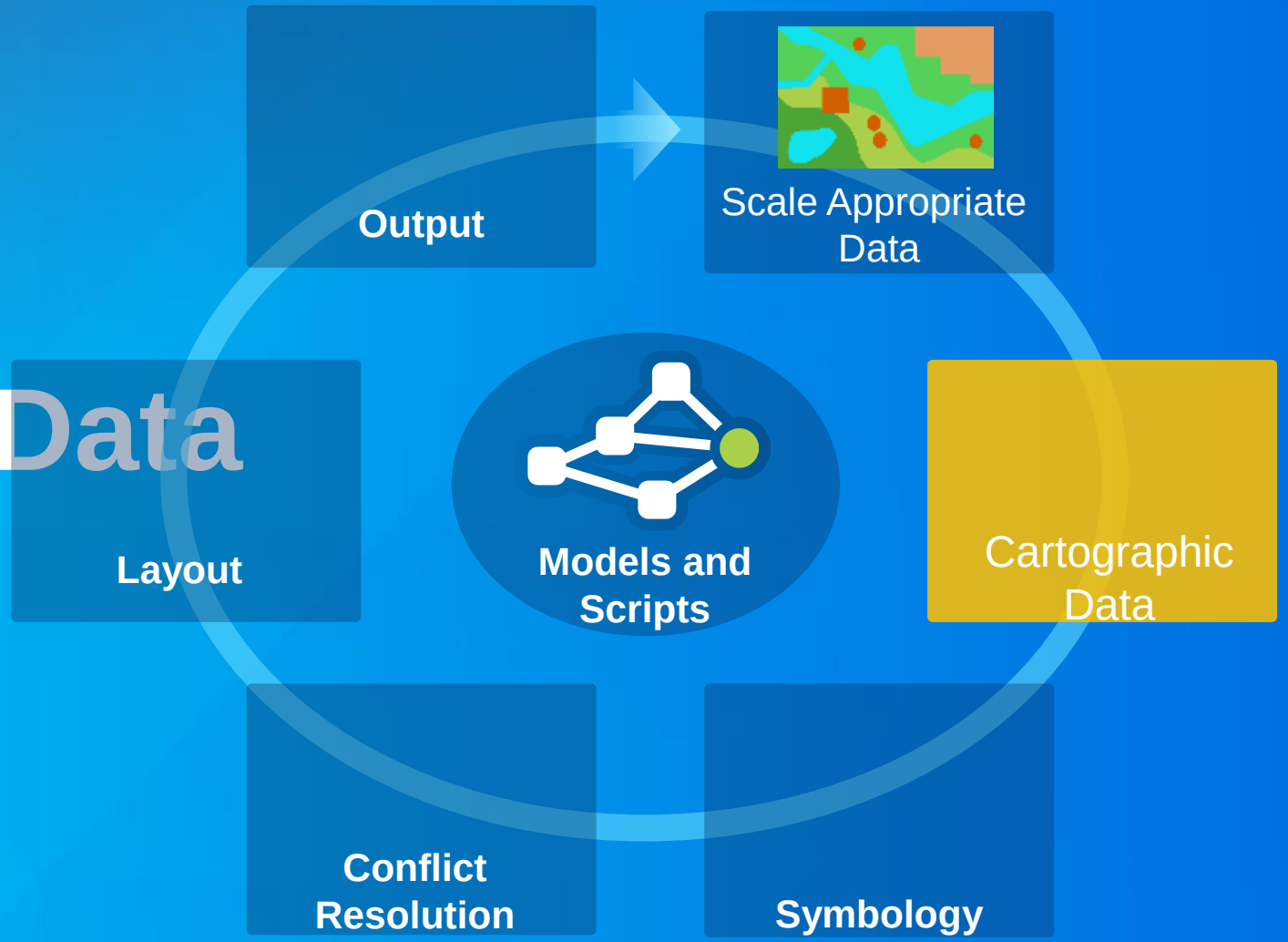




Demo

Creating Scale Appropriate Data

Cartographic Data



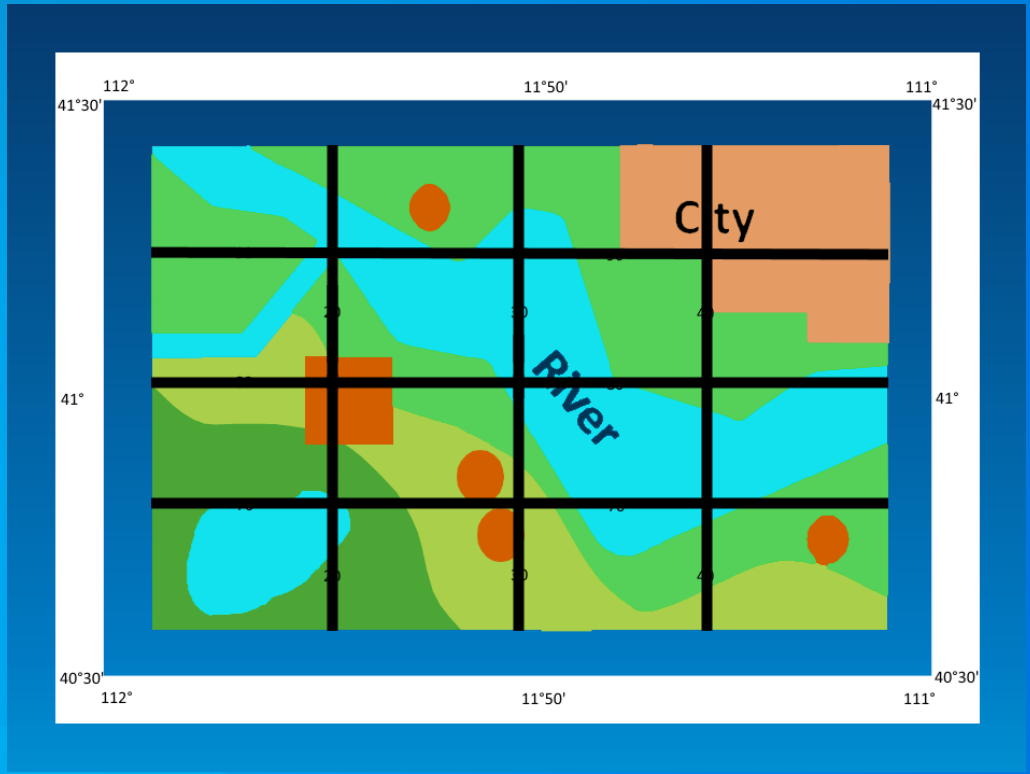
Cartographic Data

Data and text used to enrich cartographic products

Visualization of abstract concepts

Cartographic Data

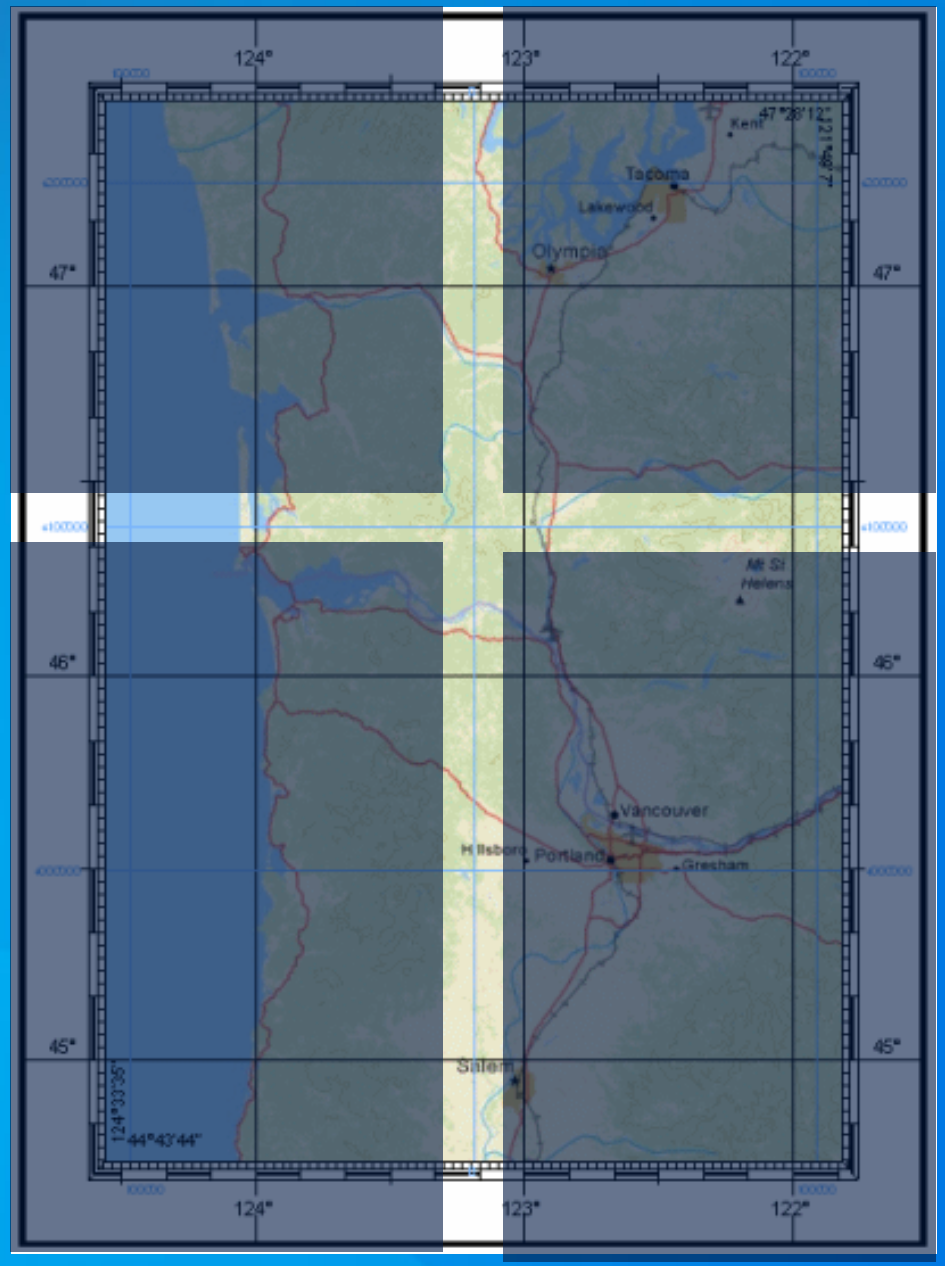
Data and text used to enrich cartographic products



Visualization of abstract concepts

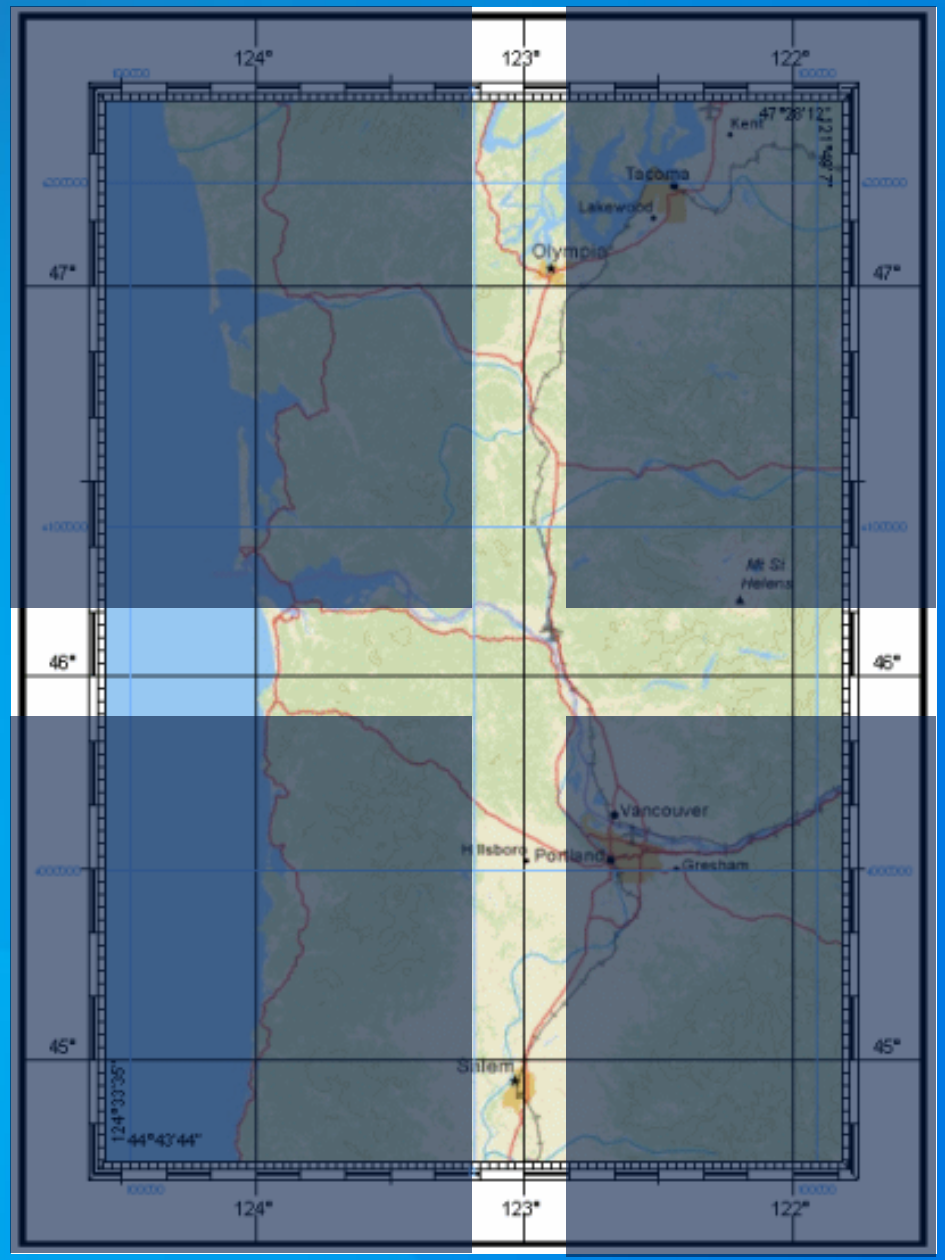
Grids and Graticules

- Grids



Grids and Graticules

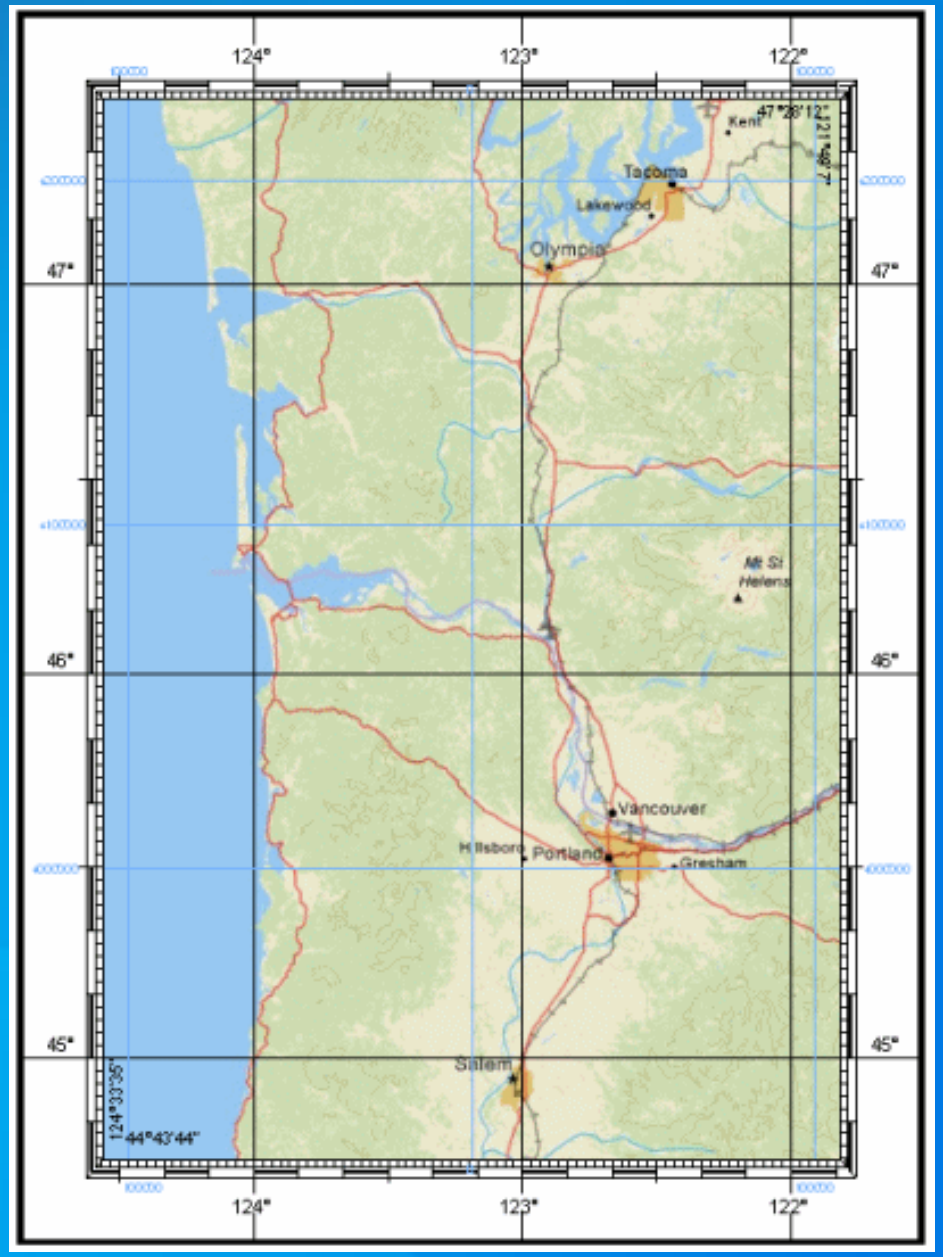
- Grids
- Graticules



Grids and Graticules

About

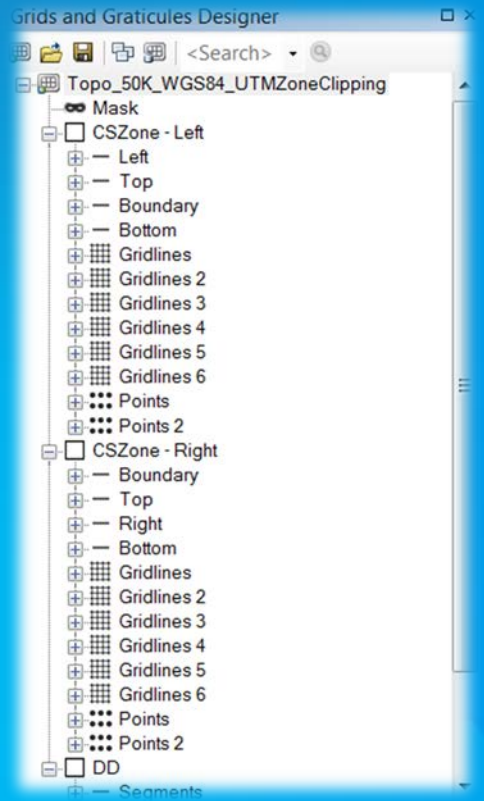
- Grids
- Graticules
- Features

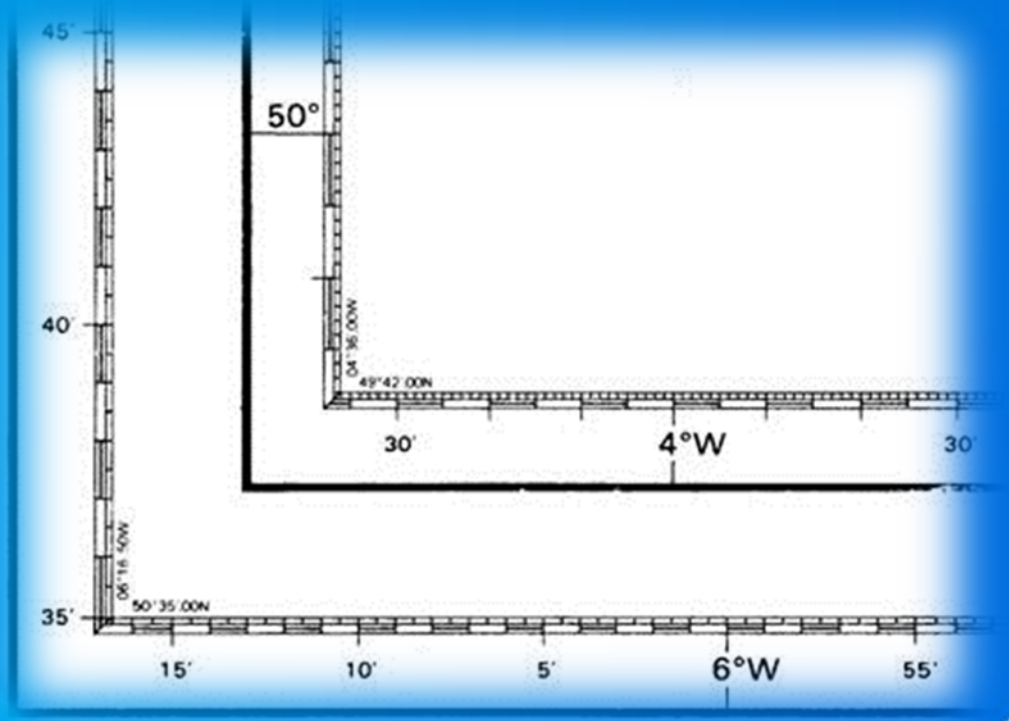


Grids and Graticules

Designing

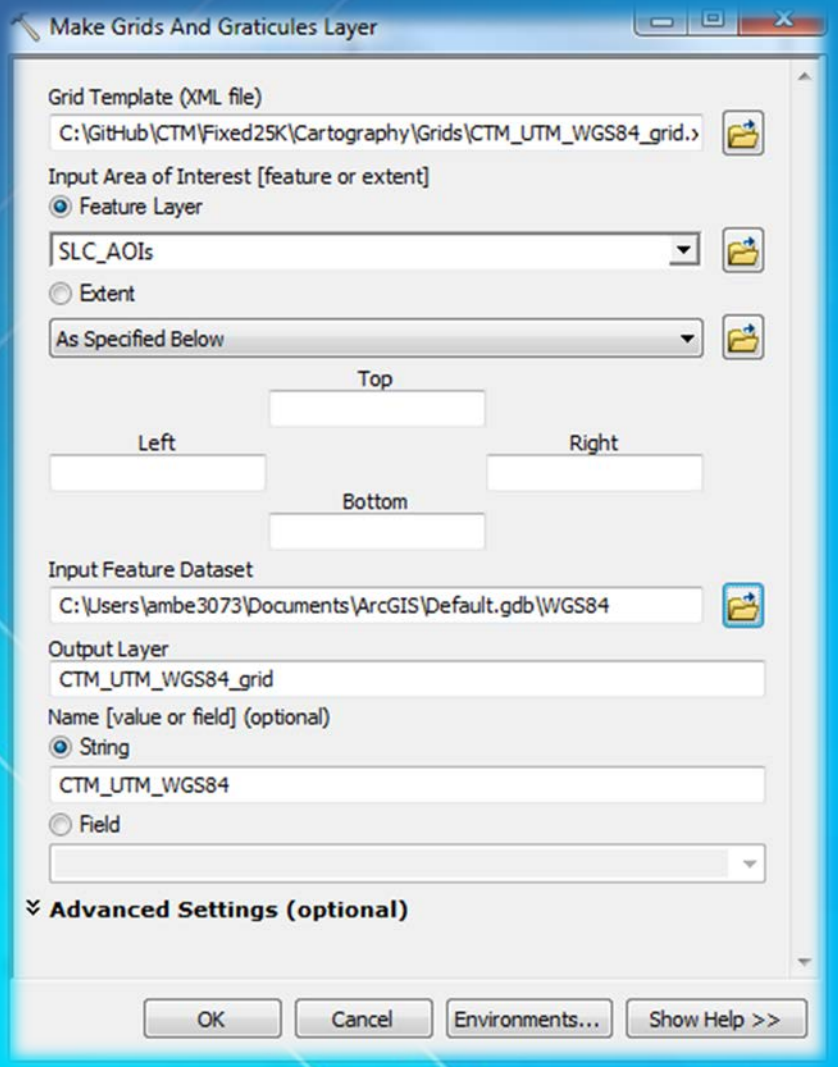
- Scale
- Coordinate system
- Grid line spacing
- Annotation placement
- Rotation
- Line weight, font

