

Using Esri Tools in the Boundary and Annexation Survey Quality Control Process

Kiara Ray

Geography Division

US Census Bureau

***The following slides DO NOT contain Title 13 data**

Boundary and Annexation Survey

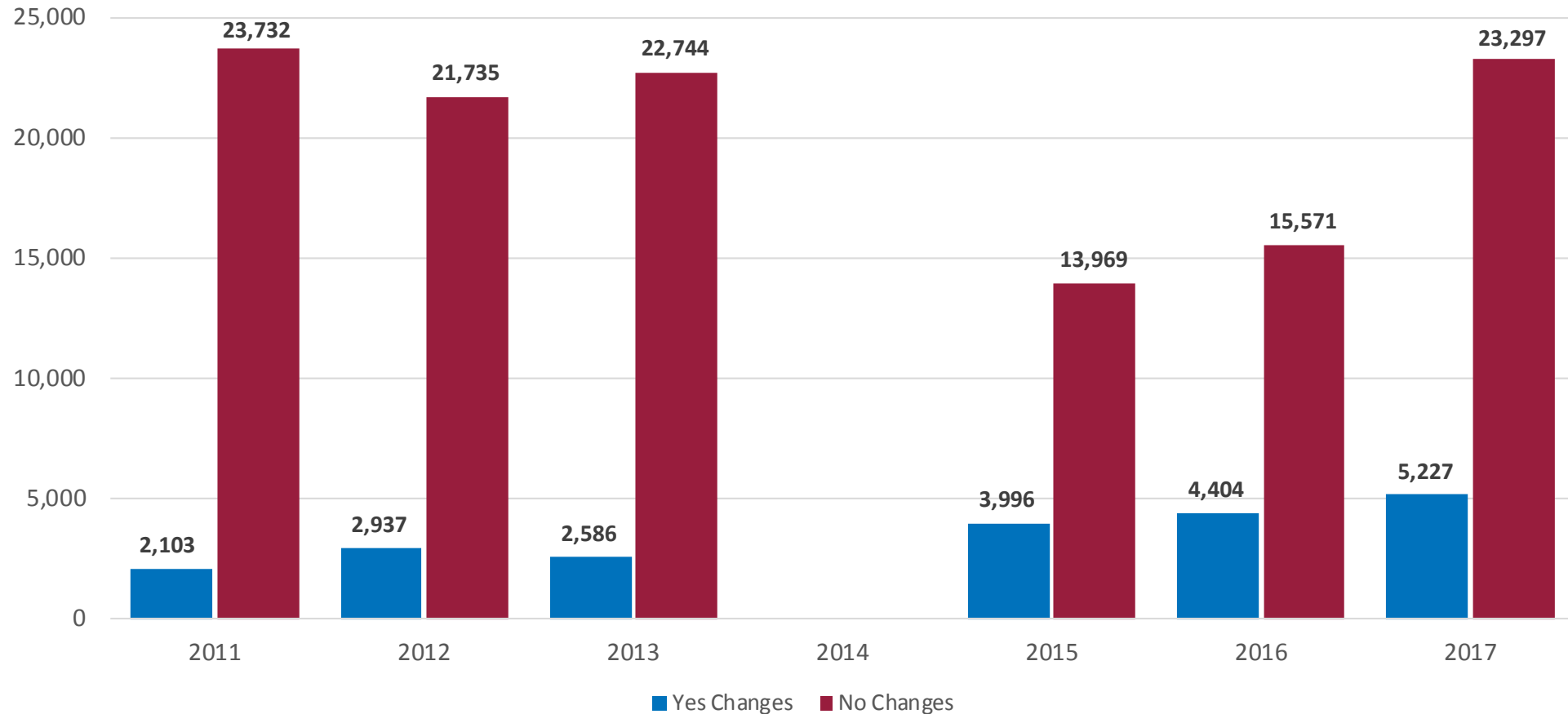


The Boundary and Annexation Survey (BAS) is an annual opportunity for invited governments to:

- o Review boundaries, names, and governmental status.
- o Make corrections and updates to these boundaries, names, and status.

