



AUTOMATED MAPPING THROUGH PYTHON

Matt Sokol and Jessica Curtis
December 10, 2013

AGENDA

- Agency Background
- Why Automated Mapping?
- Why Python?
- What are we Mapping?
- Overview of the Solution
- Process Details and Challenges
- Final Product
- Other Issues and Decisions
- Project Outcome

AGENCY BACKGROUND

- Who is MCAC?
 - Maryland Coordination and Analysis Center
 - State Fusion Center
 - Creates intelligence products about:
 - Major events
 - Criminal Activity
 - Terrorism
 - Creates intelligence products for:
 - The Governor
 - Federal Partners (FBI and DHS)
 - Local Partners (Municipalities and Counties)
- Who is DoIT GIO?
 - Geographic Information Office for the State of Maryland
 - Supports GIS efforts for various State Agencies:
 - Maryland Emergency Management Agency (MEMA)
 - Maryland State Police (MSP)
 - Department of Natural Resources (DNR)
 - Department of Health and Mental Hygiene (DHMH)

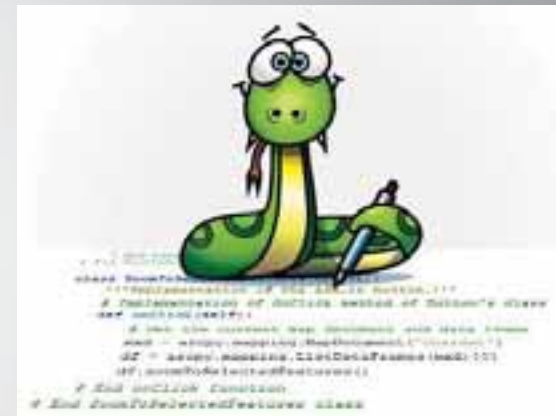


WHY AUTOMATED MAPPING?

- Monthly need for same maps with updated data
- Staff with limited GIS knowledge
- Quick turn-around time
- Allows use of a standard template

WHY PYTHON?

- Suited for different size projects and requirements
- Library/Modular based
 - OS – Operating System access
 - URLLIB – Internet Protocols (HTTP/HTTPS)
 - SMTPLIB – Email
 - Tons of others...
- Open Source with a large user community
- Heavily integrated with Esri/ArcGIS technology
 - ArcPy module



WHAT ARE WE MAPPING?

- Crime in Baltimore City
 - Monthly Data
 - Arrest Locations
 - Various Types of Part I Crimes

OVERVIEW OF THE SOLUTION

- Accept User Inputs
- Geocode Addresses
- Update Underlying Map Data
- Update Embedded Excel Table Data
- Update Map Title and Publication Information
- Export to Distribution Friendly File Formats

PROCESS: ACCEPT USER INPUTS

- Challenges
 - Limit the number of user inputs
 - Re-use inputs throughout script
 - Needs to accept Excel and CSV files
- Details
 - Used `arcpy.GetParameterAsText` to allow for user input
 - Used `arcpy.GetParameter` to allow for predefined list of choices

PROCESS: ACCEPT USER INPUTS

```
##Input Parameters##  
InputTable = arcpy.GetParameterAsText(0)  
MapMonth = arcpy.GetParameter(1)  
PublicationNumber = arcpy.GetParameterAsText(2)
```

A screenshot of a form with three input fields. The first field is labeled 'Input Table' and is empty. The second field is labeled 'Month of Data' and is empty. The third field is labeled 'Publication Number' and is empty. Each field has a green diamond icon to its left.

The 'For Presentation Properties' dialog box is shown with the 'Parameters' tab selected. It contains a table with the following data:

Display Name	Data Type
Input Table	Table
Month of Data	String
Publication Number	String

Below the table is a 'Parameter Properties' section with the following data:

Property	Value
Type	Required
Direction	Input
MultiValue	No
Default	
Environment	
Filter	Value List
Obtained from	

A 'Value List' dialog box is open over the 'Month of Data' parameter, showing a list of values: January, February, March, April, May, June, July, August, and September. The 'List of Values' tab is selected, and the values are listed in a scrollable area. The 'OK' button is highlighted.