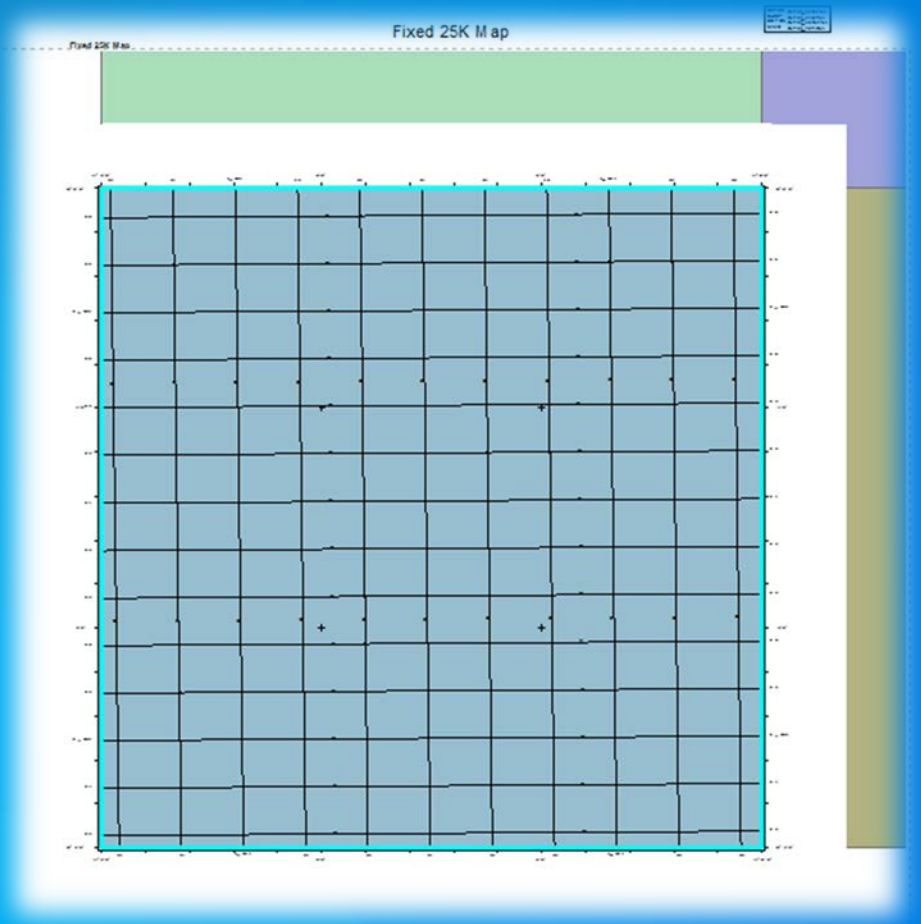




Grids and Graticules

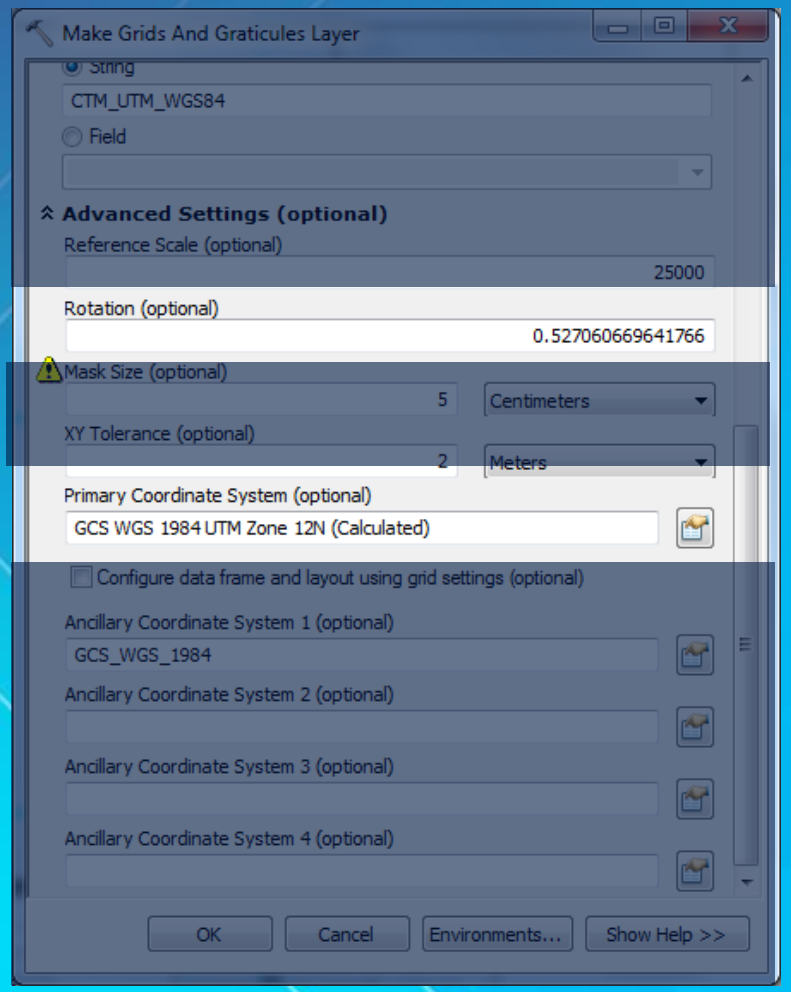
Applying

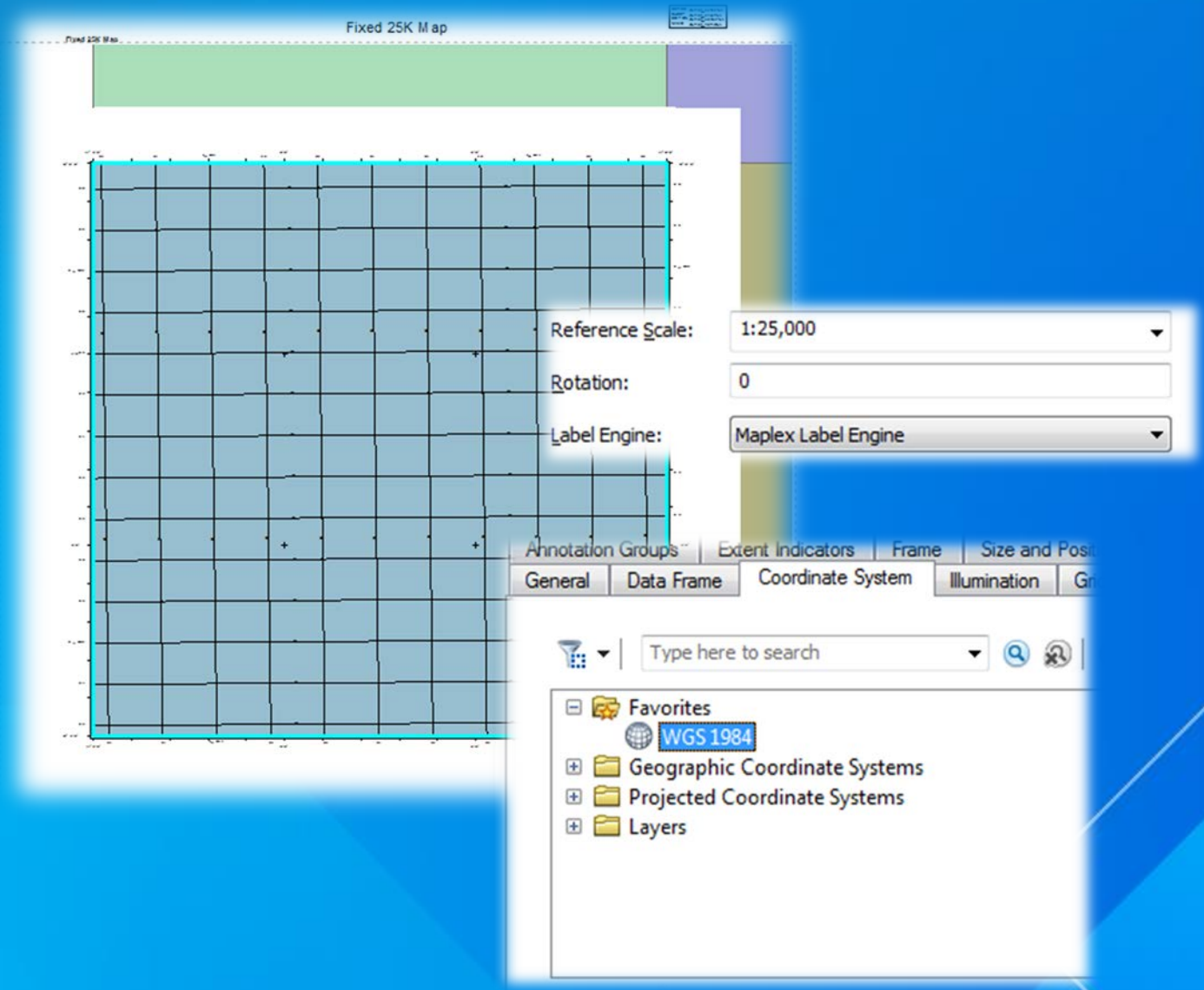




Grids and Graticules

Applying

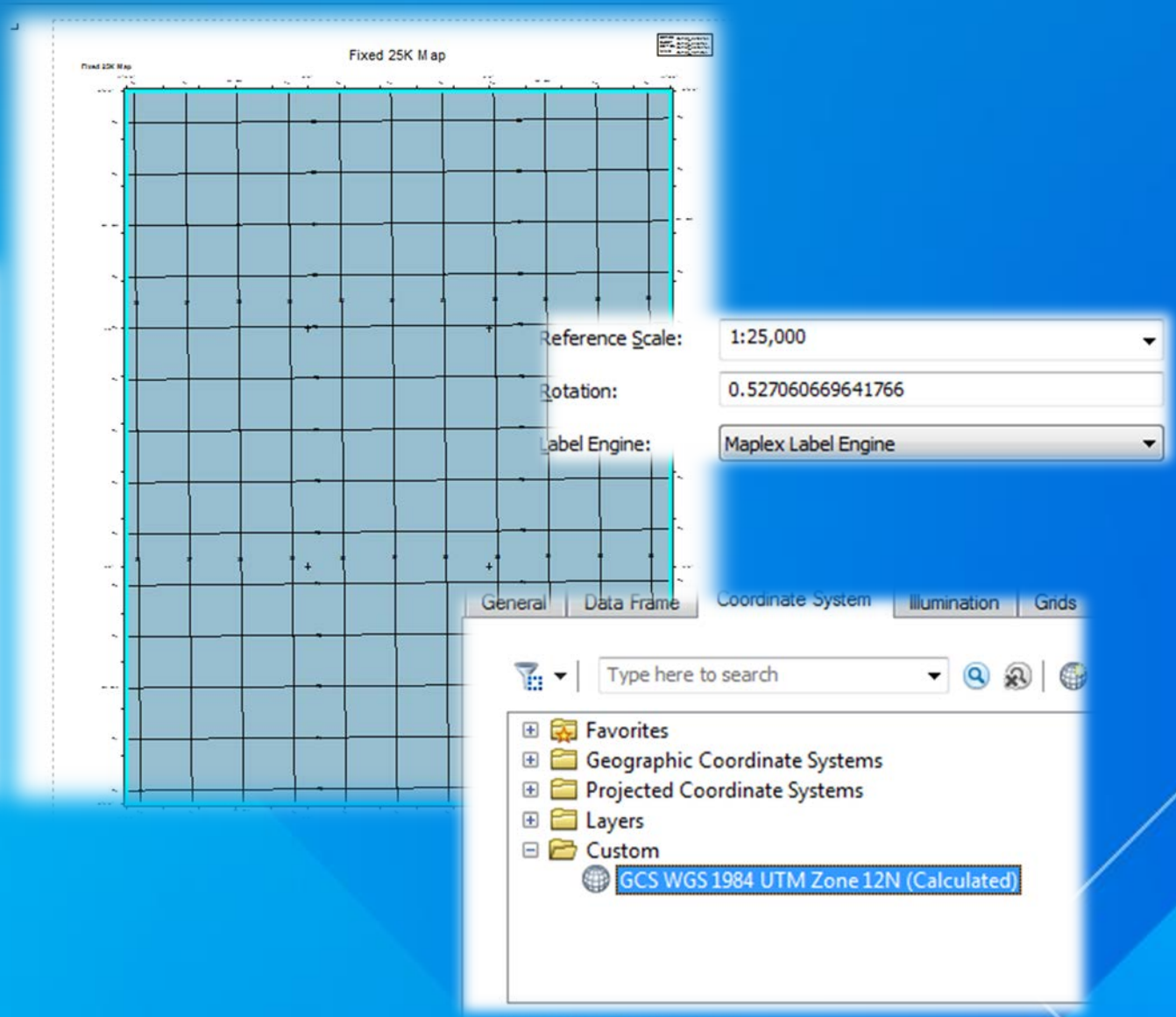




Grids and Graticules

Applying

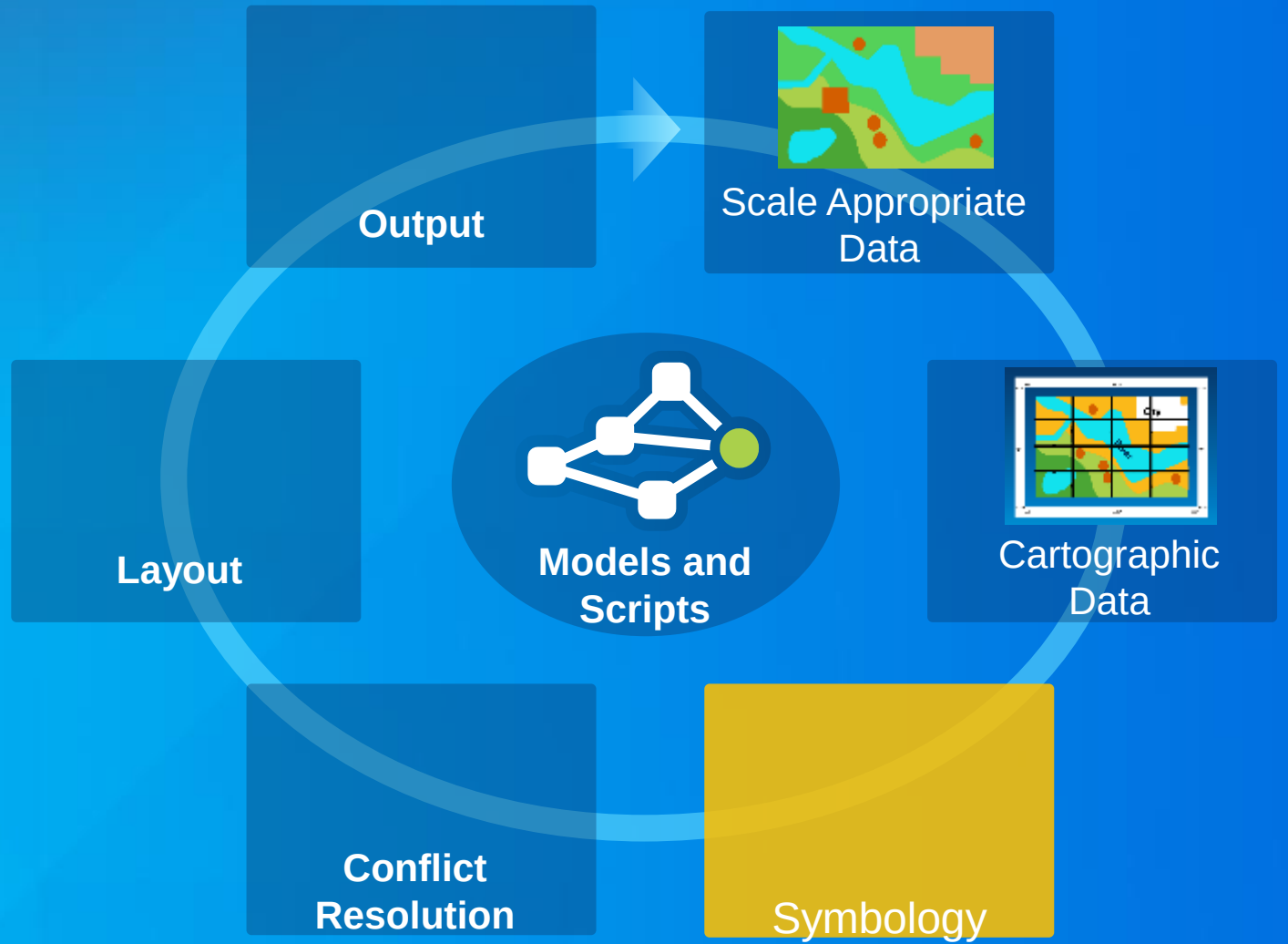
```
#Define grid object
grid = arcpyproduction.mapping.Grid(grid_xml)

#Uses the appropriate XML for to create the grid
arcpy.AddMessage("Creating the Grid...")
output_layer = map_name + '_' + grid.type
grid_result = arcpy.MakeGridsAndGraticulesLayer_cartography(grid_xml,
                aoi, gfds, output_layer, map_name)
grid_layer = grid_result.getOutput(0)

# Add the grid layer to the top of the map
arcpy.mapping.AddLayer(data_frame, grid_layer, "TOP")
arcpy.AddMessage("Grid Layer added to map...")

# Updates the data frame properties base on the grid
final_mxd.activeView = 'PAGE_LAYOUT'
grid.updateDataFrameProperties(data_frame, aoi)
```



Demo

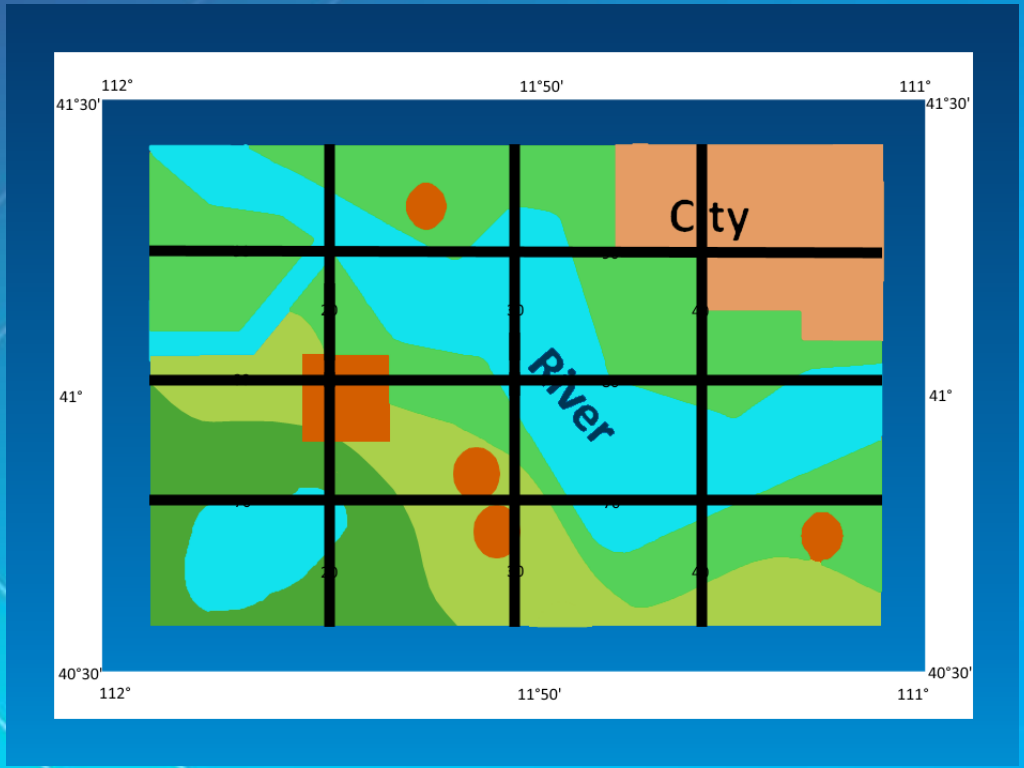
Cartographic Data

Symbology


Output
Scale Appropriate Data
Layout
Models and Scripts
Cartographic Data
Conflict Resolution
Symbology