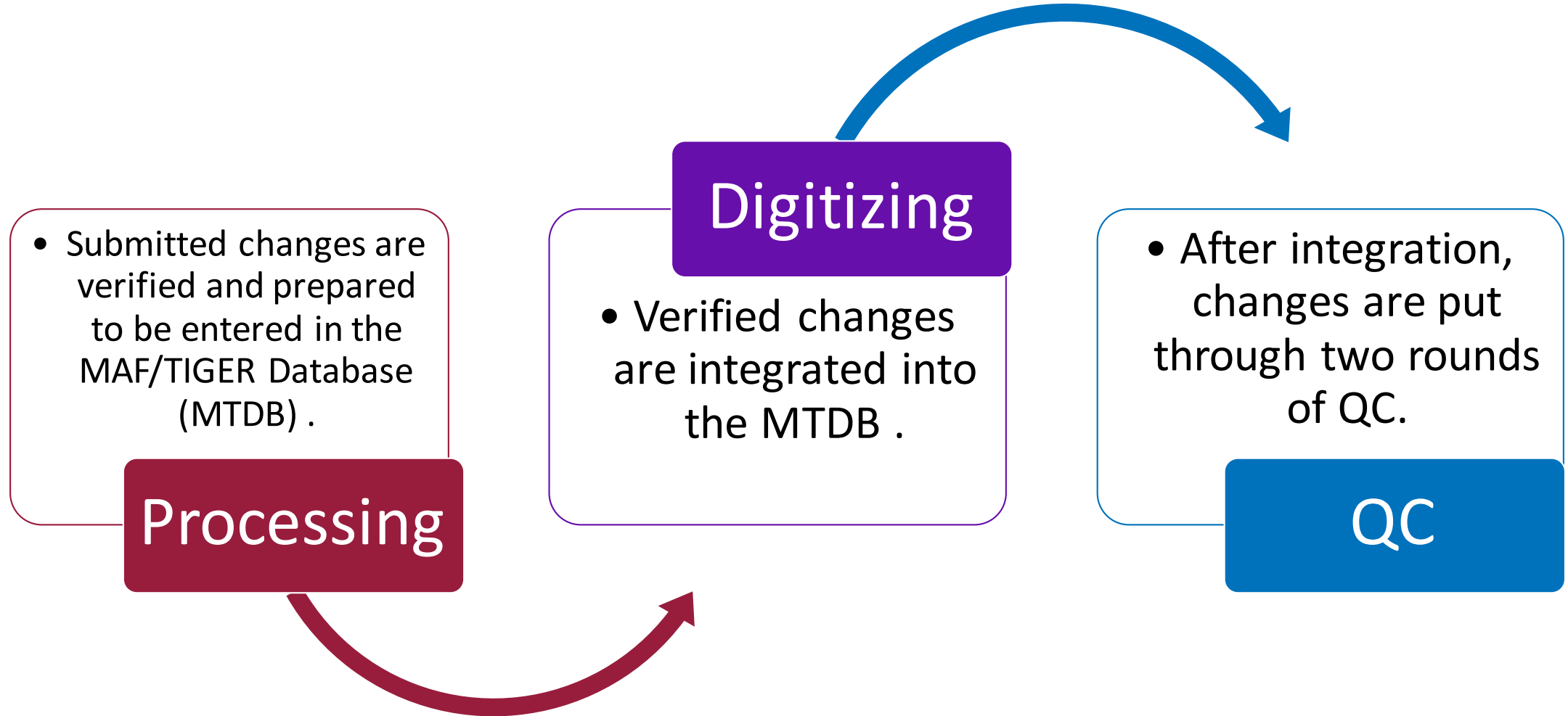
The information collected is used to tabulate data for the decennial and economic censuses, and surveys such as the American Community Survey (ACS).

BAS Response Rates



BAS 2017 had a 92% response rate and processed 67,158 changes!

BAS Process



With Great Power Comes Great Responsibility

Higher response rates and increased participation is great because our data will be more accurate...

....but that means we must process, integrate and QC more changes in the same time frame!

Problem:

How do we make our QC process quicker and more efficient while maintaining our high internal data standards?