PROCESS: GEOCODE ADDRESSES

- Challenges
 - Is there existing connection to ArcGIS Server Geocode Service?
 - What do we do with Tied and Unmatched addresses?
 - What if an address geocodes outside the city?
- Details
 - Used Maryland Cascading Geocoder from MD iMap
 - Used os.env to get Windows username
 - Used os.walk to jump through Windows directories
 - Compared Baltimore ZIP Codes to ZIP Codes of geocoded records for Tied results

PROCESS: GEOCODE ADDRESSES

```
##Variables for Address Locator Check##
MDiMapConnection = "ARCGIS on mdimap.us (user)"
AGSConnection = []
connections = ['.ags']
GrabUsername = os.getenv('USERNAME')
AddressLocator = "C:\\Users\\" + GrabUsername + "\\AppData\\Roaming\\ESRI\\Desktop10.1\\ArcCatalog\\" + MDiMapConnection +

##Address Locator Check##
for root, dirnames, filenames in os.walk("C:\\Users\\" + GrabUsername + "\\AppData\\Roaming\\ESRI\\Desktop10.1\\ArcCatalog"):
    for extensions in connections:
        for filename in fnmatch.filter(filenames, extensions):
            AGSConnection.append(filename)
            arcpy.AddMessage(AGSConnection)
#If iMap connection exists, continue, if not, copy the connection file to the default connection location
if MDiMapConnection in AGSConnection:
    arcpy.AddMessage("Found it!")
else:
    arcpy.AddMessage(":( --- Don't worry, I'll create the connection!")
    copyfile("S:\\GPTools\\Geocode\\arcgis on mdimap.us (user).ags", "C:\\Users\\" + GrabUsername + "\\AppData\\Roaming\\ES
```

PROCESS: GEOCODE ADDRESSES

```
BaltimoreZips = [21201, 21202, 21205, 21206, 21207, 21208, 21209, 21210, 21211,
                21234, 21236, 21237, 21239, 21251, 21287, 21203, 21233, 21263, 21264, 21265,
```

```
##Geocode Addresses##
```

```
#Geocode
```

```
arcpy.GeocodeAddresses_geocoding(IncidentGDB + "\\\" + OutputTable, AddressLocator, "Street ARREST_LOCATION;City City;ZIP <None>", IncidentGDB + "\\\" + OutputGeocode)
```

```
#Make Feature Layer for selection
```

```
arcpy.MakeFeatureLayer_management(IncidentGDB + "\\\" + OutputGeocode, RematchLayer)
```

```
#Create Search Cursor with Where clause for Tied geocoded records
```

```
ZipCheckCursor = arcpy.SearchCursor(IncidentGDB + "\\\" + OutputGeocode, """Match_addr LIKE '%,%' AND "Status" = 'I'""", "", "ObjectID;Match_addr;Status", "")
```

```
#Loop through records in cursor to check if Tied records are actually in a Baltimore Zipcode
```

```
for row in ZipCheckCursor:
```

```
    Match_addrZip = int(row.getValue("Match_addr")[-5:])
```

```
    arcpy.AddMessage(str(Match_addrZip))
```

```
    #If Tied record is in Baltimore, set Status to Match (Tied status records have been geocoded already)
```

```
    if Match_addrZip in BaltimoreZips:
```

```
        OID = str(row.getValue("ObjectID"))
```

```
        arcpy.AddMessage("Match Found!")
```

```
        arcpy.SelectLayerByAttribute_management(RematchLayer, "NEW_SELECTION", "ObjectID"='"+OID)
```

```
        arcpy.CalculateField_management(RematchLayer, "Status", 'M', "VB")
```

```
    #If Tied record is not in Baltimore, set Status to Unmatched (Does not remove record from being geocoded)
```

```
    else:
```

```
        OID = str(row.getValue("ObjectID"))
```

```
        arcpy.AddMessage("No match found :(")
```

```
        arcpy.SelectLayerByAttribute_management(RematchLayer, "NEW_SELECTION", "ObjectID"='"+OID)
```

```
        arcpy.CalculateField_management(RematchLayer, "Status", 'U', "VB")
```