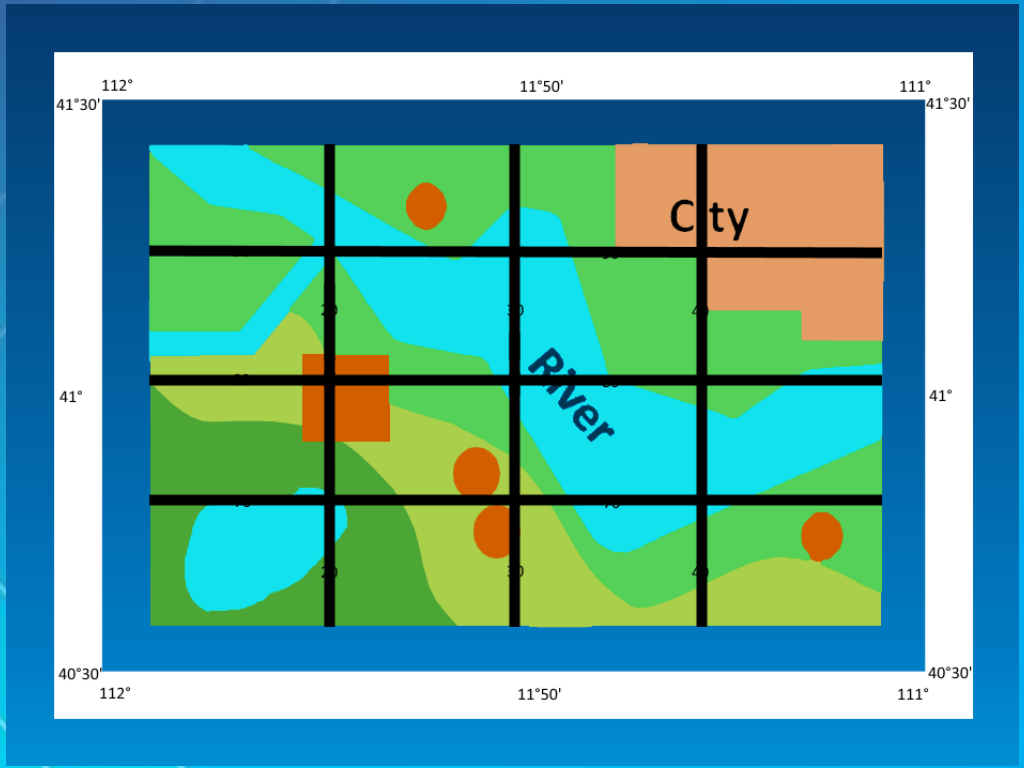
Symbology

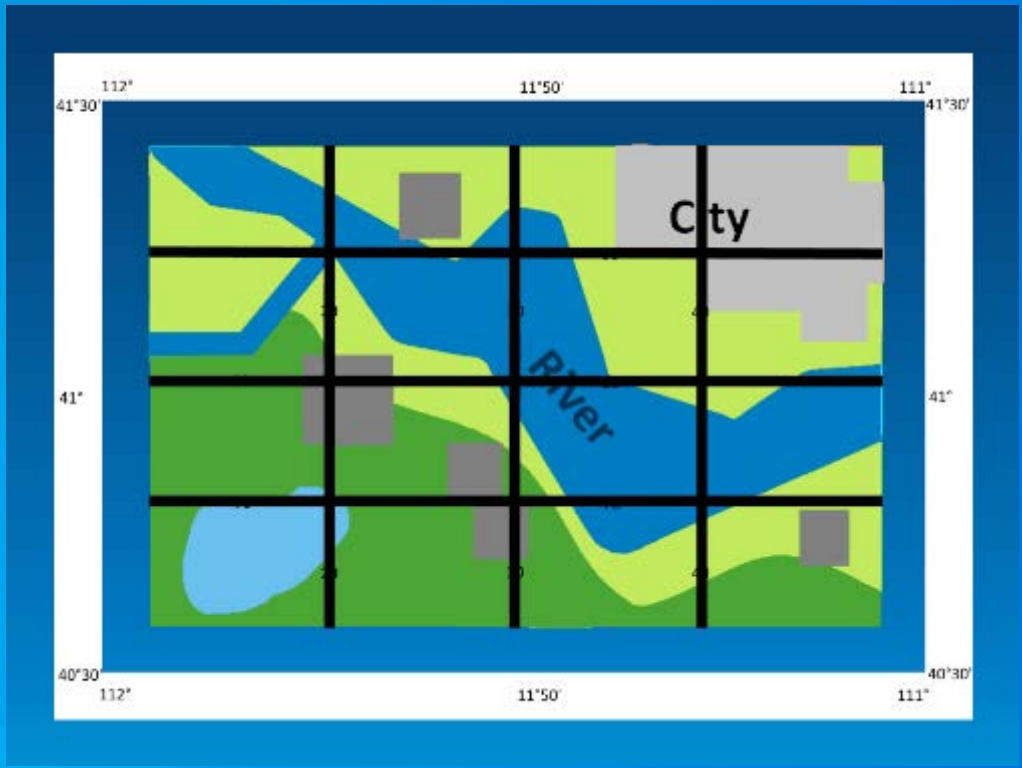
Intuitively displaying information



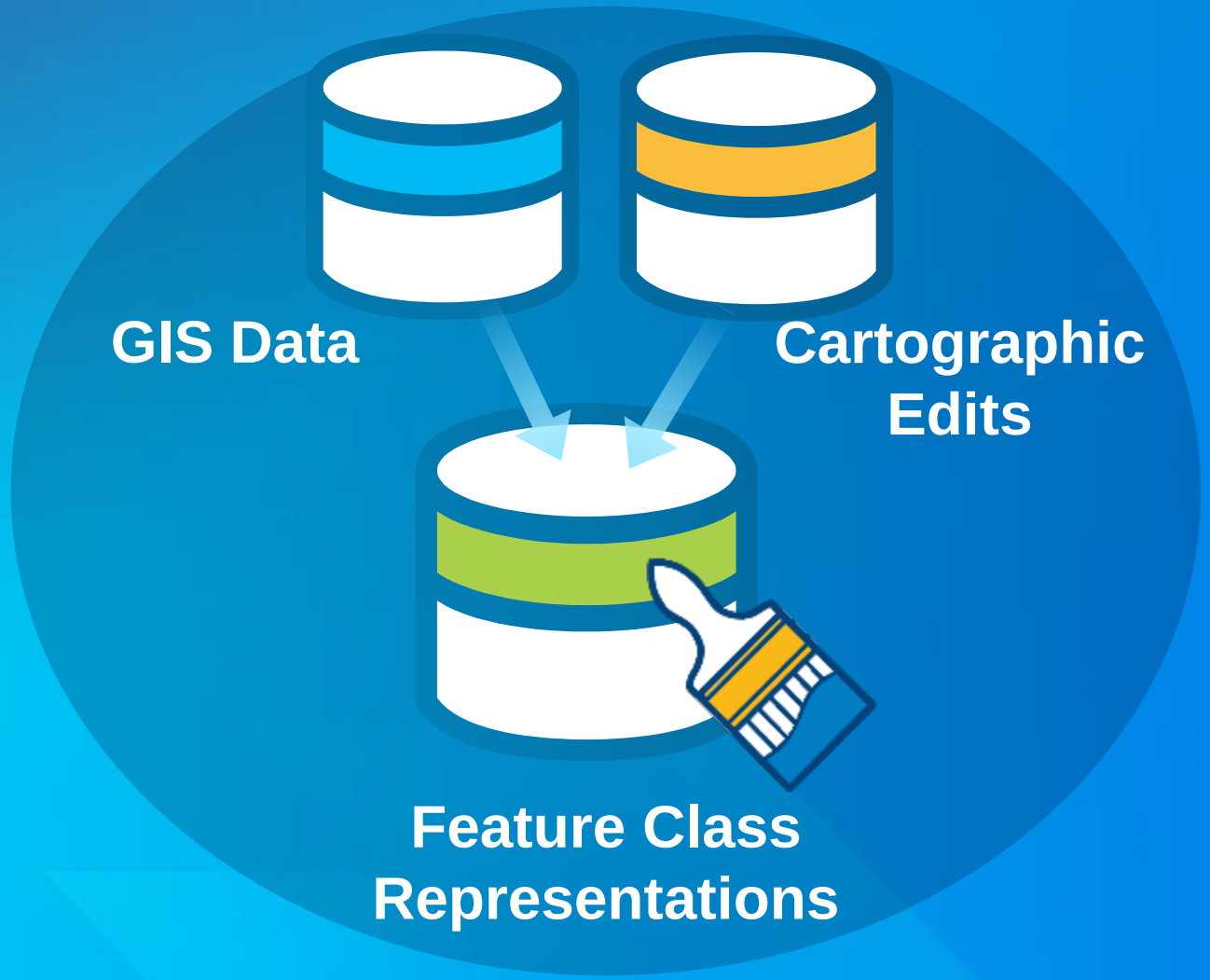
Symbology

Intuitively displaying information





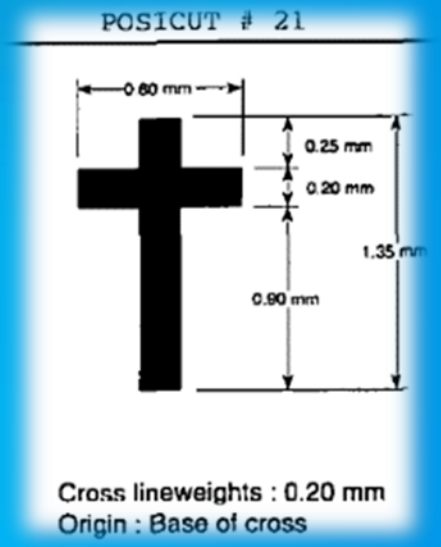
Feature Class Representations



Visual Specifications

Defining

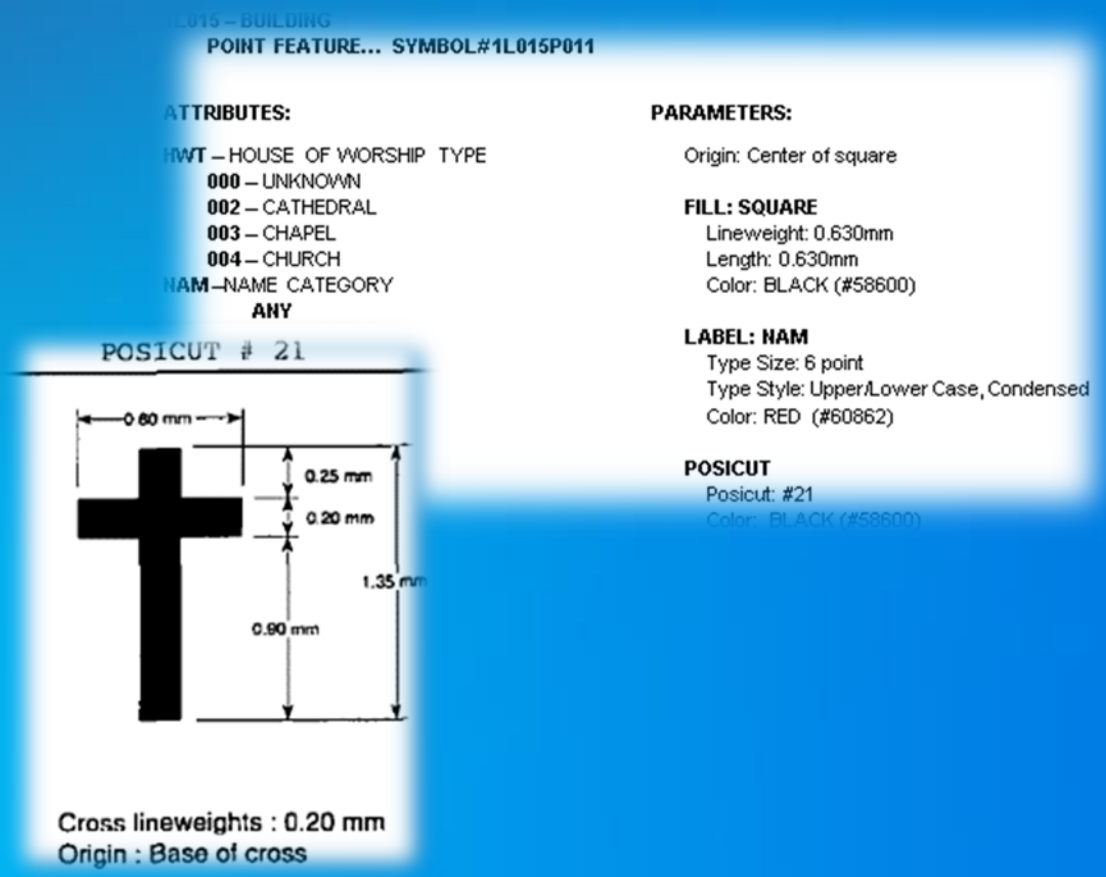
- **Create Symbology**



Visual Specifications

Defining

- **Create Symbology**
- **Know Your Rules**



L015 – BUILDING
POINT FEATURE... SYMBOL#1L015P011

ATTRIBUTES:

HWT – HOUSE OF WORSHIP TYPE
000 – UNKNOWN
002 – CATHEDRAL
003 – CHAPEL
004 – CHURCH
NAM – NAME CATEGORY
ANY

PARAMETERS:

Origin: Center of square

FILL: SQUARE
Lineweight: 0.630mm
Length: 0.630mm
Color: BLACK (#58600)

LABEL: NAM
Type Size: 6 point
Type Style: Upper/Lower Case, Condensed
Color: RED (#60862)

POSICUT
Posicut: #21
Color: BLACK (#58600)

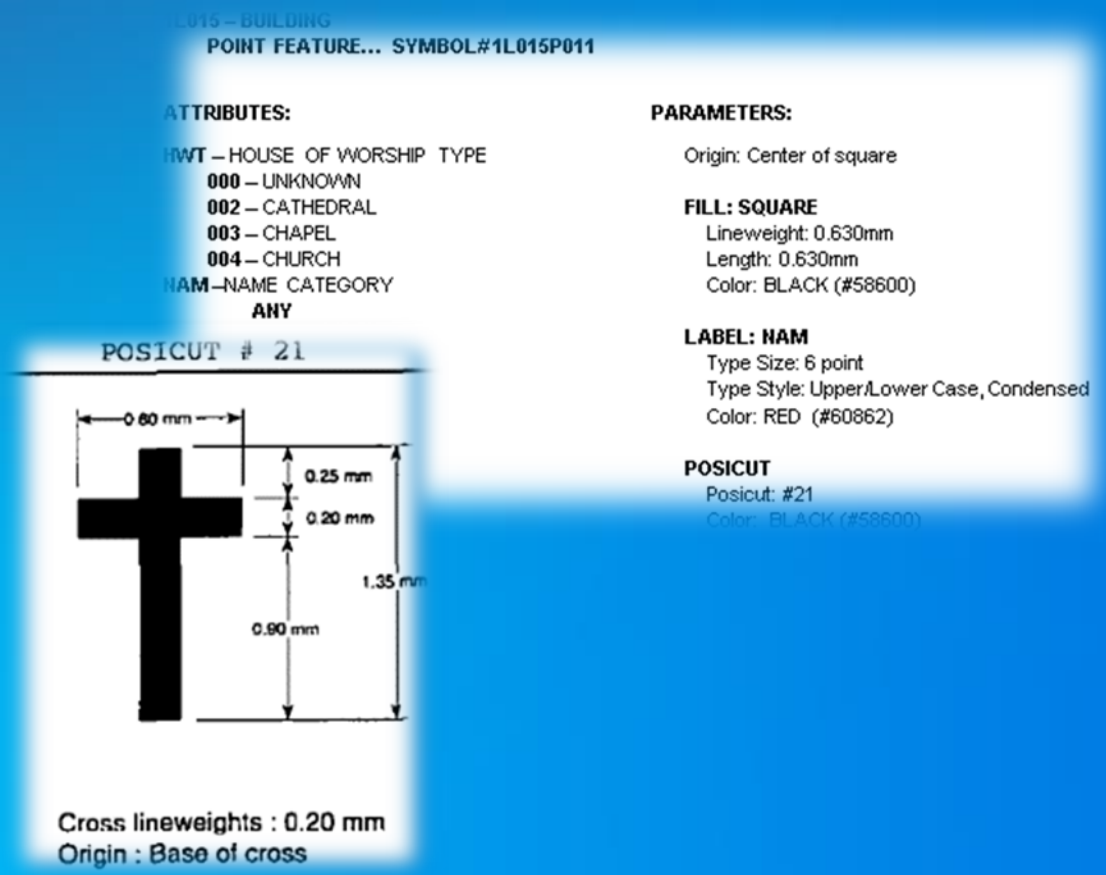
POSICUT # 21

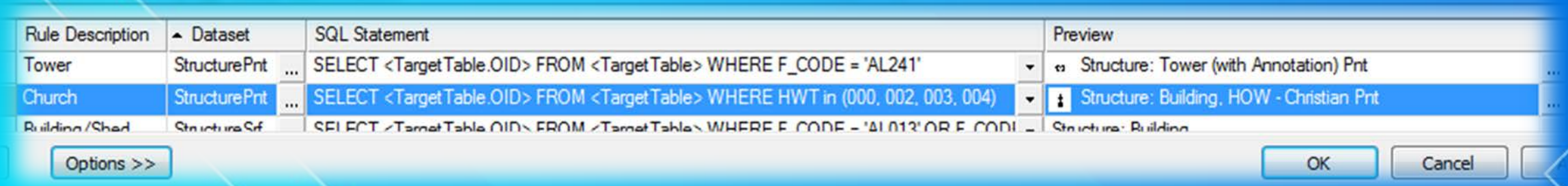
Cross lineweights : 0.20 mm
Origin : Base of cross

Visual Specifications

Defining

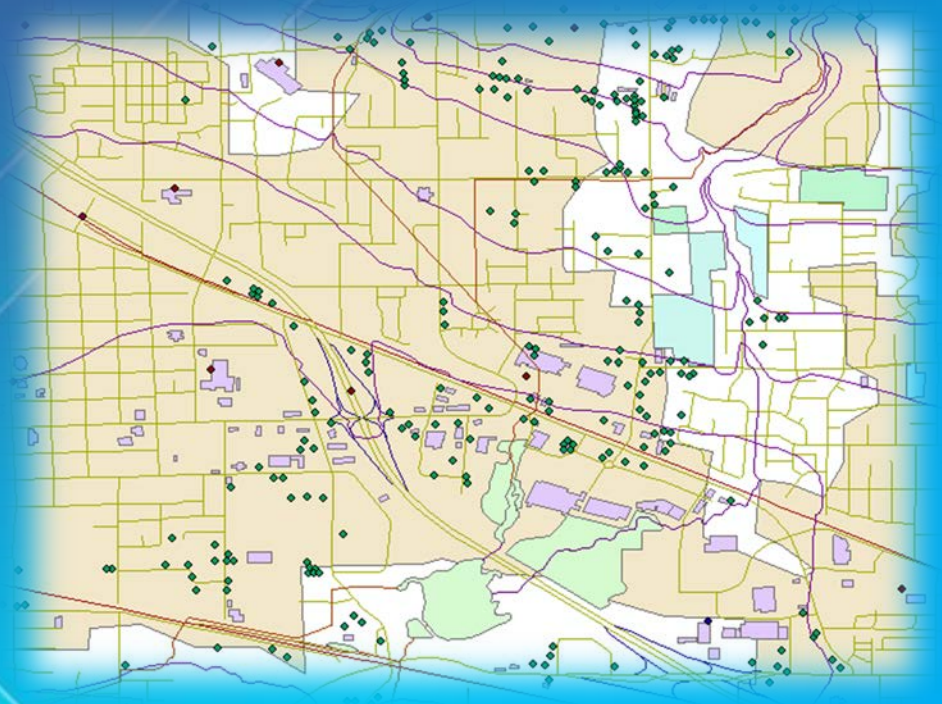
- Create Symbology
- Know Your Rules
- Define Specifications