Previous QC Script

Real time data is extracted from MTDB and converted to a shapefile.



Map layers for various types of geography are dissolved from the shapefile based on the Geography ID (GEOID).



A .mxd file is created from a map template.

Previous QC Process

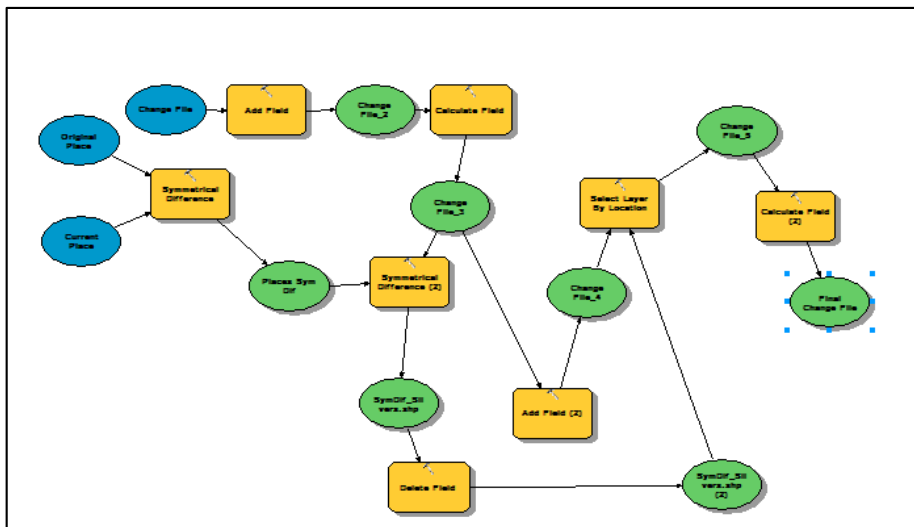
Manually review each polygon, comparing the current geography in our database and the change shapefile submitted by the partner to ensure:

Each update was completed.

No small areas were missed.

All of layers of geography were properly updated.

ModelBuilder and ArcPy to the Rescue!



ModelBuilder and ArcPy were used to improve the QC process.