PROCESS: UPDATE UNDERLYING MAP DATA

- Challenges
 - Data backup in case of data append error
 - Updating definition query for the new month
- Details
 - Create copy of master geocoded dataset to a backup GDB
 - Append monthly geocoded addresses to master dataset
 - Access definition query through arcpy.mapping module

PROCESS: UPDATE UNDERLYING MAP DATA

```
#Conditional Statements for Start Date and End Date based on Input Month
if MapMonth == "January":
    Start_Date = "2013-01-01"
    End_Date = "2013-01-31"
elif MapMonth == "February":
    Start_Date = "2013-02-01"
    End_Date = "2013-02-28"
```

```
##Update Master Dataset##
#Get Current Date
today = datetime.date.today()
FormatDate = str(today.year) + "_" + str(today.month) + "_" + str(today.day)
#Make Copy of Feature Class before Appending new data
arcpy.Copy_management(IncidentGDB + "\\ " + MasterGeocode, BackupGDB + "\\ " + "Master_Geocoding_Backup_" + FormatDate)
#Append Geocoded records to Master dataset
arcpy.Append_management(IncidentGDB + "\\ " + OutputGeocode, IncidentGDB + "\\ " + MasterGeocode, "NO_TEST")
```

```
##Update Definition Query##
#Loop through MXD and List Layers
for lyr in arcpy.mapping.ListLayers(mxd, ""):
    #If layer name equals Gang Member Affiliation
    if lyr.name == "Crime Type":
        #Set Definition Query on layer for Arrest Date field with Start Date/Time and End Date/Time
        lyr.definitionQuery = "" "ARREST_DATE" >= date '"" + Start_Date + " " + Start_Time + ""' AND "ARREST_DATE" <= date '"" + End_Date + " " + End_Time + ""'
```

PROCESS: UPDATE EMBEDDED EXCEL TABLE DATA

- Challenges
 - Pulling values from ArcMap table into specific cells in Excel
 - Formatting the table to be suitable for use in the map
- Details
 - Calculate values from different types of mapped and unmapped incidents into ArcMap table view
 - Used xlrd and xlwt Python modules to access Excel
 - Excel table dynamically linked to ArcMap

PROCESS: UPDATE EMBEDDED EXCEL TABLE DATA

```
##Perform Selections##
#Crime A AND Mapped
arcpy.SelectLayerByAttribute_management("geocode_lyr", "NEW_SELECTION", """ABSESS_DATE <= date '"" + Start_Date + " " + Start_Time + """" AND ABSESS_DATE <= date '"" + End_Date + " " + End_Time + """"
arcpy.SelectLayerByAttribute_management("geocode_lyr", "SUBSET_SELECTION", """(!CrimeType) = 'A' AND (!Status) = 'M'")
CrimeA_Mapped = (arcpy.GetCount_management("geocode_lyr").getOutput(0))
arcpy.SelectLayerByAttribute_management("mapchart_view", "NEW_SELECTION", """(!CrimeType) = 'A'")
arcpy.CalculateField_management("mapchart_view", "Current_Month_Mapped_Incidents", CrimeA_Mapped)
#Crime A AND Unmapped
arcpy.SelectLayerByAttribute_management("geocode_lyr", "NEW_SELECTION", """ABSESS_DATE <= date '"" + Start_Date + " " + Start_Time + """" AND ABSESS_DATE <= date '"" + End_Date + " " + End_Time + """"
arcpy.SelectLayerByAttribute_management("geocode_lyr", "SUBSET_SELECTION", """(!CrimeType) = 'A' AND (!Status) = 'U'")
CrimeA_Unmapped = (arcpy.GetCount_management("geocode_lyr").getOutput(0))
arcpy.SelectLayerByAttribute_management("mapchart_view", "NEW_SELECTION", """(!CrimeType) = 'A'")
arcpy.CalculateField_management("mapchart_view", "Current_Month_Unmapped_Incidents", CrimeA_Unmapped)
```