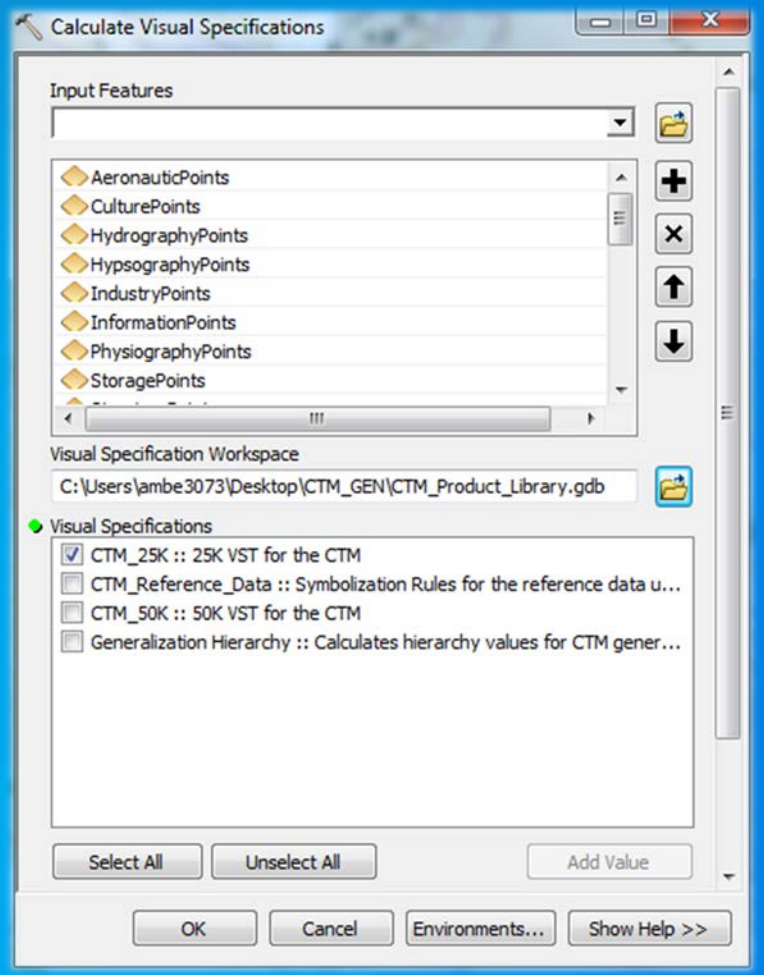




Visual Specifications

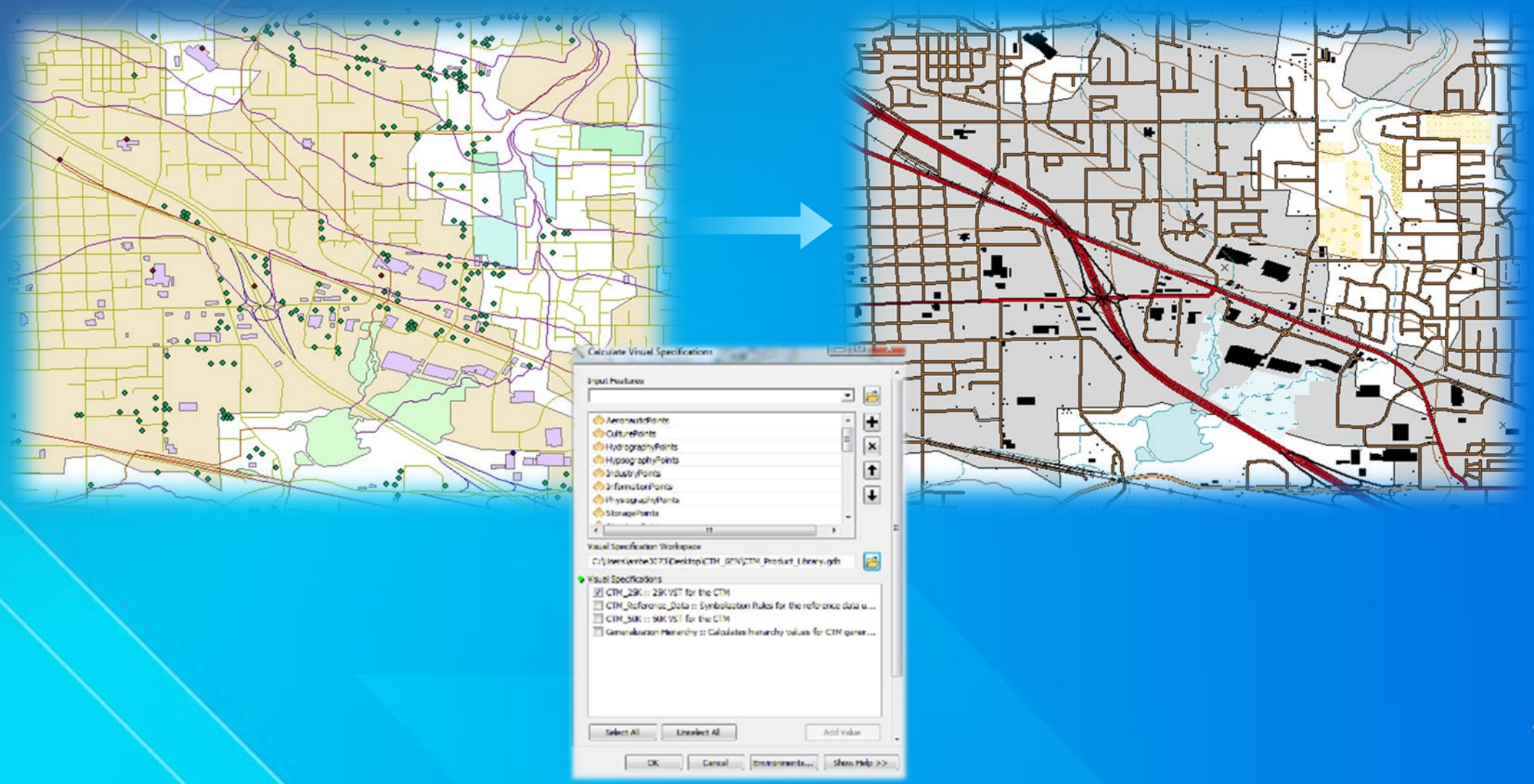
Applying



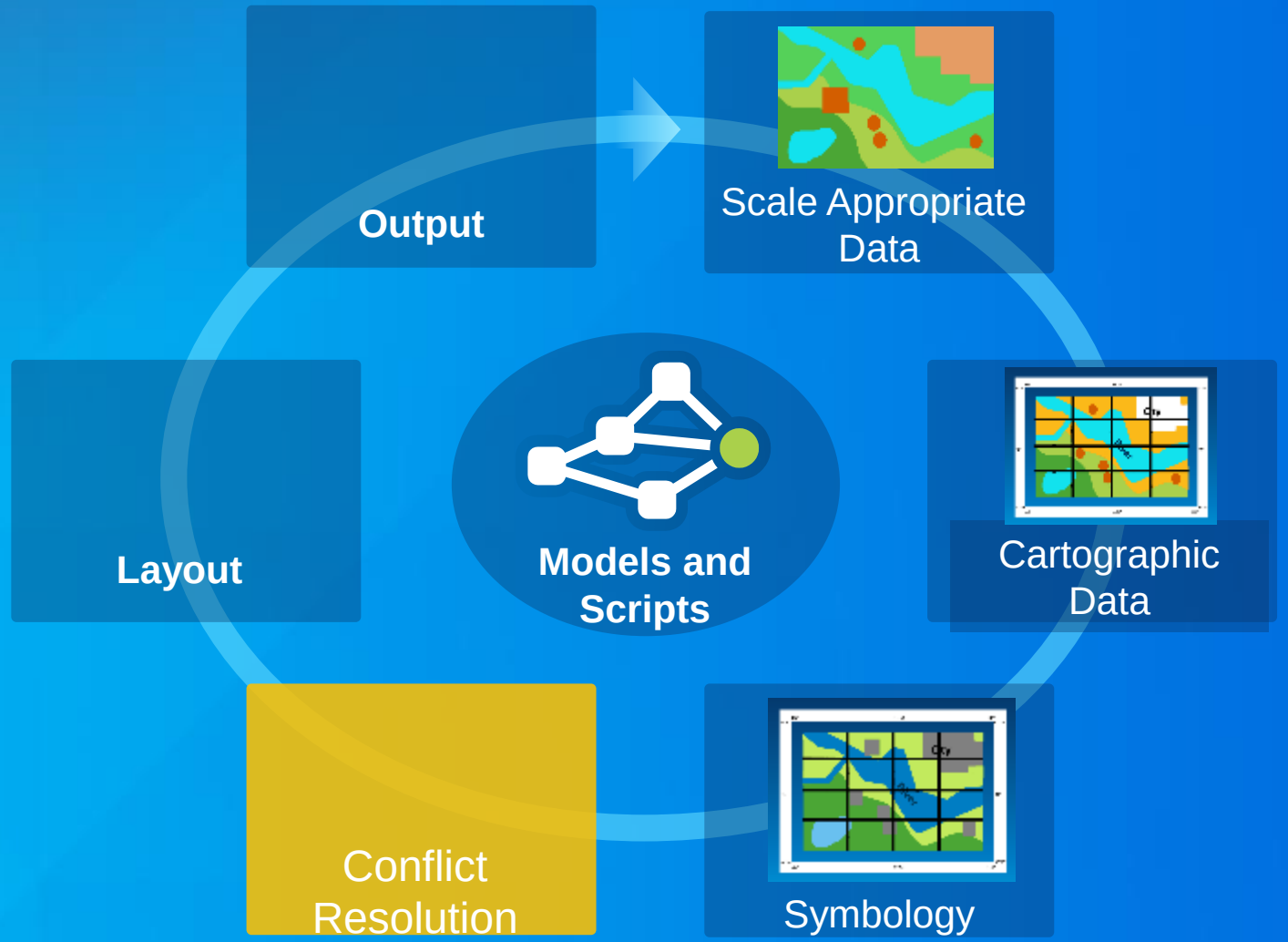


Visual Specifications

Applying

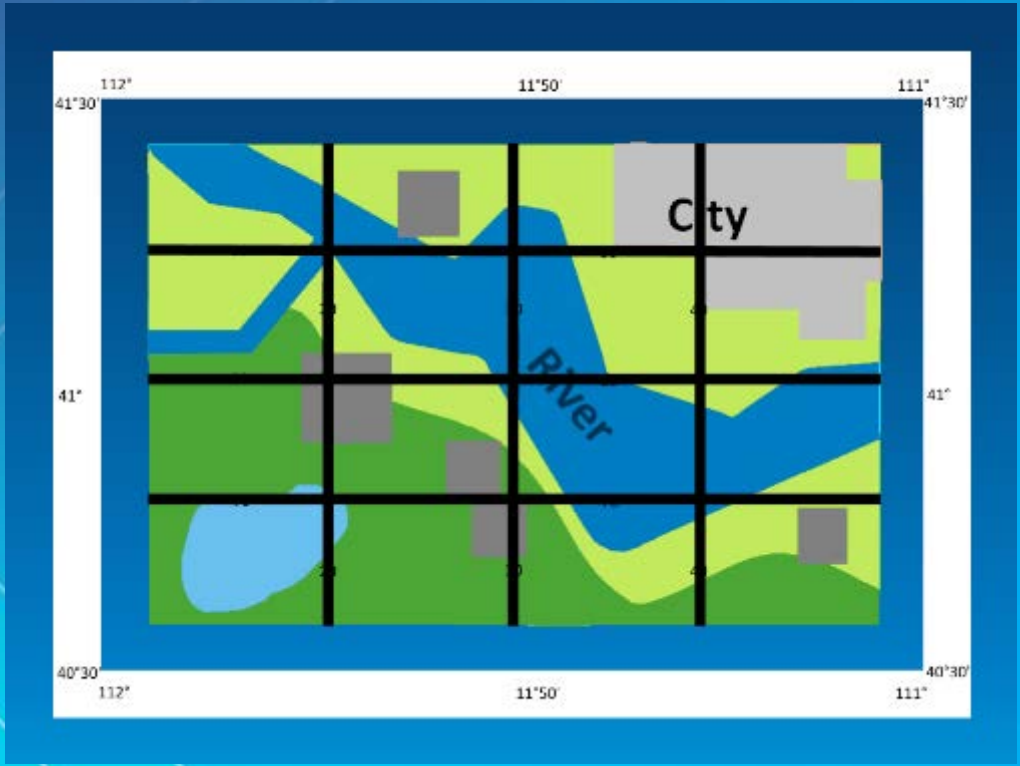


Cartographic Edits



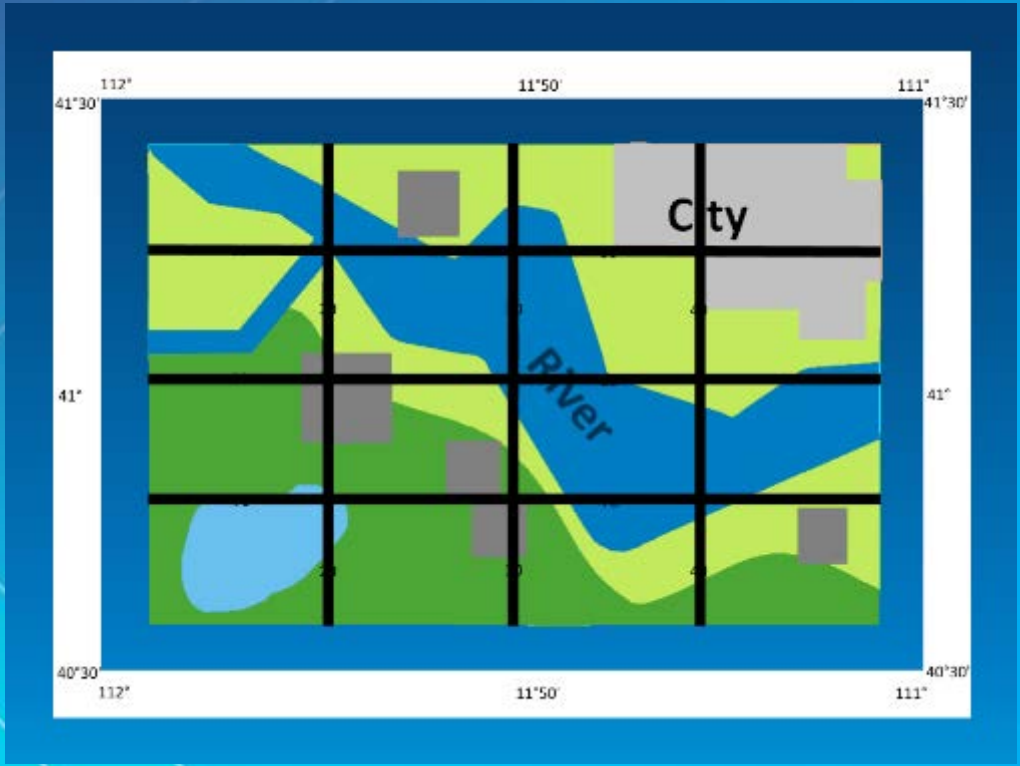
Cartographic Edits

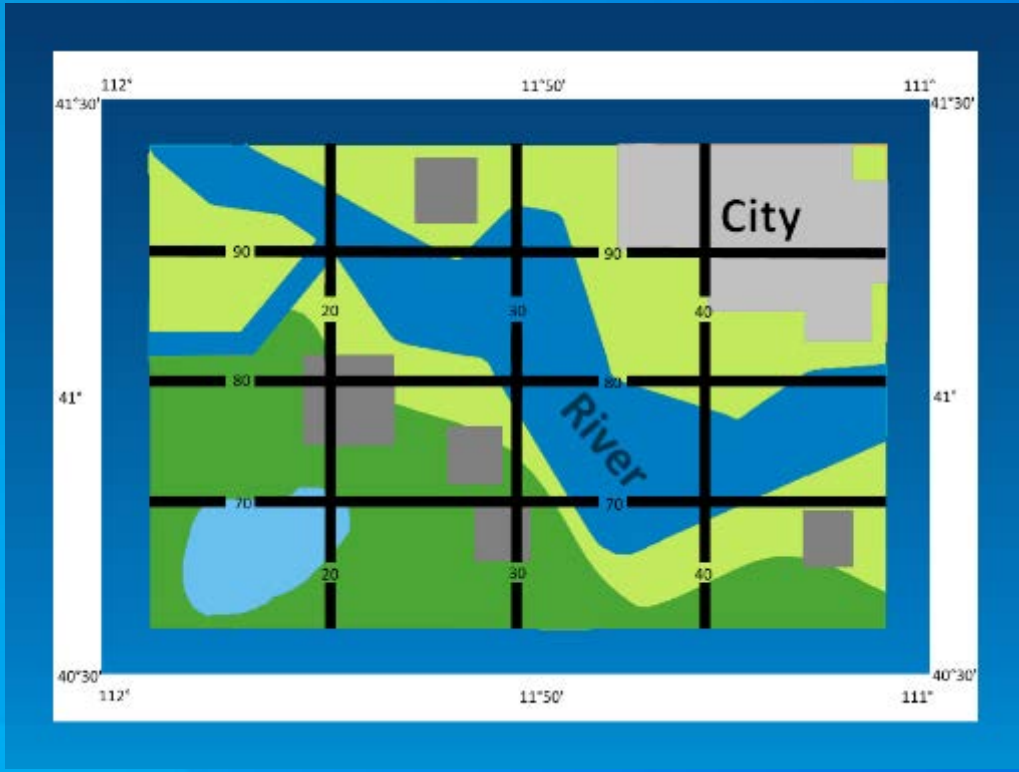
Improving map clarity



Cartographic Edits

Improving map clarity



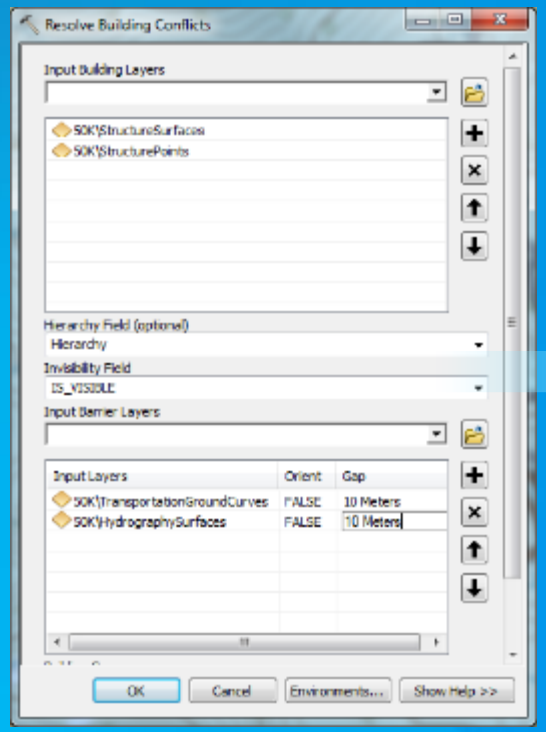


Conflict Resolution

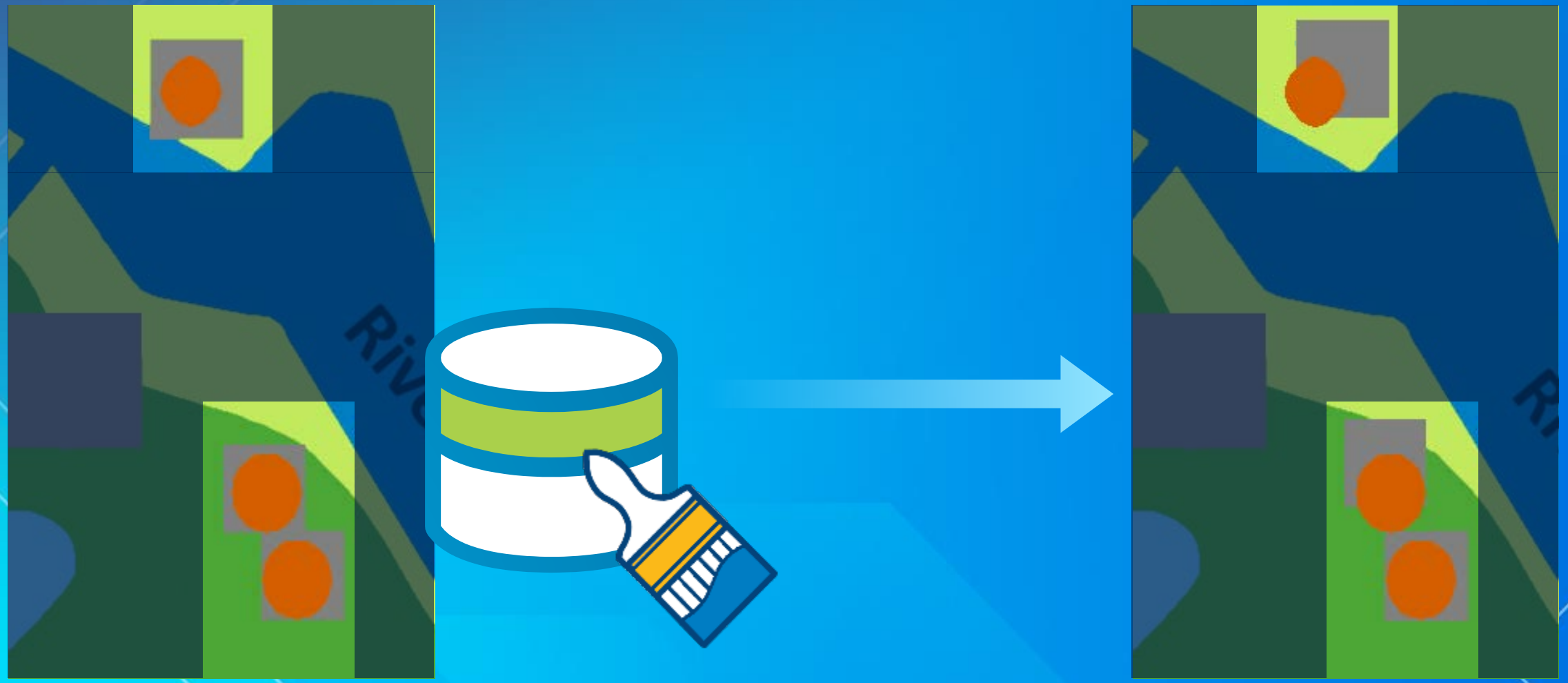
Managing the extent and placement of symbolized features on maps