Easily identify updates that have errors associated with them and quickly locate the errors.

```
# -----
# Find_Slivers.py
# Created on: 2018-05-22 16:13:43.00000
# (generated by ArcGIS/ModelBuilder)
# Description:
# -----

# Set the necessary product code
# import arcinfo

# Import arcpy module
import arcpy

# Local variables:
Change_File = "bas17_changes_inplace"
Change_File_2 = Change_File
Change_File_3 = Change_File_2
Change_File_4 = Change_File_3
Change_File_5 = Change_File_4
Current_Place = "Current Oceanside"
Original_Place = "Original Oceanside"
Places_Sym_Dif = "Test\\Places_SymDif.shp"
SymDif_Slivers_shp = "Test\\SymDif_Slivers.shp"
SymDif_Slivers_shp_2 = SymDif_Slivers_shp
Final_Change_File = Change_File_5

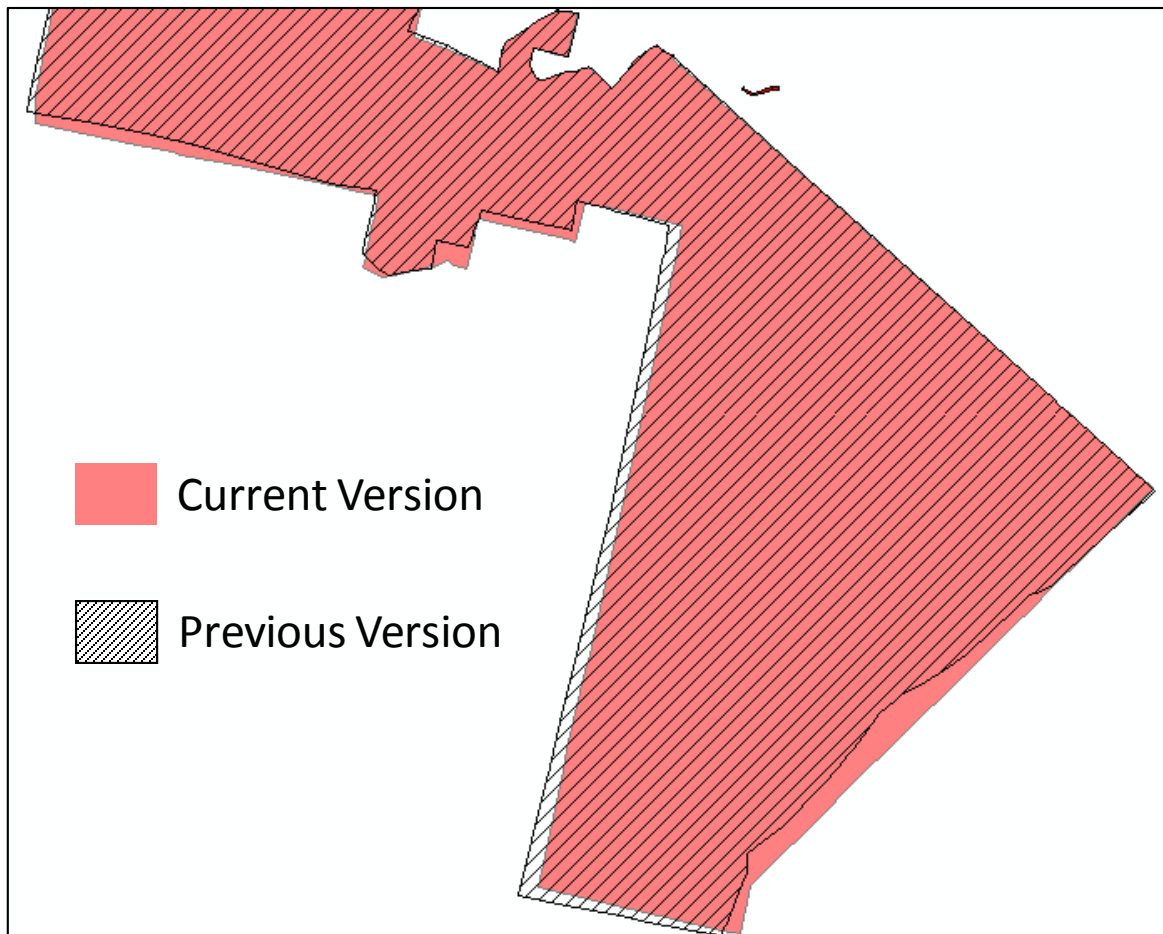
# Process: Add Field
arcpy.AddField_management(Change_File, "Feature_ID", "SHORT", "", "", "", "", "NULLABLE", "NON_REQUIRED", "")

# Process: Calculate Field
arcpy.CalculateField_management(Change_File_2, "Feature_ID", "{OBJECTID}", "VB", "")

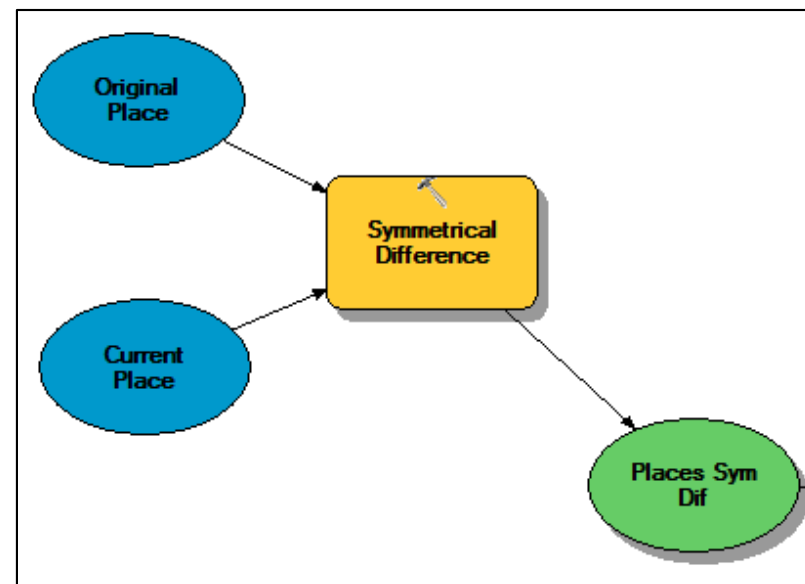
# Process: Add Field (2)
arcpy.AddField_management(Change_File_3, "Error", "TEXT", "", "", "", "", "NULLABLE", "NON_REQUIRED", "")

# Process: Symmetrical Difference
arcpy.SymDiff_analysis(Current_Place, Original_Place, Places_Sym_Dif, "ALL", "")
```

ModelBuilder – Find the Errors



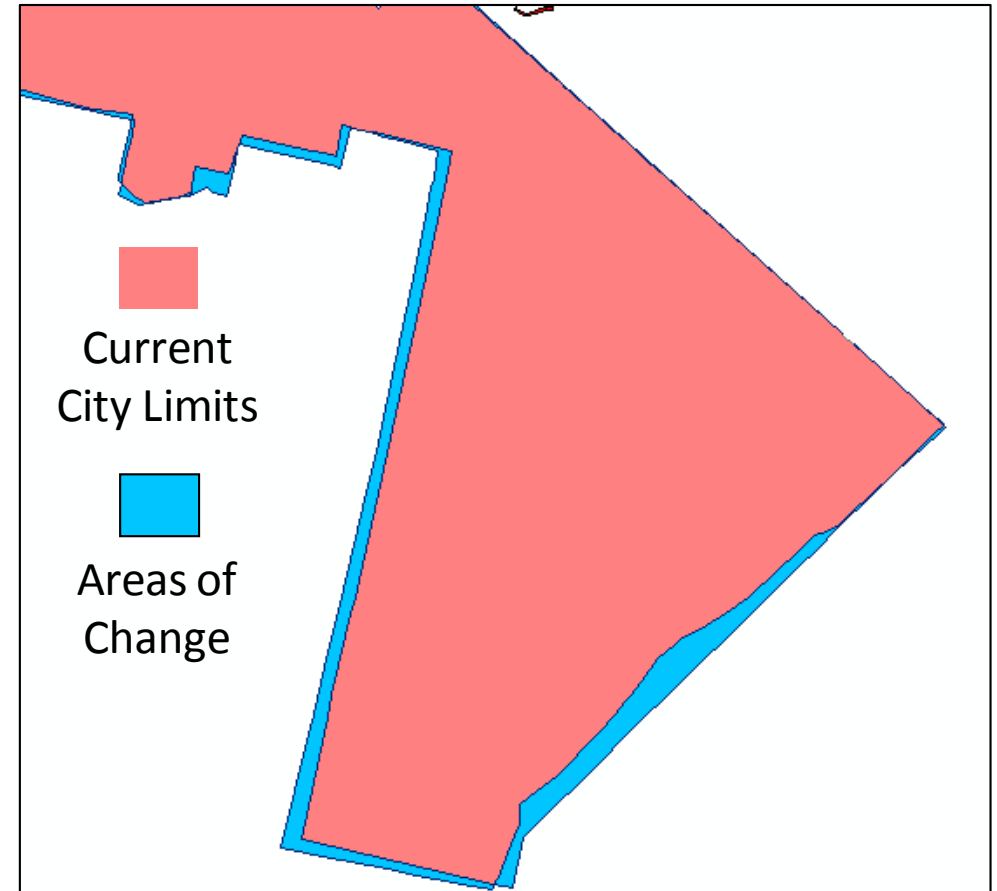
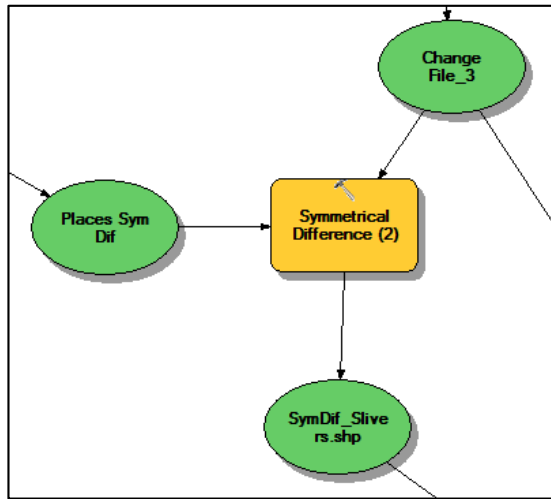
Run a symmetrical difference between the previous version of the city and the current version of the city to find the areas of change.