Conflict Resolution

Managing the extent and placement of symbolized features on maps

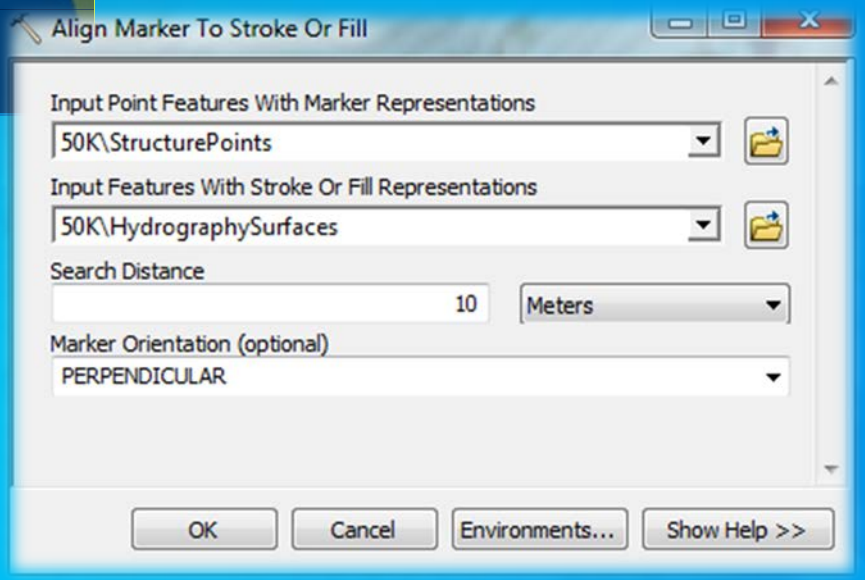


Feature Class Representations

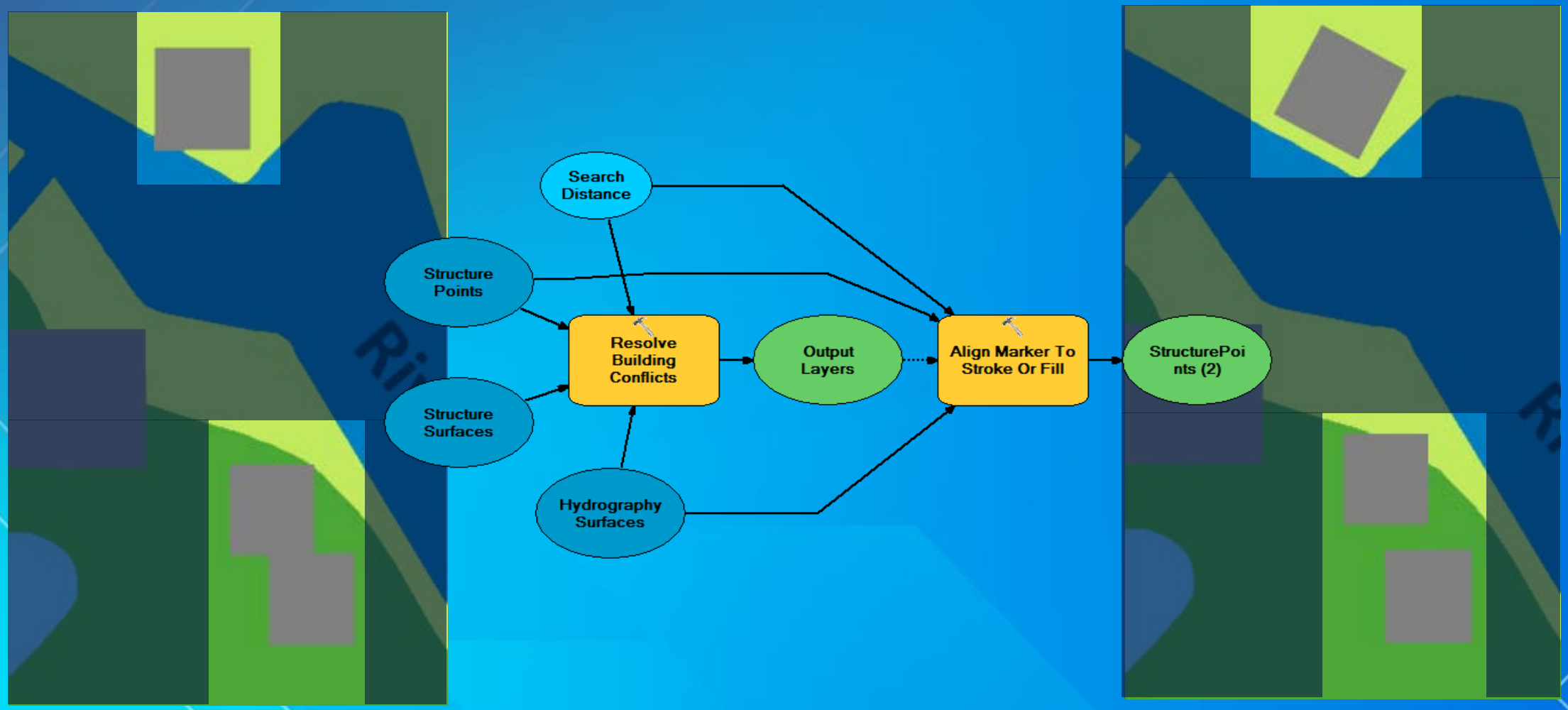


Cartographic Refinement

Adjusting the alignment and arrangement of symbols

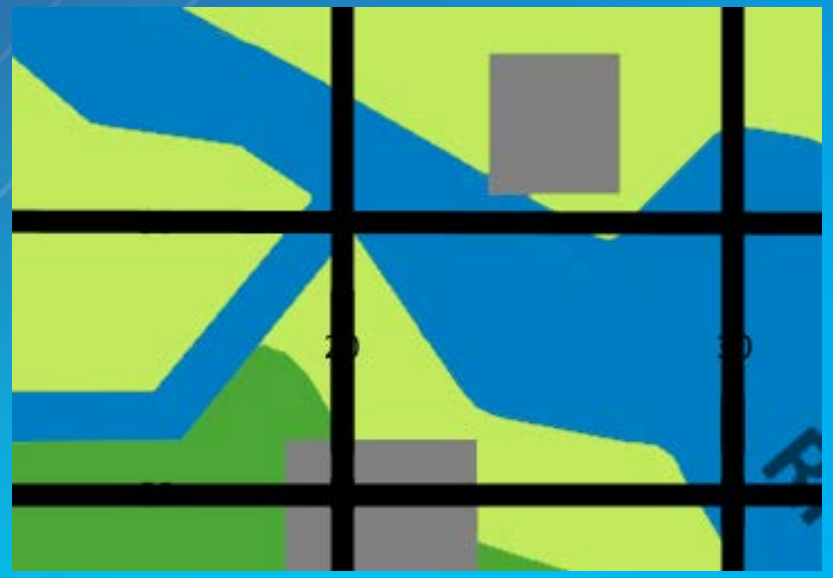


Automating Conflict Resolution



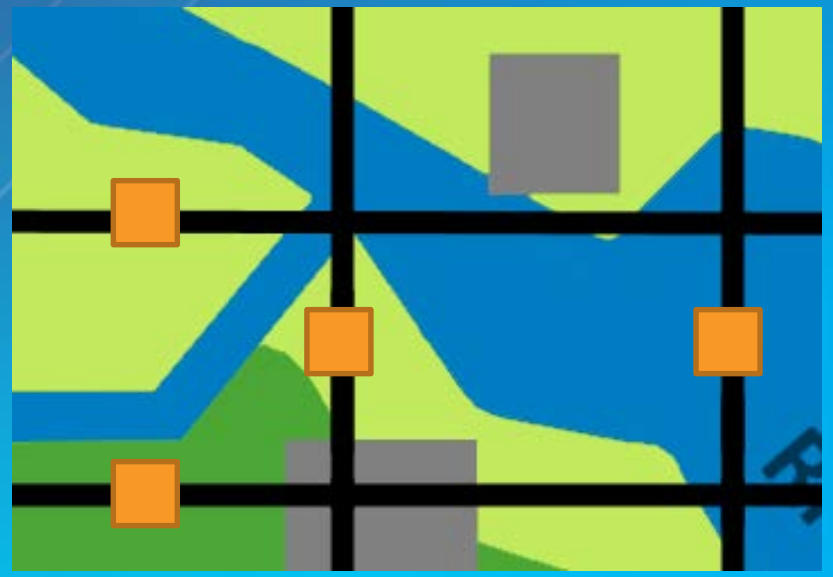
Masking

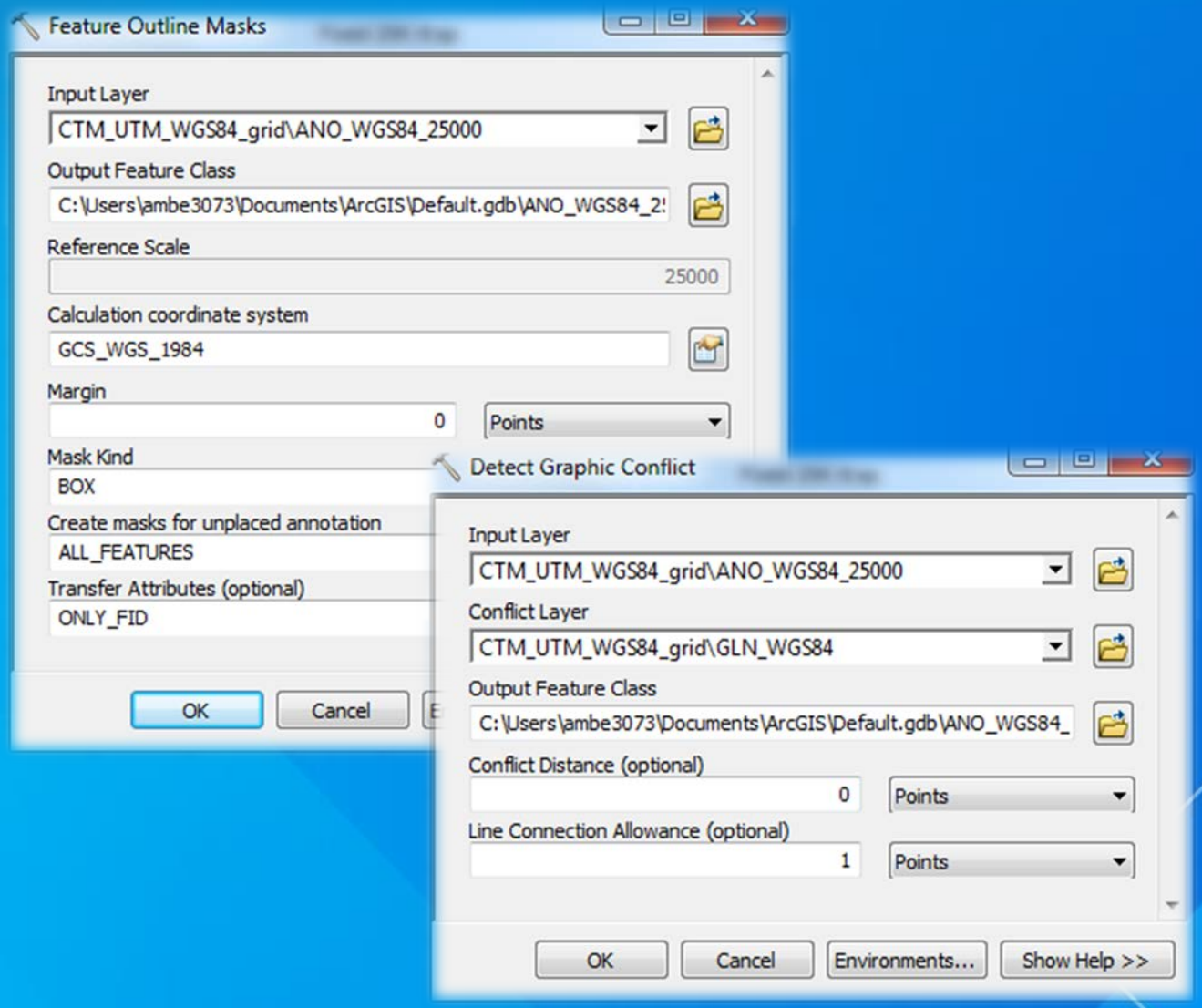
Enhancing cartographic display by obscuring conflicting symbols



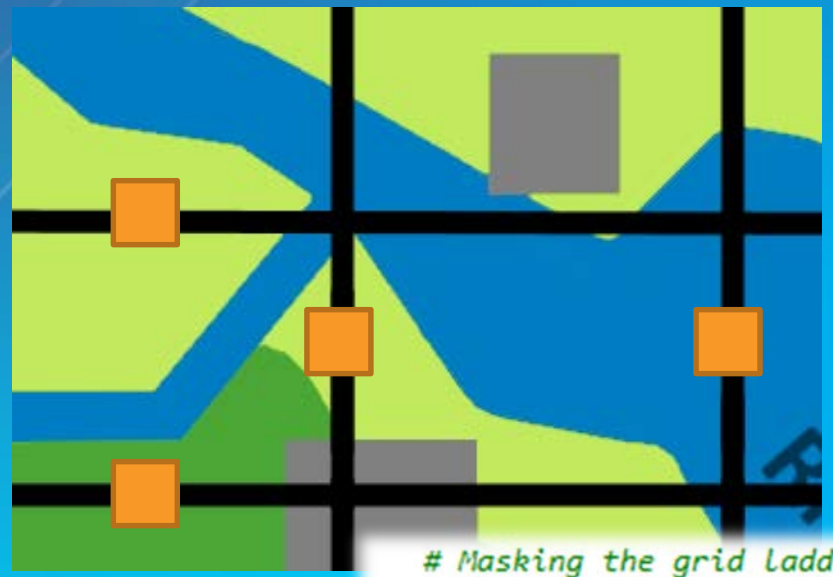
Masking

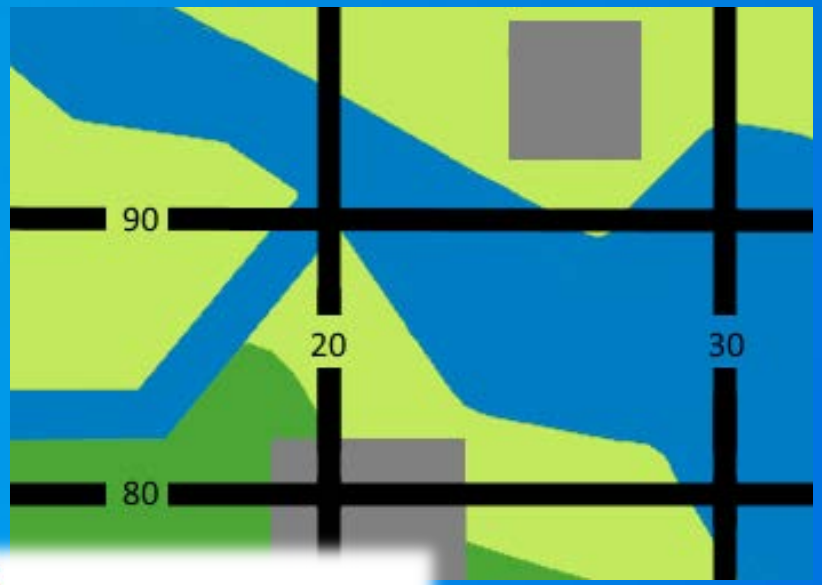
Enhancing cartographic display by obscuring conflicting symbols





Masking

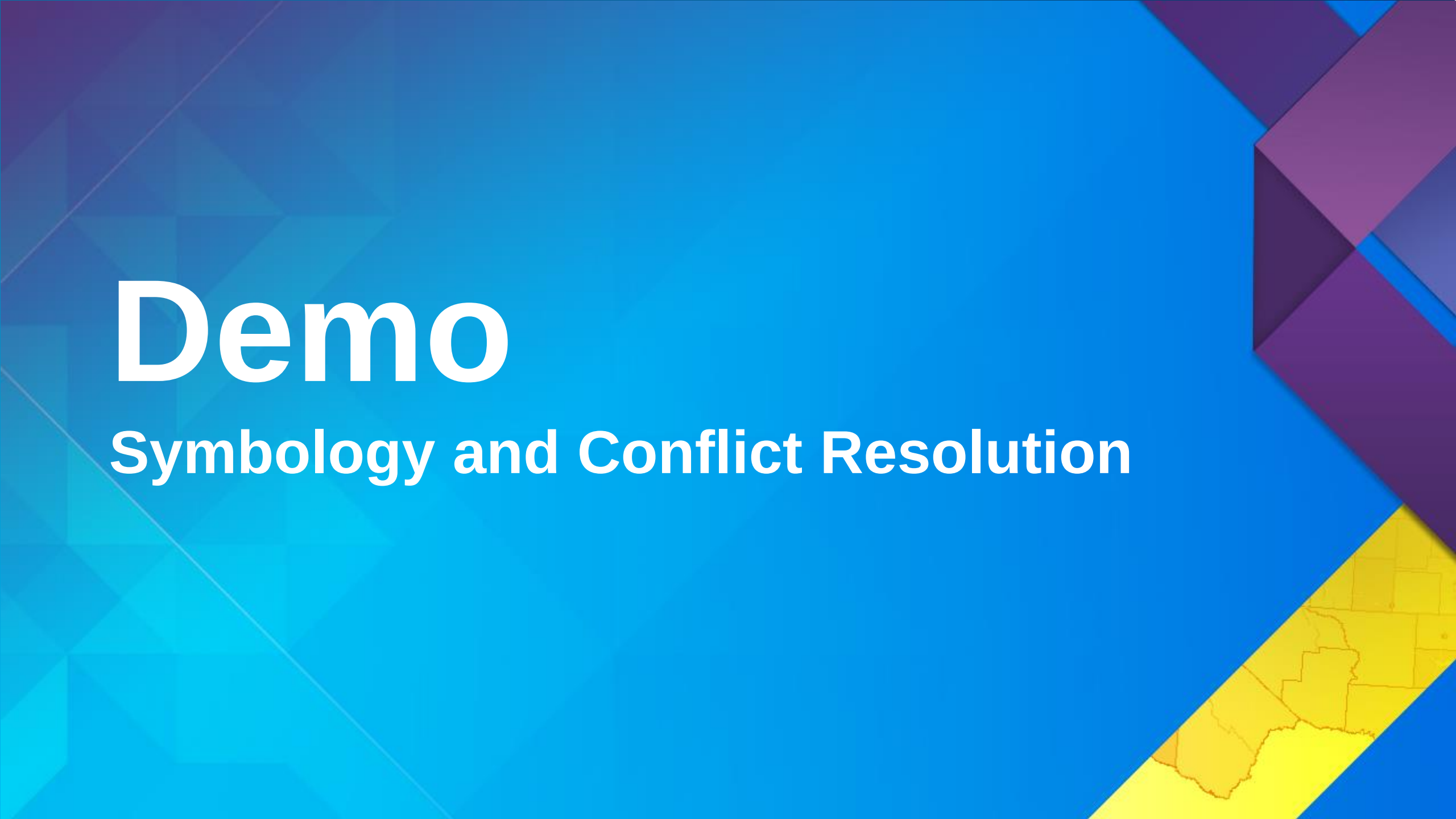




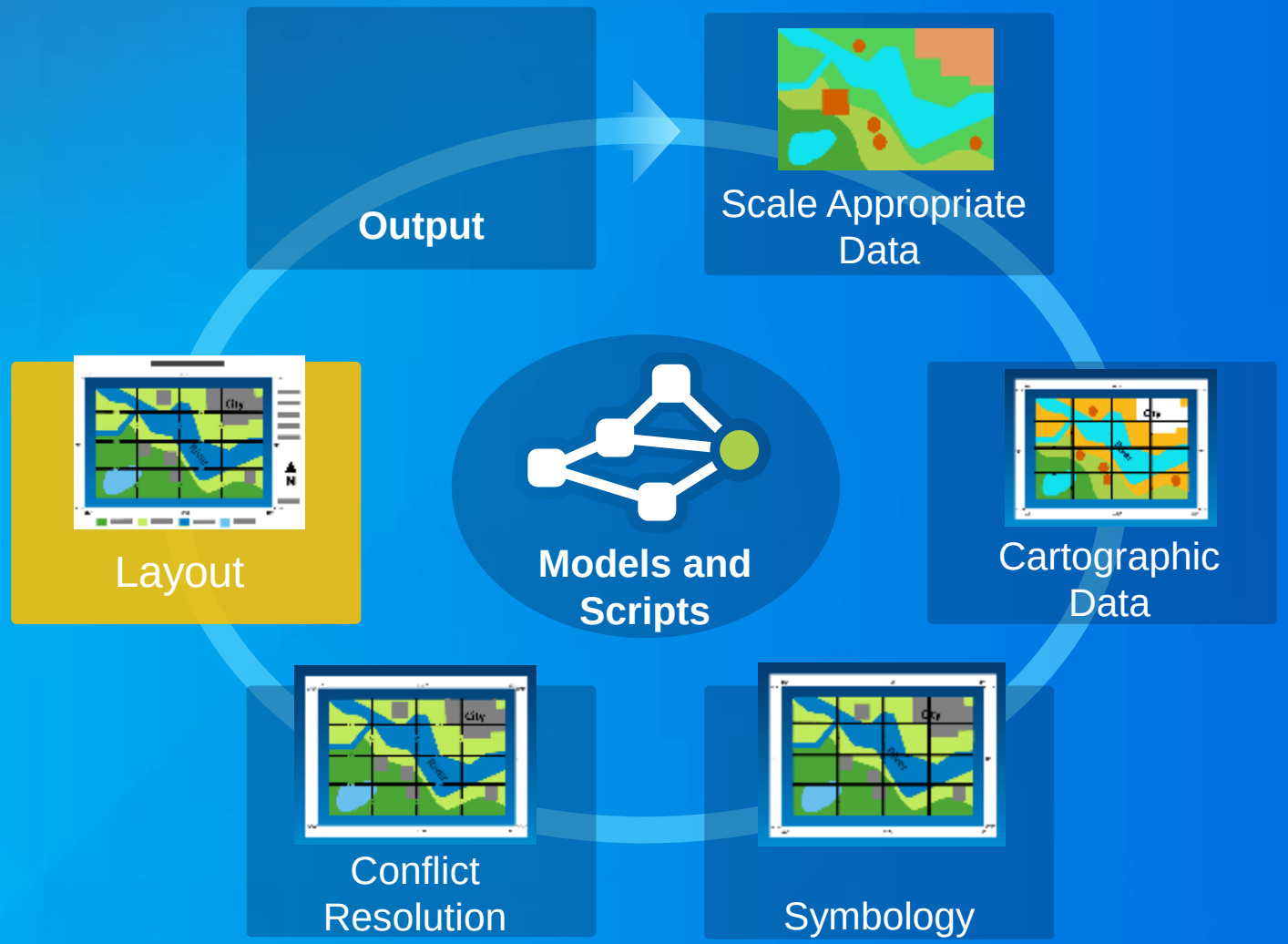

```
# Masking the grid ladder values and annotations
arcpy.AddMessage("getting output of masks.")
mask_layer = arcpy.mapping.Layer(masks.getOutput(0))
arcpy.mapping.AddLayer(data_frame, anno_mask_layer, 'BOTTOM')
anno_mask = arcpy.mapping.ListLayers(final_mxd, mask_layer.name,
                                     data_frame)[0]
arcpy.AddMessage("Annotation Mask layer added to the map...")
arcpyproduction.mapping.EnableLayerMasking(data_frame, 'true')
arcpyproduction.mapping.MaskLayer(data_frame, 'APPEND', anno_mask,
                                  gridline_layer)
arcpy.AddMessage("Masking applied to gridlines...")
```

Demo

Symbology and Conflict Resolution

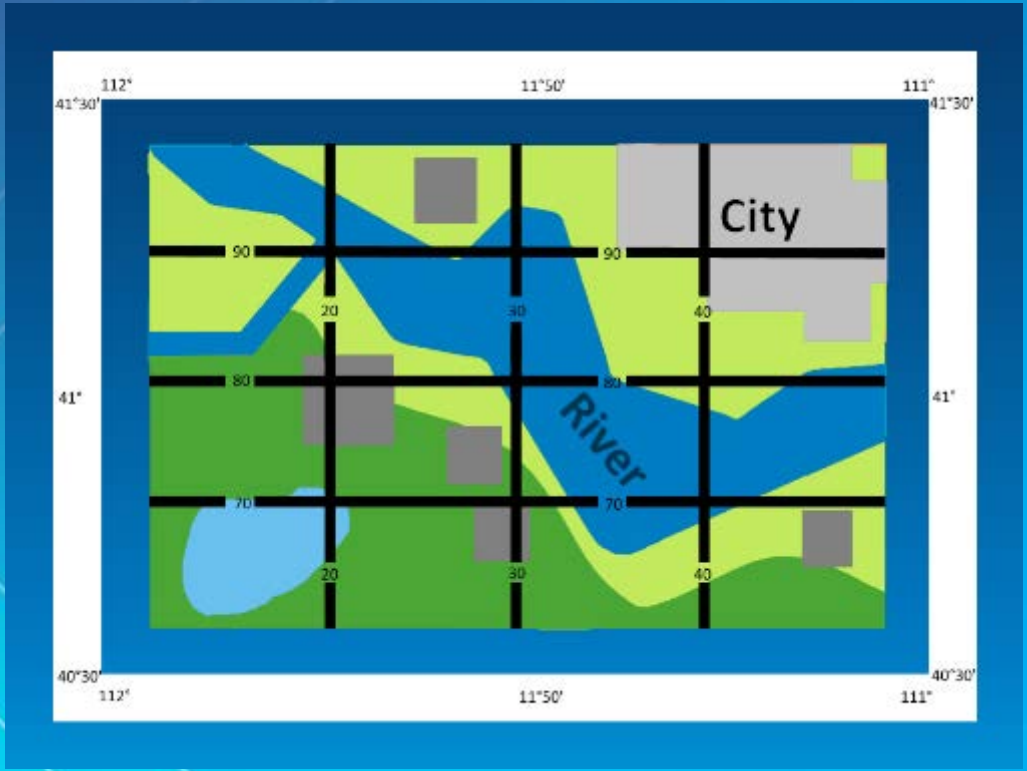


Layout


Output
Scale Appropriate Data
Layout
Models and Scripts
Cartographic Data
Conflict Resolution
Symbology