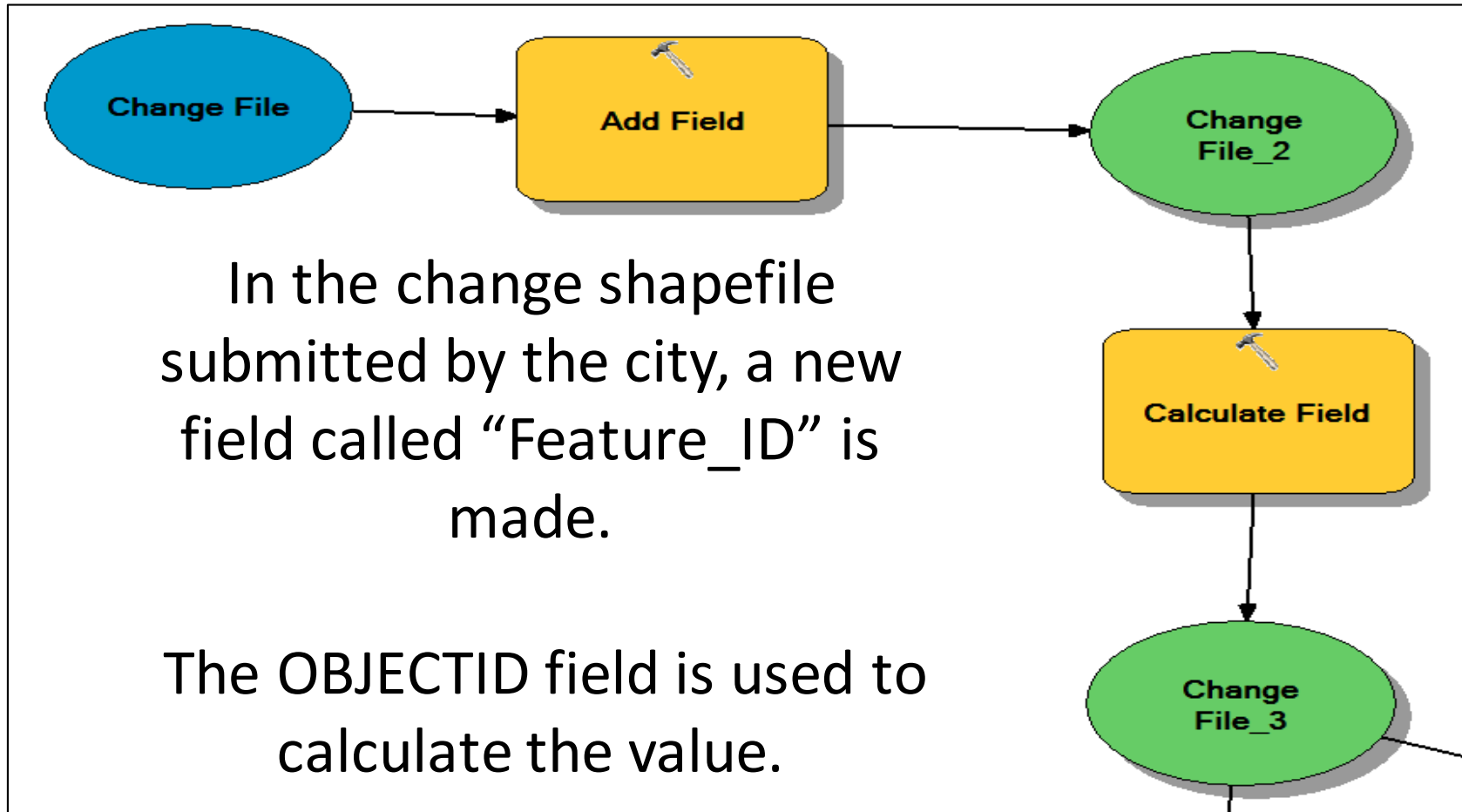
ModelBuilder – Find the Errors

Run a second symmetrical difference between the areas of change and the change shapefile submitted by the city.