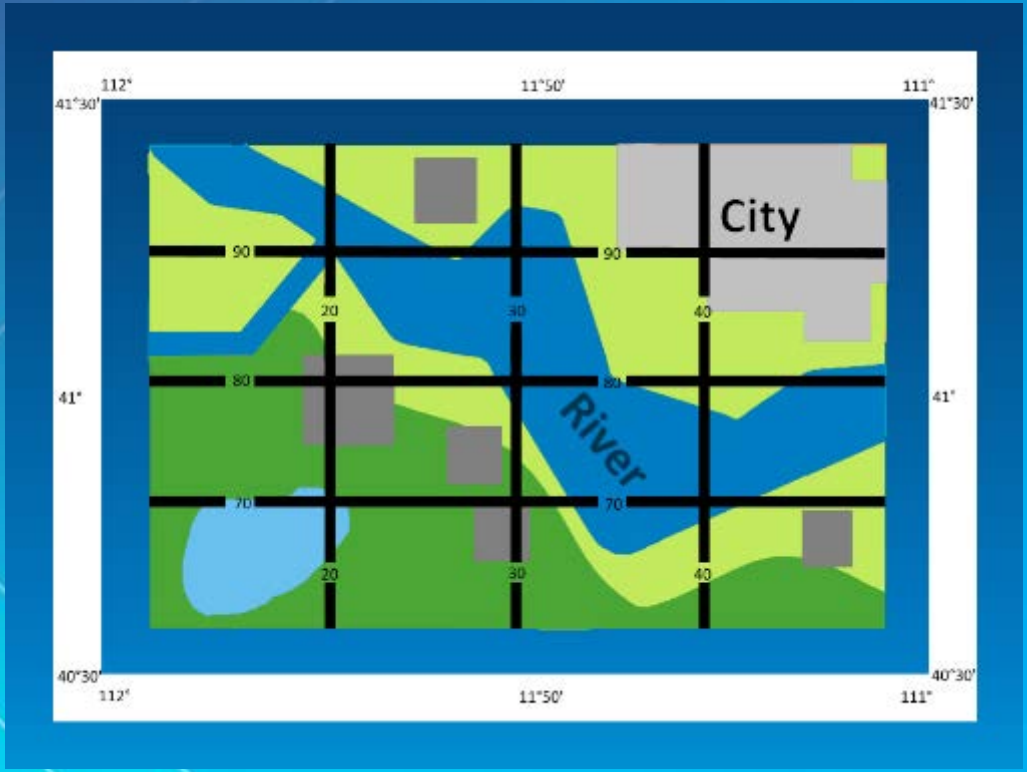
Layout

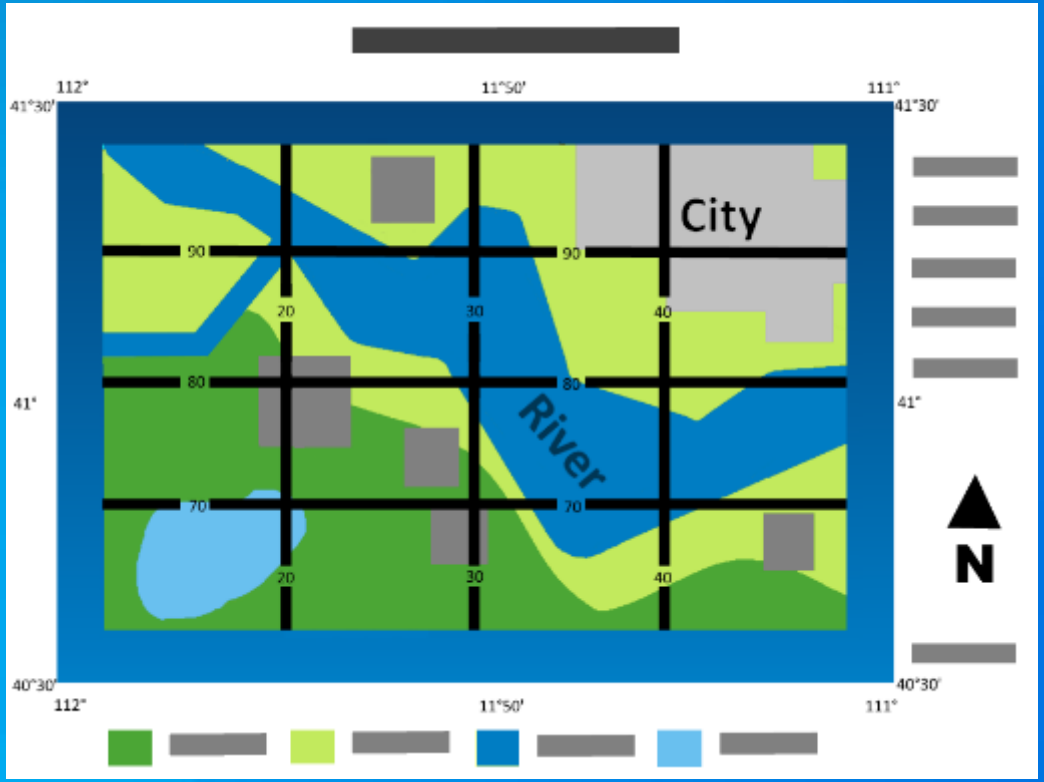
Providing Context



Layout

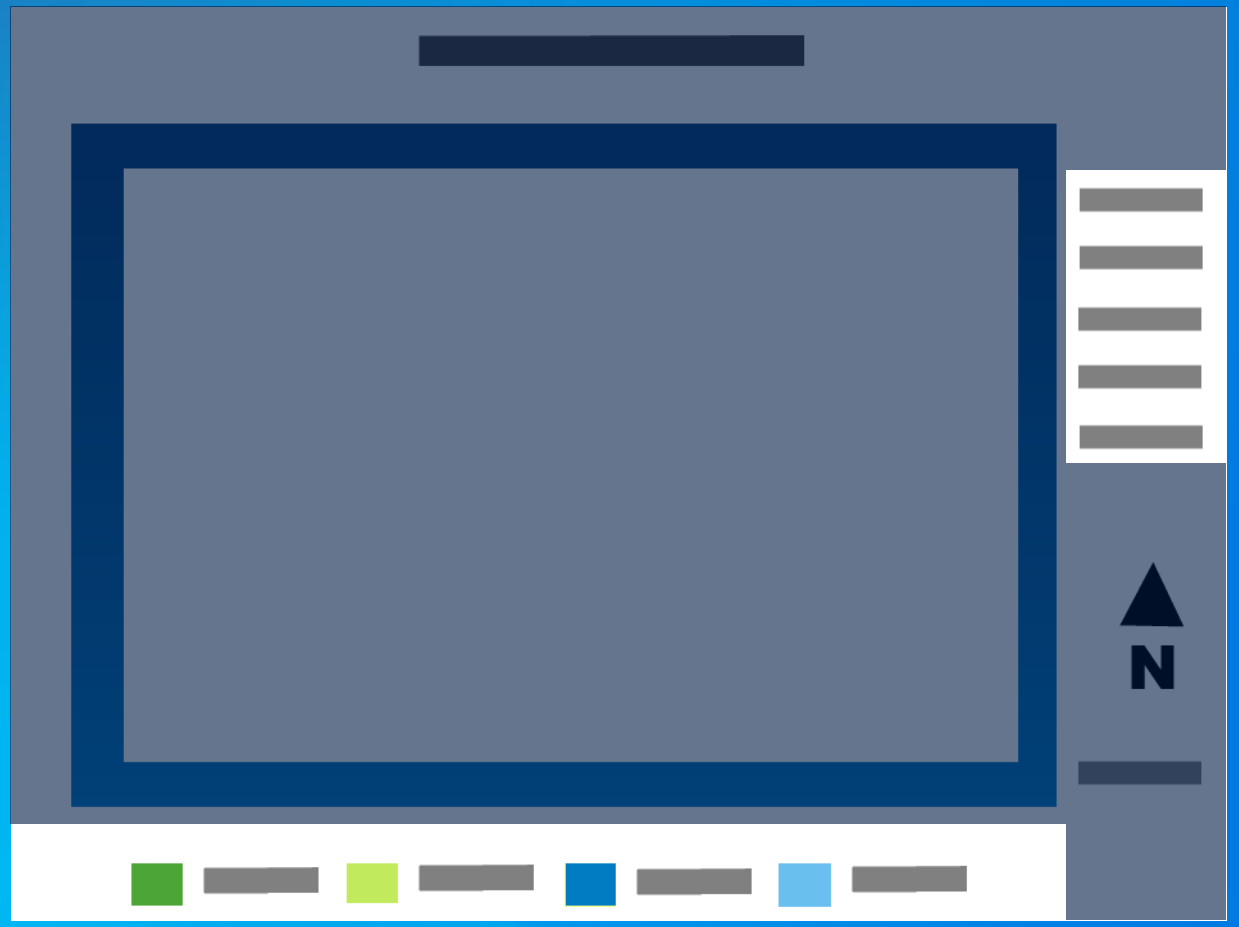
Providing Context





Templates

- Static Elements

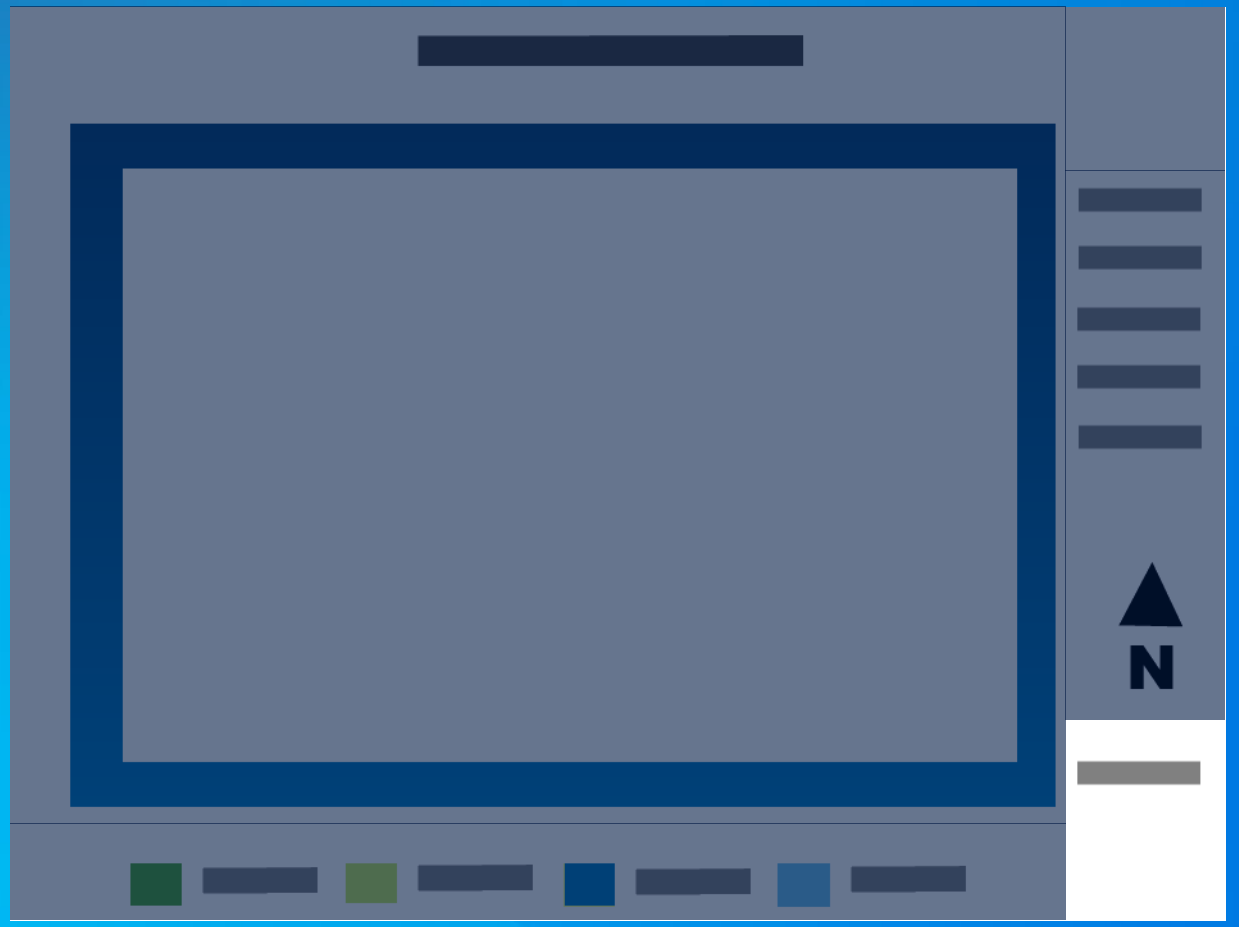


Company Information

Legend

Templates

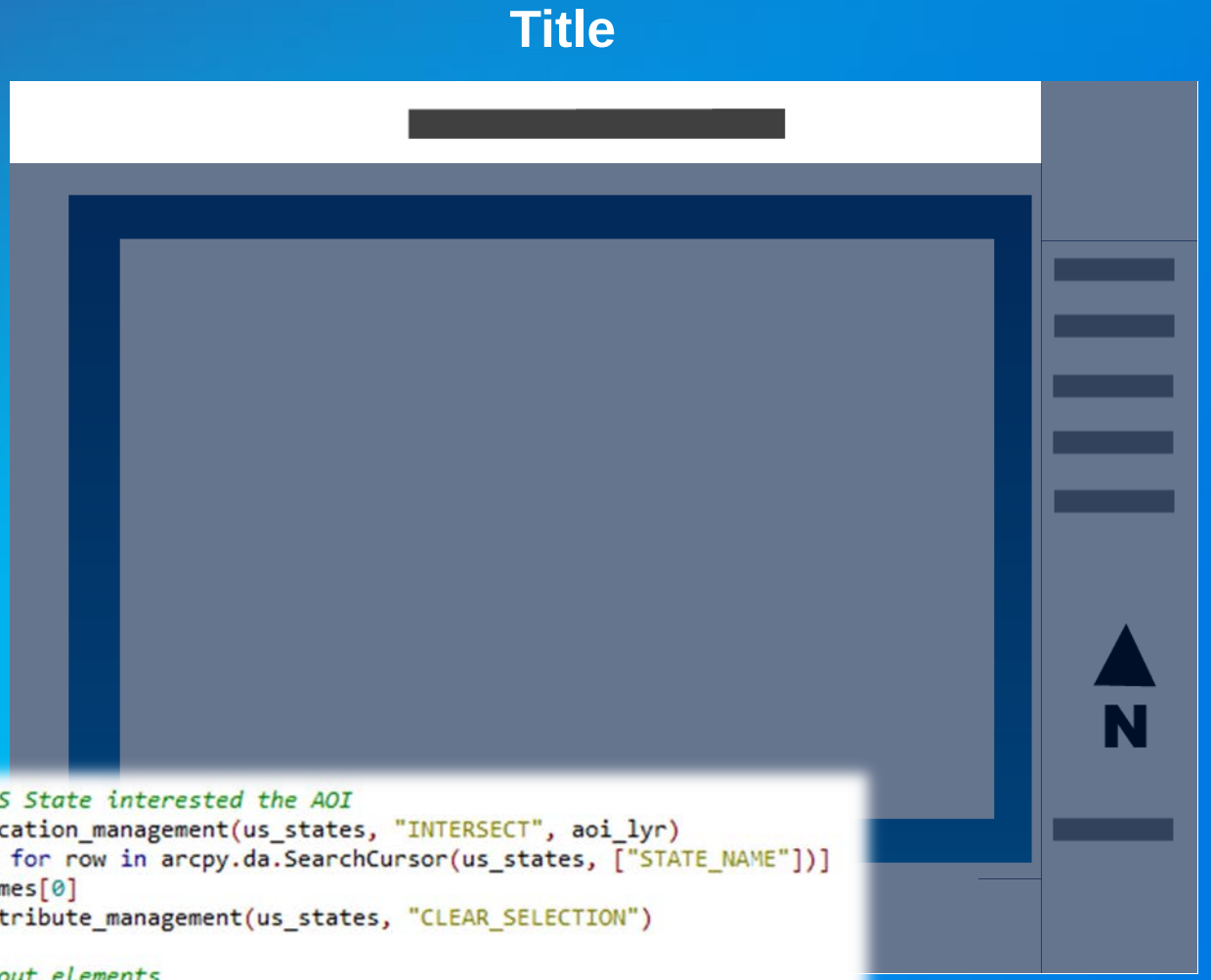
- Static Elements
- Dynamic Elements
  - Dynamic Text



Projection Information

Templates

- Static Elements
- Dynamic Elements
  - Dynamic Text
  - Python




```
# Determining which US State interested the AOI
arcpy.SelectLayerByLocation_management(us_states, "INTERSECT", aoi_lyr)
state_names = [row[0] for row in arcpy.da.SearchCursor(us_states, ["STATE_NAME"])]
state_name = state_names[0]
arcpy.SelectLayerByAttribute_management(us_states, "CLEAR_SELECTION")

#Gets the list of layout elements
layout_elements = arcpy.mapping.ListLayoutElements(final_mxd)
for element in layout_element_list:

    #Update State Name text element
    if element.name == "State Name":
        element.text = state_name.upper()
```

Templates

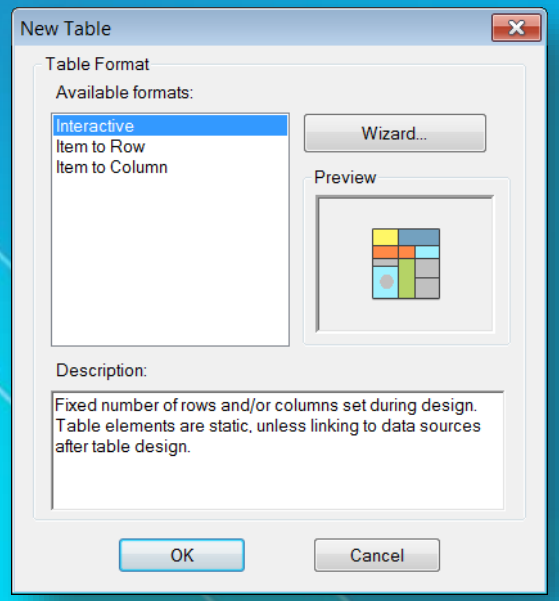
- Static Elements
- Dynamic Elements
  - Dynamic Text
  - Python
  - Graphic Table Element

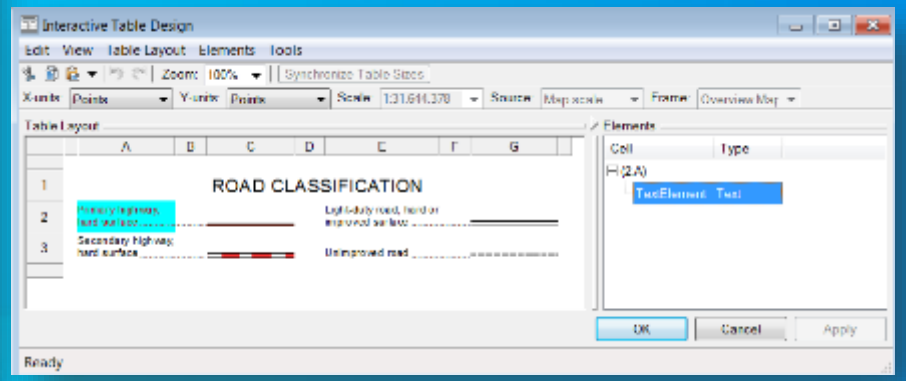
Dynamic Legend
or Table

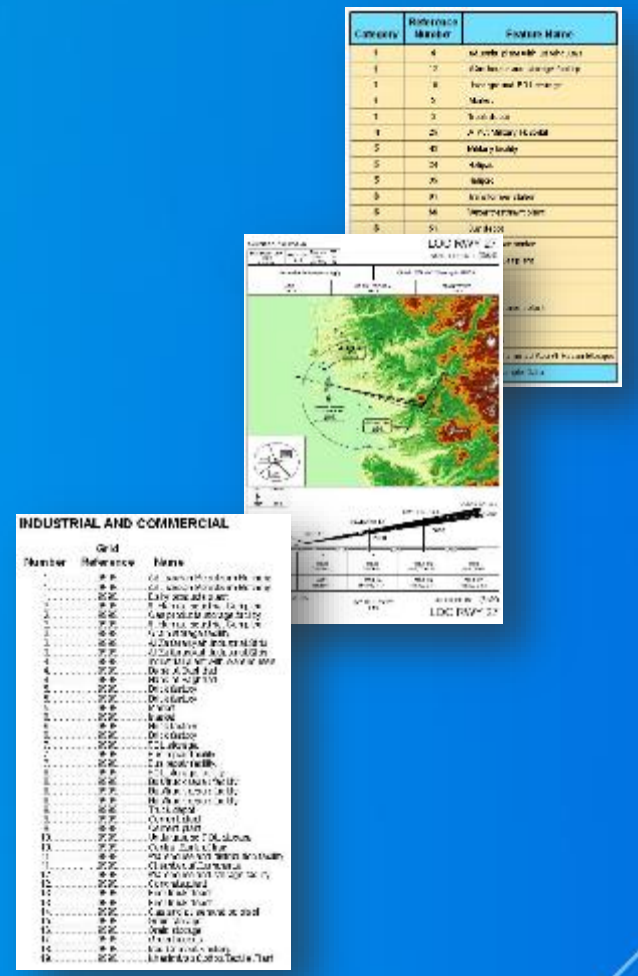
Graphic Table

Dynamic illustrated table

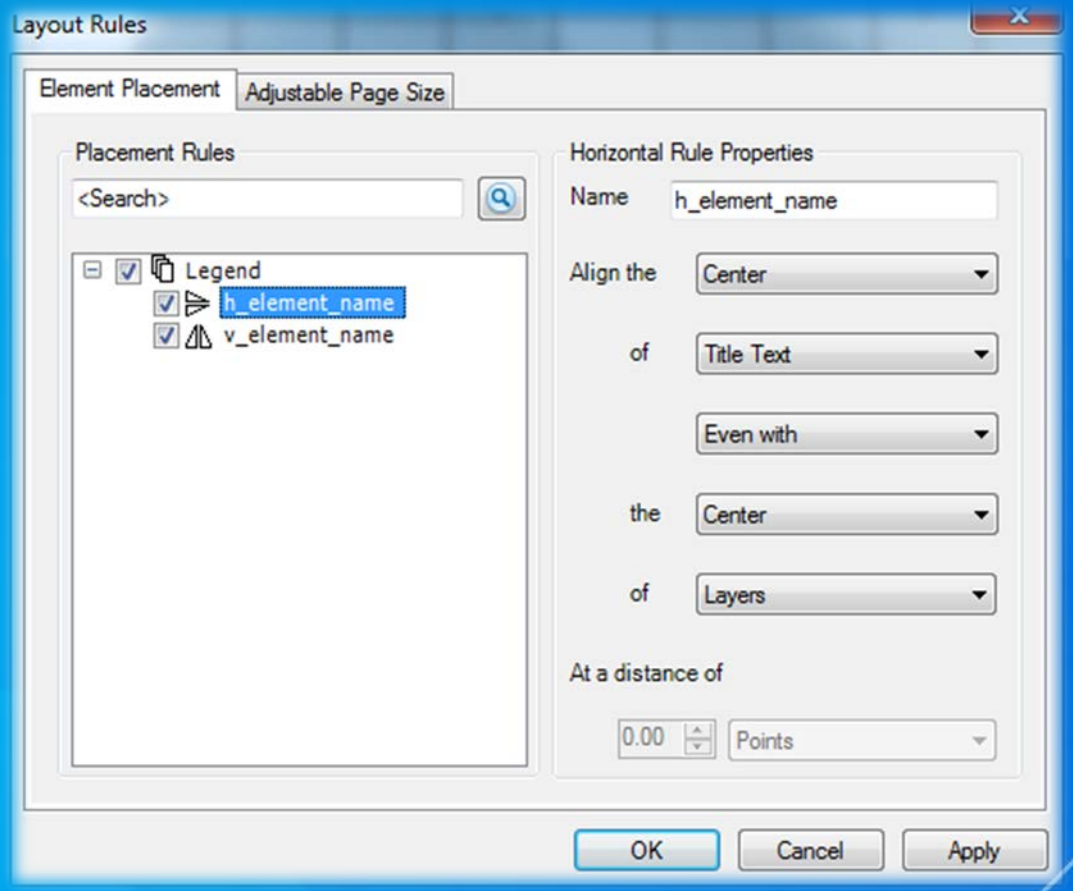
- Dynamic table creation capabilities
- Tables are linked or independent of feature layers
- Ability to include text, symbols and graphics