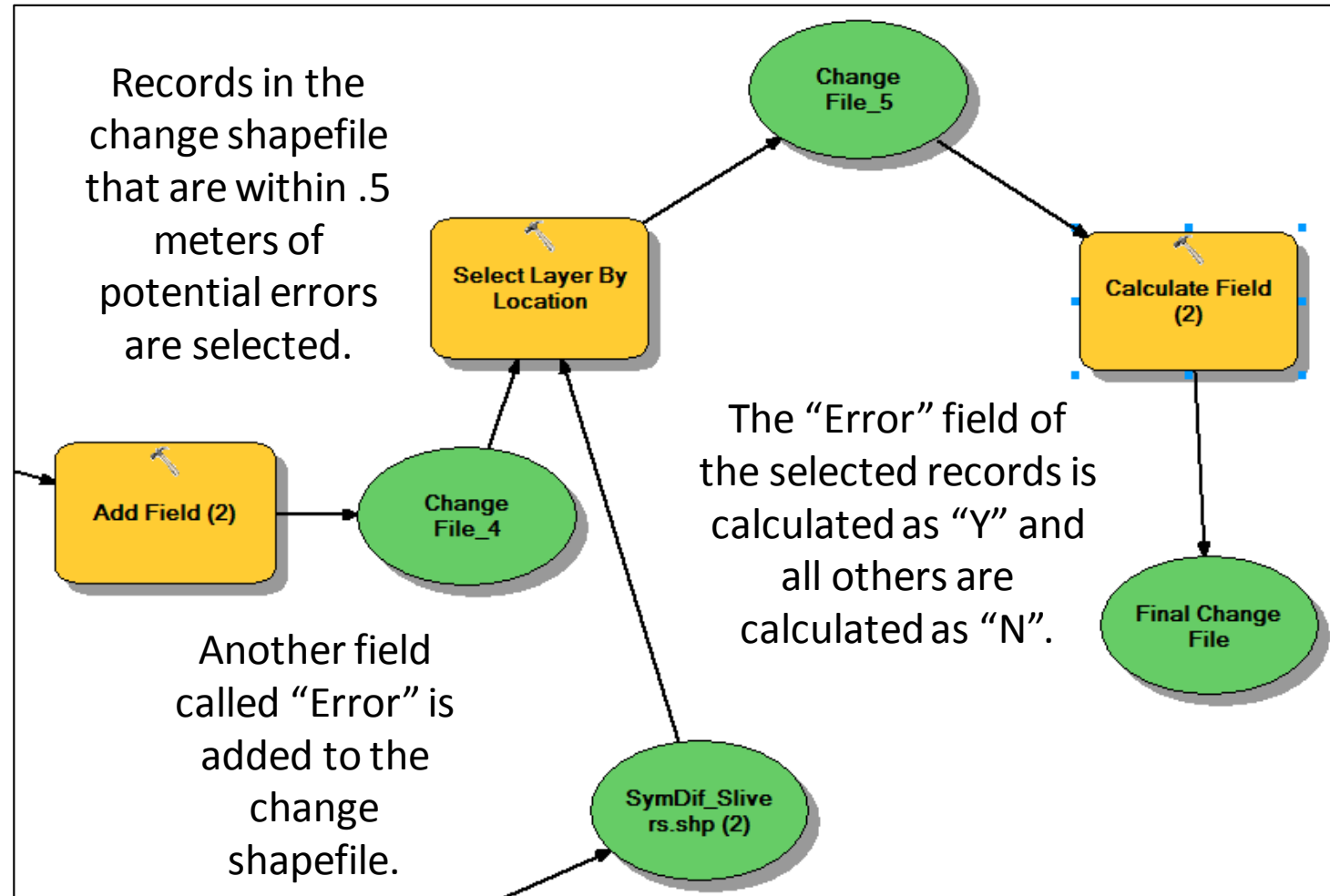
The result is a shapefile that contains all the of potential mistakes.