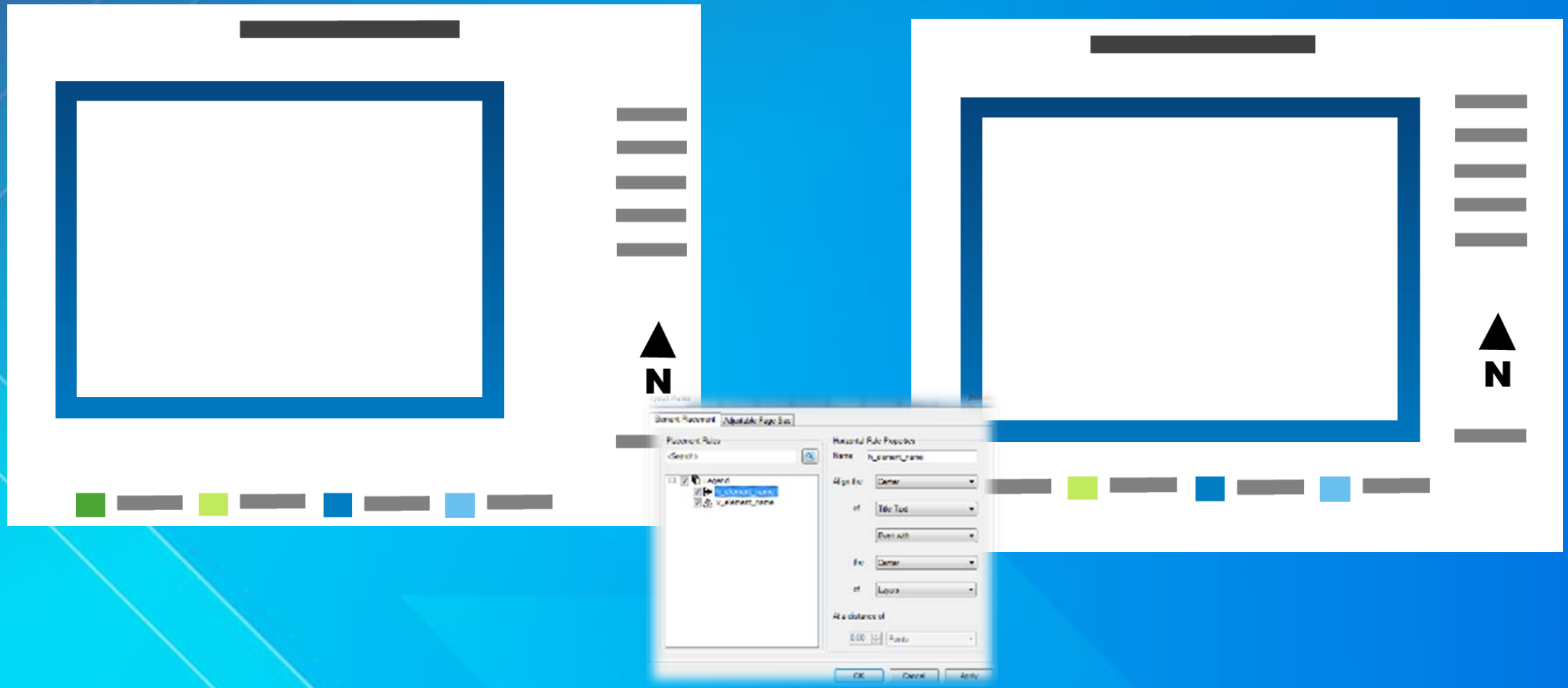




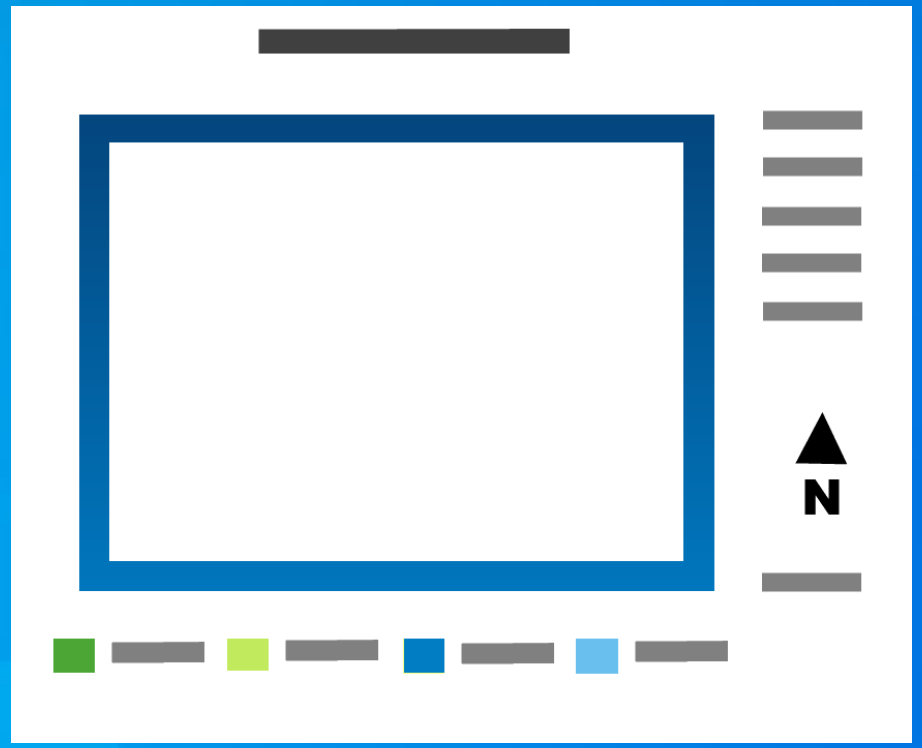
Element Placement



Element Placement

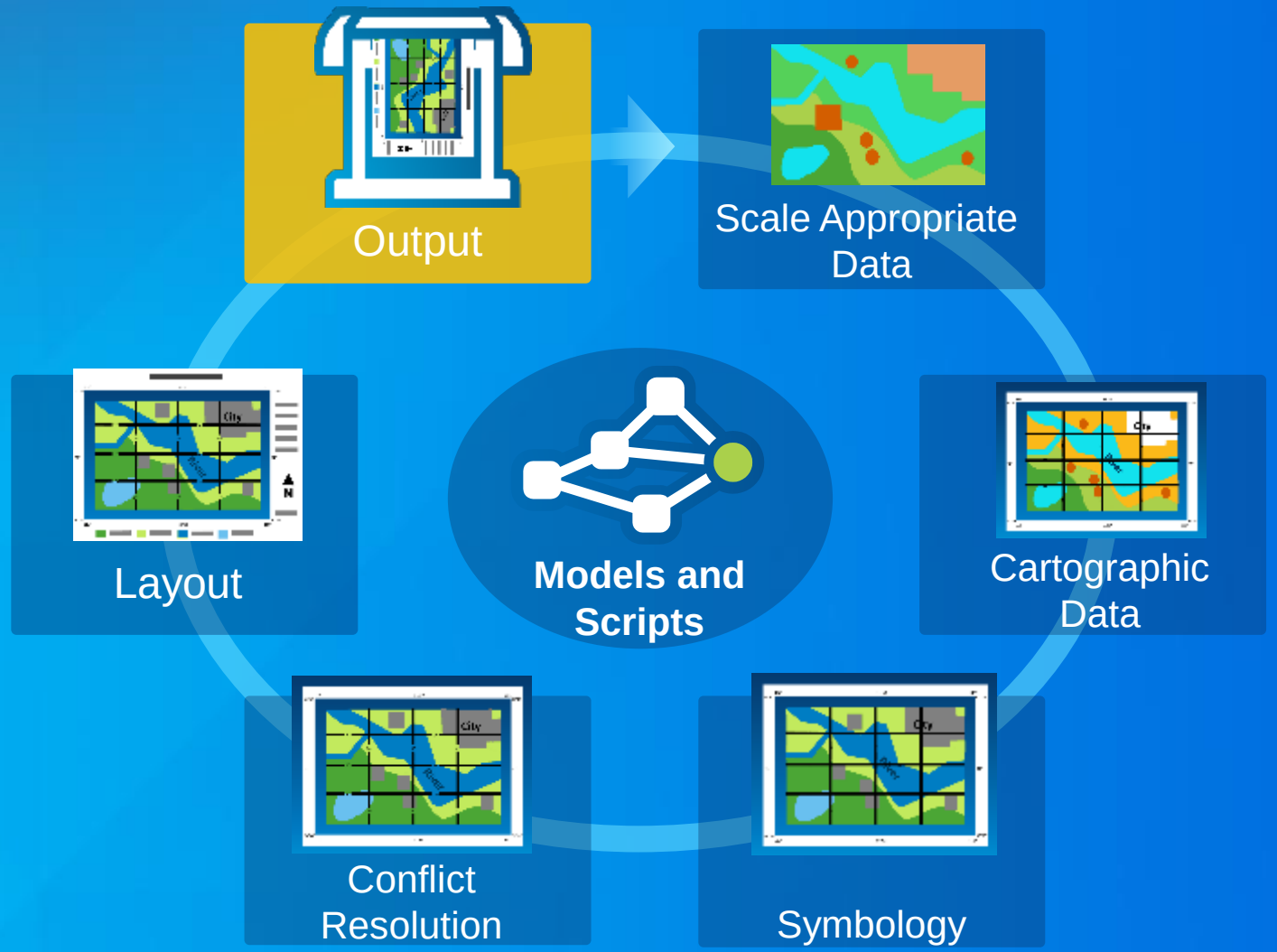


Element Placement



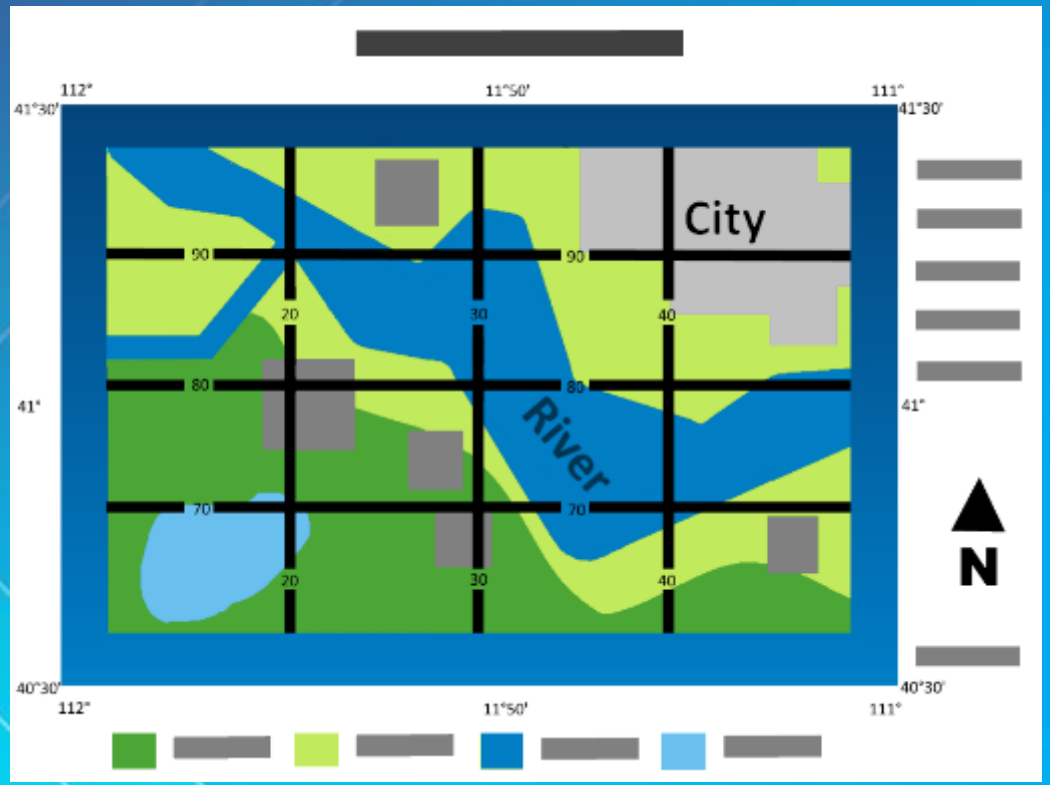

```
mxd = arcpy.mapping.MapDocument("CURRENT")
arcpyproduction.mapping.ApplyLayoutRules(mxd, layout_rules.xml)
```

Output


Output
Scale Appropriate Data
Cartographic Data
Symbology
Conflict Resolution
Layout
Models and Scripts

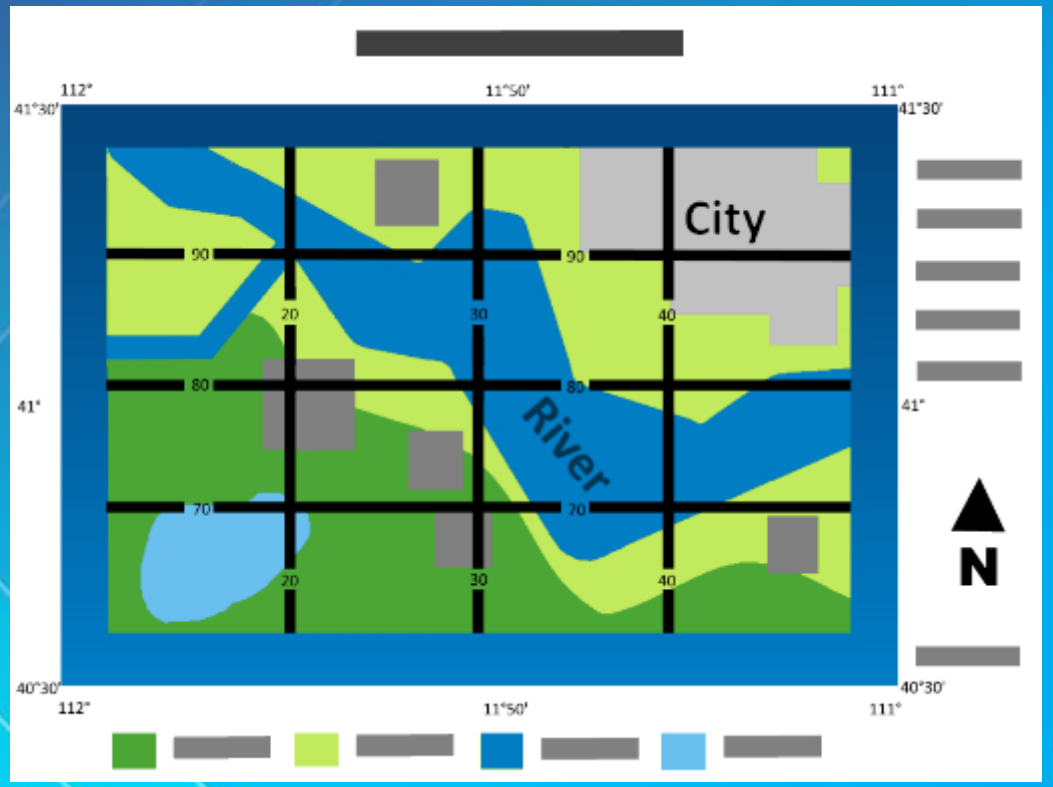
Output

Getting your map to the people who need it

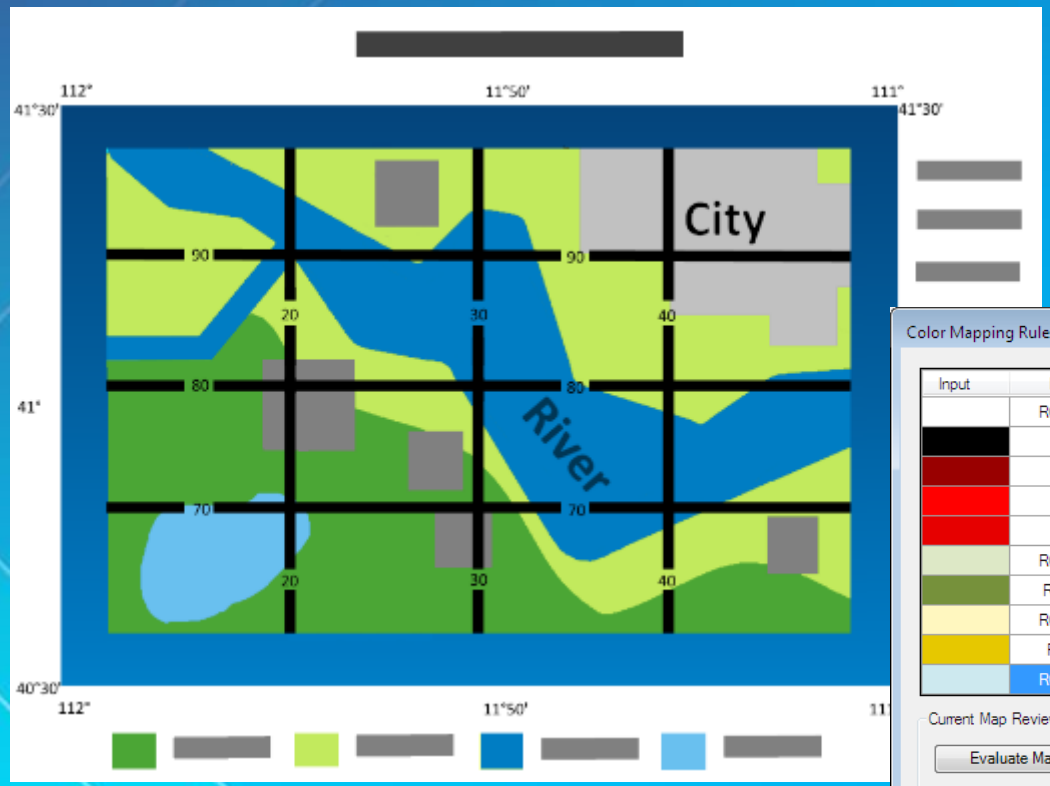


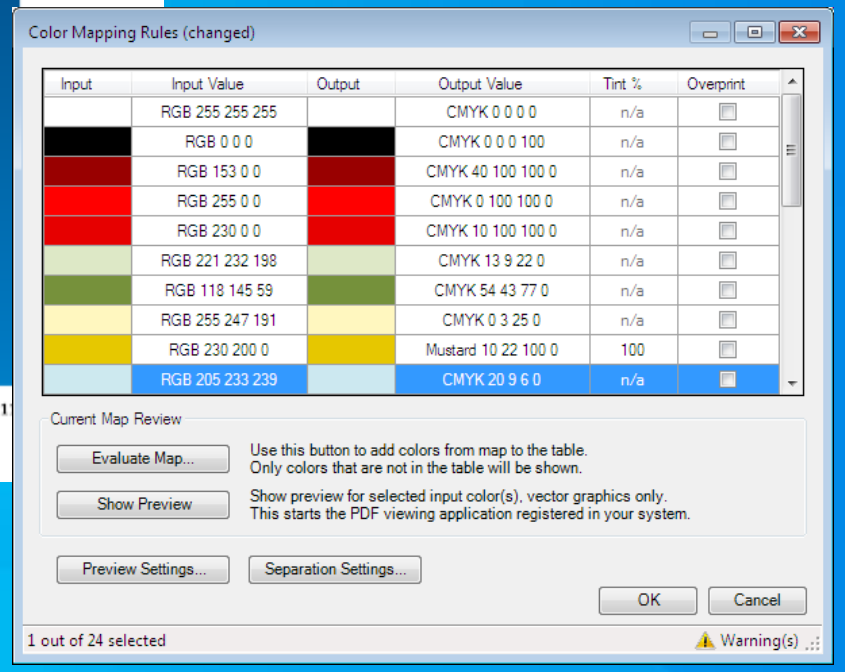
Output

Getting your map to the people who need it



Color Separation





Automating Export

Using Python

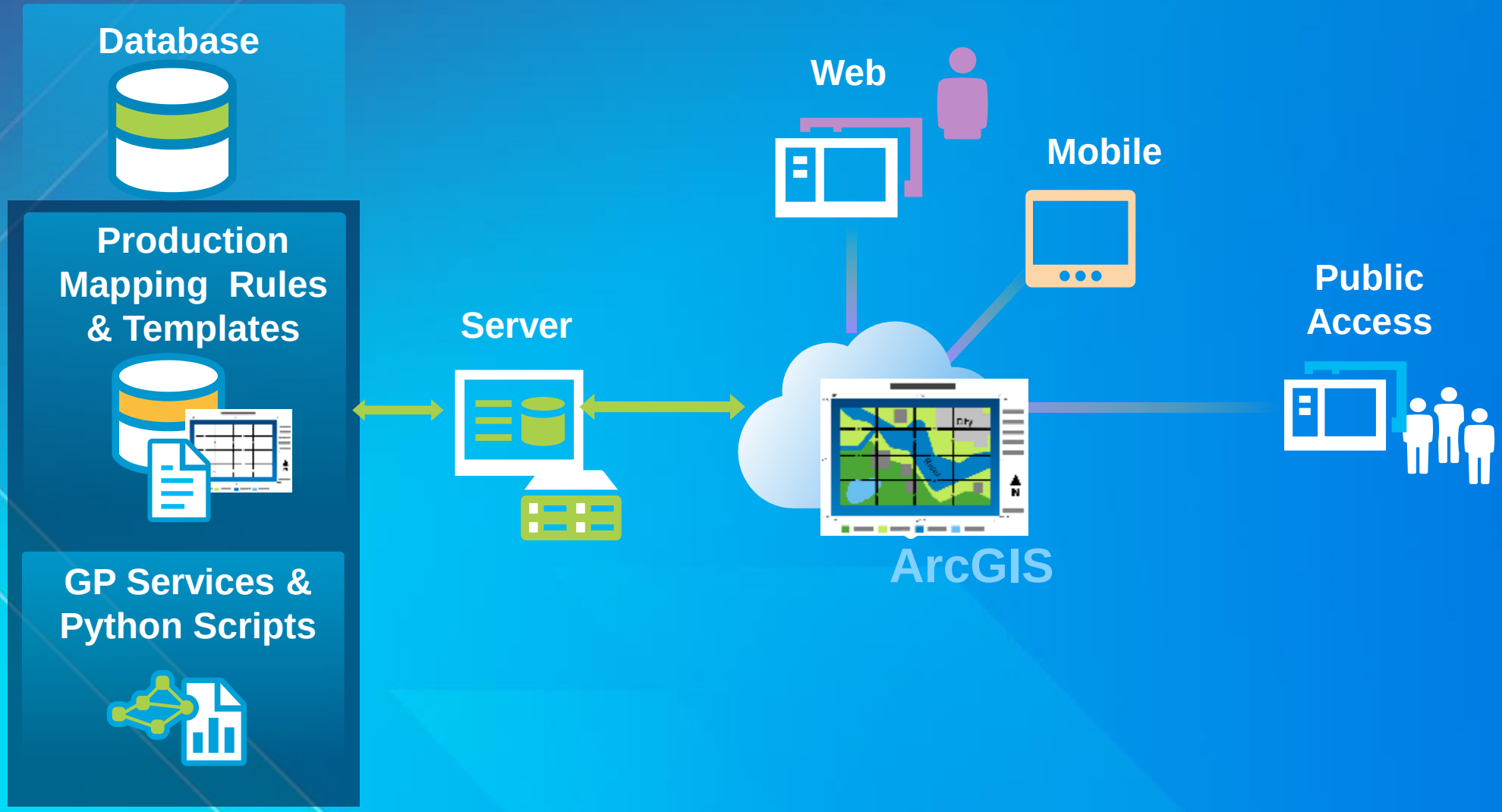
```
if export == 'JPEG':
    filename = map_doc_name  + ".jpg"
    outfile = os.path.join(outputdirectory, filename)

    # Run the export tool
    arcpy.mapping.ExportToJPEG(mxd, outfile)


elif export == "MAP PACKAGE":
    filename = map_doc_name + ".mpk"
    outfile = os.path.join(outputdirectory, filename)
    mxd = mxd.filePath

    # Run the export tool
    arcpy.PackageMap_management(mxd, outfile)


elif export == 'PRODUCTION PDF':
    filename = ap_doc_name  + ".pdf"
    outfile = os.path.join(outputdirectory, filename)
    setting_file = os.path.join(product_location, "colormap.xml")

    arcpyproduction.mapping.ExportToProductionPDF(mxd, outfile, setting_file)

arcpy.AddMessage("Output is located: " + outfile)
```