ModelBuilder – Associate Errors



ModelBuilder – Associate Errors



ArcPy

```
# This will check to see if processed changes are within an acceptable tolerance of the requested change
def rightTool(gdb):
    arcpy.AddMessage("Checking tolerances.")
    basID = gdb[-15:-4]
    benchmark = os.path.join(gdb, 'benchmark')
    submission = os.path.join(gdb, 'submission')

    inplace_changes = os.path.join(submission, 'bas18_{}_changes_inplace'.format(basID))
    cousub_changes = os.path.join(submission, 'bas18_{}_changes_cousub'.format(basID))
    county_changes = os.path.join(submission, 'bas18_{}_county_changes'.format(basID))

    live_place = os.path.join(gdb, 'live_places')
    live_cousub = os.path.join(gdb, 'live_cousubs')
    live_county = os.path.join(gdb, 'live_cousubs')

    benchmark_place = os.path.join(benchmark, 'bas_place')
    benchmark_cousub = os.path.join(benchmark, 'bas_cousub')
    benchmark_county = os.path.join(benchmark, 'bas_county')

    if inplace_changes:
        arcpy.AddField_management(inplace_changes, "ERROR", "TEXT", 1)

        symmetricalDiff1 = arcpy.SymDiff_analysis(live_place, benchmark_place, os.path.join(gdb, "symmetricalDiff1"))
        symmetricalDiff2 = arcpy.SymDiff_analysis(symmetricalDiff1, inplace_changes, os.path.join(gdb, "symmetricalDiff2"), "ALL", "0.4 Meters")

        inplace_changes_layer = arcpy.MakeFeatureLayer_management(inplace_changes, "inplace_changes_layer")

        arcpy.SelectLayerByLocation_management(inplace_changes_layer, "WITHIN_A_DISTANCE", symmetricalDiff2, "0.5 Meters", "NEW_SELECTION", "INVERT")
        arcpy.CalculateField_management(inplace_changes_layer, "ERROR", "\"N\"", "PYTHON_9.3", "")
        arcpy.SelectLayerByAttribute_management(inplace_changes_layer, "CLEAR_SELECTION")

        arcpy.SelectLayerByLocation_management(inplace_changes_layer, "WITHIN_A_DISTANCE", symmetricalDiff2, "0.5 Meters", "NEW_SELECTION", "NOT_INVERT")
        arcpy.CalculateField_management(inplace_changes_layer, "ERROR", "\"Y\"", "PYTHON_9.3", "")
        arcpy.SelectLayerByAttribute_management(inplace_changes_layer, "CLEAR_SELECTION")
```

The model was exported as a python script and modified to accommodate various types of BAS submissions.

The modified script was inserted in the current QC script as a function and called before the creation of .mxd file.

```
try:
    rightTool(gdb)
except:
    pass

arcpy.AddMessage("Creating QC MXD.")

# Create and populate the mxd
basid = os.path.split(gdb)[1][2:-4]
state = basid[1:3]

mxd = gdb.replace('.gdb', 'QC.mxd')
shutil.copy(mxdtemplate, mxd)
MXD = arcpy.mapping.MapDocument(mxd)
layers = arcpy.mapping.ListLayers(MXD)

for layer in layers:
    if layer.isFeatureLayer:
        newfc = layer.datasetName.replace('template', basid)
        newpath = os.path.join(gdb, newfc)
        if layer.isFeatureLayer and arcpy.Exists(newpath):
            layer.replaceDataSource(gdb, "FILEGDB_WORKSPACE", newfc)
            layer.name = layer.name.replace('template', basid)
        else:
            continue

DF = arcpy.mapping.ListDataFrames(MXD)[0]
layers = arcpy.mapping.ListLayers(MXD)
```

Results

This new process is more precise and saves time:

- Decreased QA time for shapefiles with more than 50 records by 28%.
- Caught small mistakes that were missed by human eyes.
- Easily located mistakes.
- Located areas that were incorrectly modified and not associated with any changes.

Contact

Kiara.ray@census.gov

Boundary and Annexation Survey –

<https://www.census.gov/programs-surveys/bas.html>