Production Mapping for ArcGIS Server

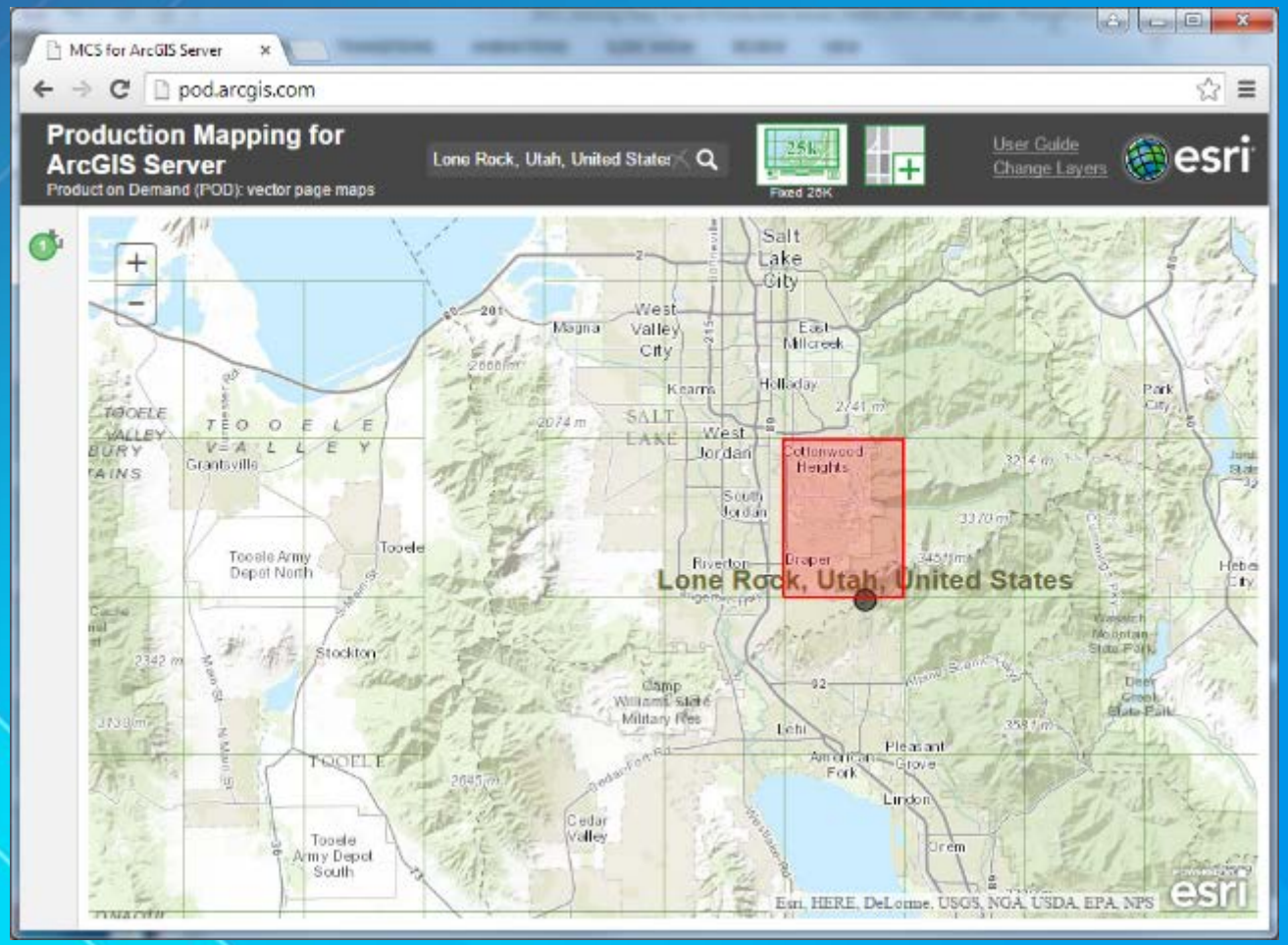
Authoritative Map Product Services

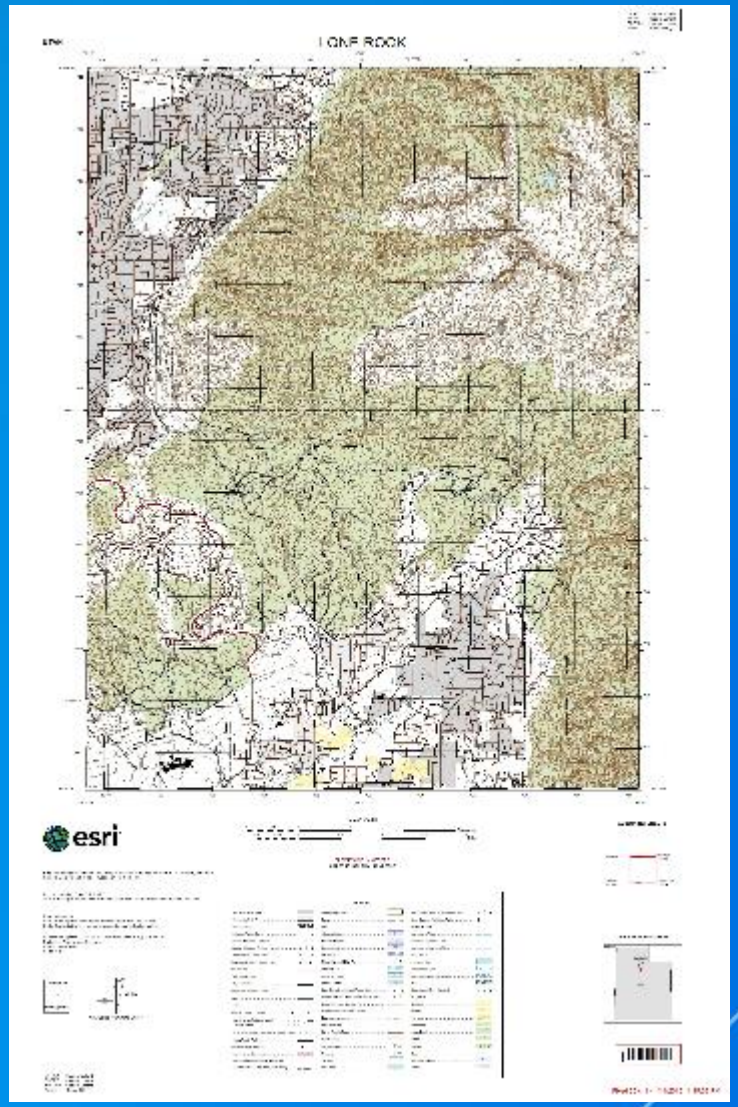
Production Mapping for ArcGIS Server

Authoritative Map Product Services



Product on Demand



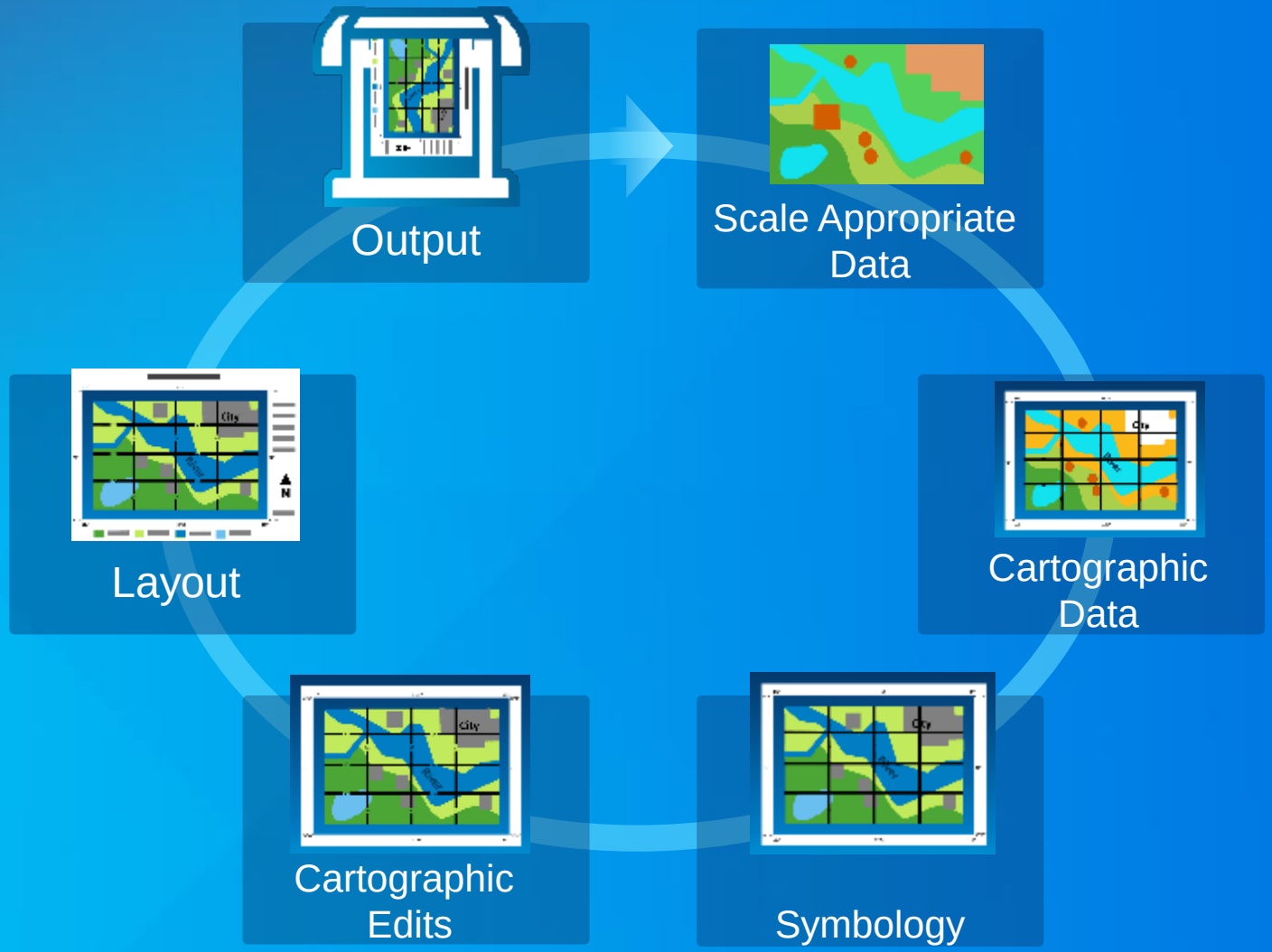


Demo

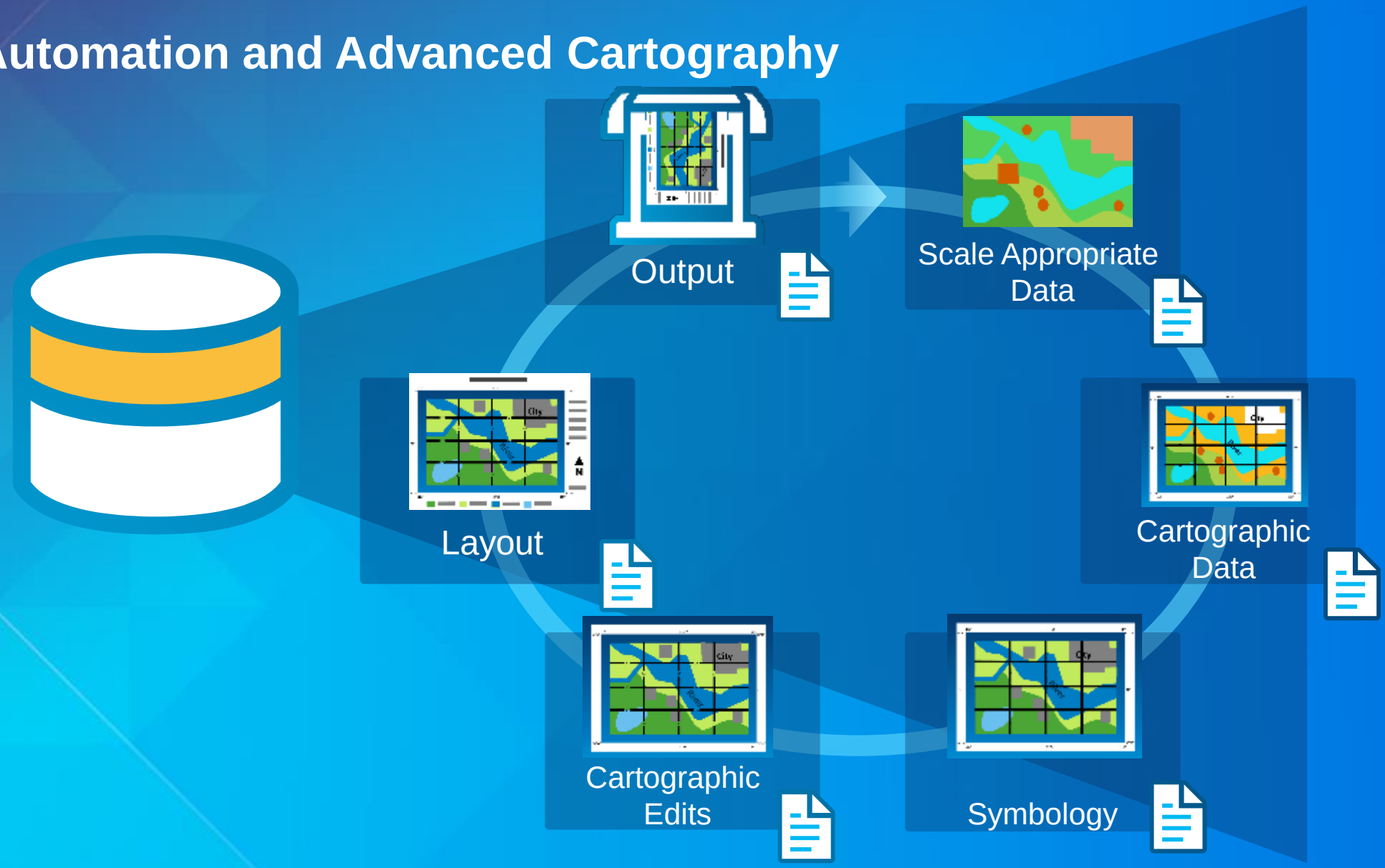
Product on Demand

Conclusion

Map Automation and Advanced Cartography



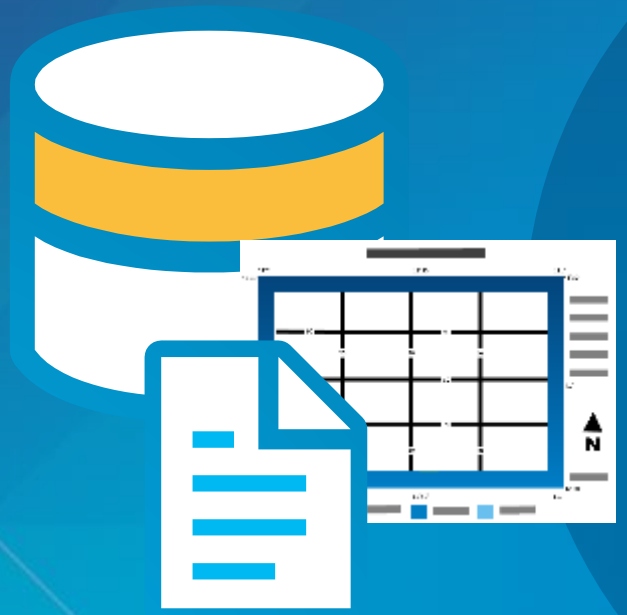
Map Automation and Advanced Cartography



Map Automation and Advanced Cartography

Production Mapping
Cartographic Rules

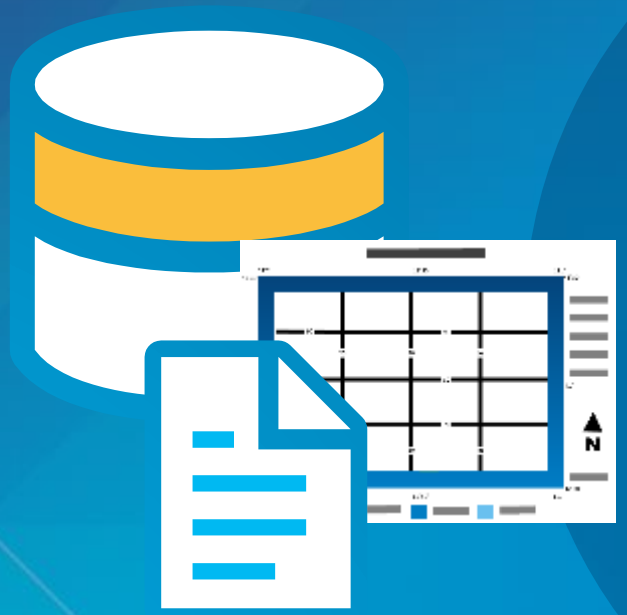
Map Automation and Advanced Cartography



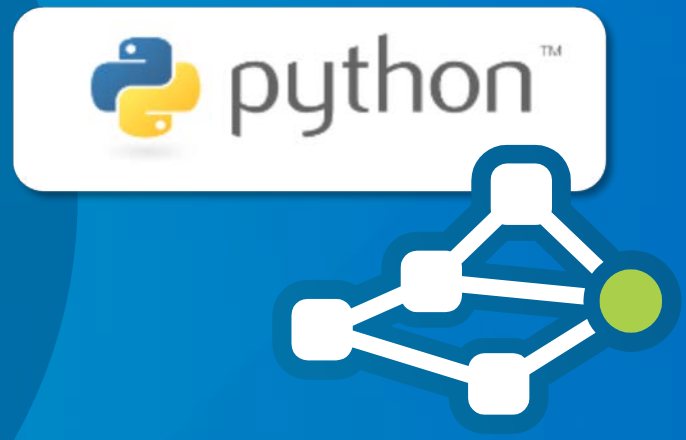
Production Mapping
Cartographic Rules

Geoprocessing
& Python

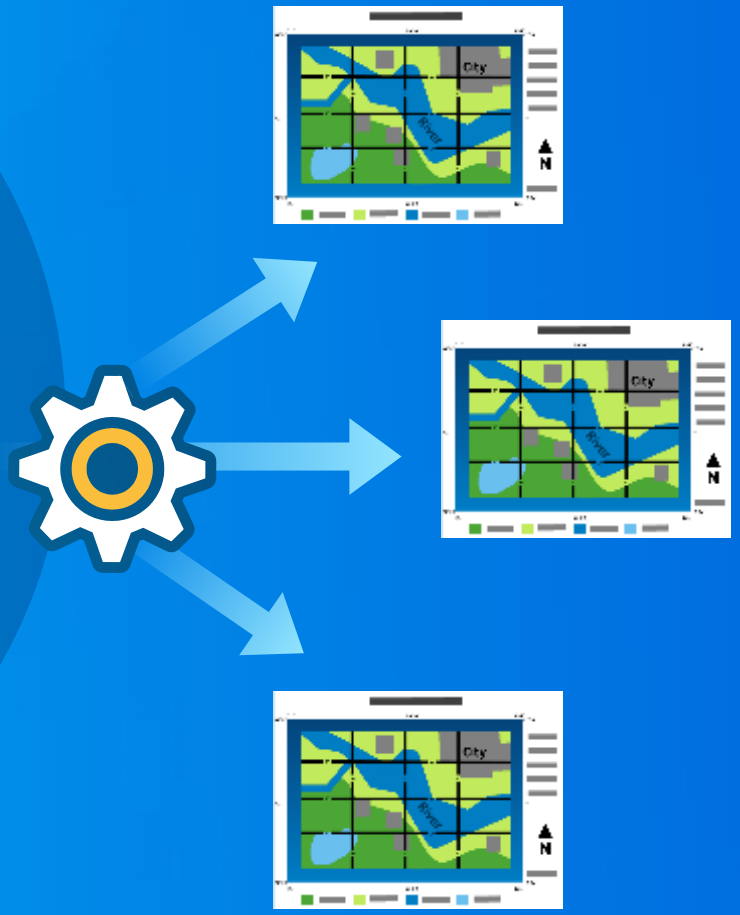
Map Automation and Advanced Cartography



Production Mapping
Cartographic Rules



Geoprocessing
& Python



Thank you…

- **Please fill out the session survey:**

  Esri Production Mapping: Map Automation and Advanced Cartography

UC App

Paper – pick up and put in drop box

References

Get the configurations:

https://github.com/esri/ctm

Product on Demand

Get it:

https://github.com/Esri/product-on-demand

Try it:

http://pod.arcgis.com/

Production Mapping

Learn More:

http://www.esri.com/productionmapping

Email us:

productionmapping@esri.com

Reach out:

MapAutomation_External@esri.com


esri
Understanding our world.