PROCESS: UPDATE EMBEDDED EXCEL TABLE DATA

```
##Write values from ArcMap table to Excel table##
#Open Excel table
ExcelTable = r'U:\Analysis\Crimes\CrimeActivity.xls'
rb = open_workbook(ExcelTable, formatting_info=True)
rs = rb.sheet_by_index(0)
wb = copy(rb)
ws = wb.get_sheet(0)

#Write values to specific cells and format them
ws.write(2,1,"Crime A",easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(2,2,CrimeA_Mapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(2,3,CrimeA_Unmapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(2,4,CrimeA_Total,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(3,1,"Crime B",easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(3,2,CrimeB_Mapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(3,3,CrimeB_Unmapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(3,4,CrimeB_Total,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(4,1,"Crime C",easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(4,2,CrimeC_Mapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(4,3,CrimeC_Unmapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(4,4,CrimeC_Total,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(5,1,"Crime D",easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(5,2,CrimeD_Mapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(5,3,CrimeD_Unmapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(5,4,CrimeD_Total,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(6,1,"Other Crimes",easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(6,2,Other_Mapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
ws.write(6,3,Other_Unmapped,easyxf('alignment: horizontal center, vertical center;' 'borders: left medium, right medium, top medium, bottom medium;' 'font: name Calibri, height 220'))
```

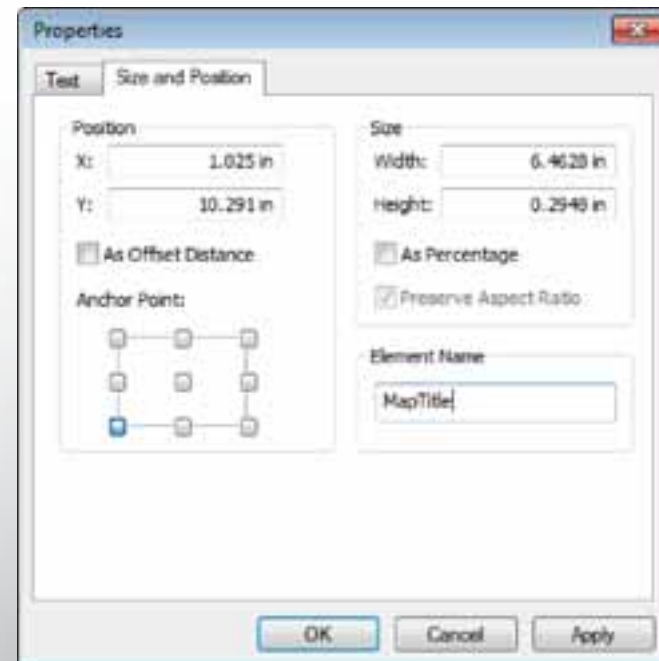
Crime Type	2013 Previous Month(s)	Current Month Mapped Incidents	Current Month Unmapped Incidents	2013 Total to Date
Crime A	387	11	8	406
Crime B	500	32	12	544
Crime C	235	11	7	253
Crime D	91	3	5	99
All other crimes	169	10	8	187
Total	1382	67	40	1489
Total Incidents			107	

PROCESS: UPDATE MAP TITLE AND PUBLICATION INFORMATION

- Challenges
 - Maximizing title size with changing months
- Details
 - Used Layout Elements to define what gets updated
 - Updated through user inputs and dynamic title
 - Month of Map
 - Product Number
 - Dynamic text pulls current date map was created

PROCESS: UPDATE MAP TITLE AND PUBLICATION INFORMATION

```
##Update Dynamic Layout Elements (Map Title and Footer)##  
#Assign Variable to Title  
mxd.title = MapTitle  
#Variables for Font Size and Title Width  
x=100  
elmWidth = 6.617  
#Update Map Title and Footer  
for elm in arcpy.mapping.ListLayoutElements(mxd, "TEXT_ELEMENT"):  
    if elm.name == "footnote":  
        elm.text = "Product Number: " + Product_Number + "\n" + 'Date: <dyn type="date" format="d MMMM YYYY"/>  
    if elm.name == "MapTitle":  
        while elm.elementWidth > float(elmWidth):  
            elm.fontSize = x  
            x = x-1
```







PROCESS: EXPORT TO DISTRIBUTION FRIENDLY FILE FORMATS

- Challenges
 - None!
- Details
 - Saves a new copy of the MXD
 - Used arcpy.mapping module to export to JPEG and PDF

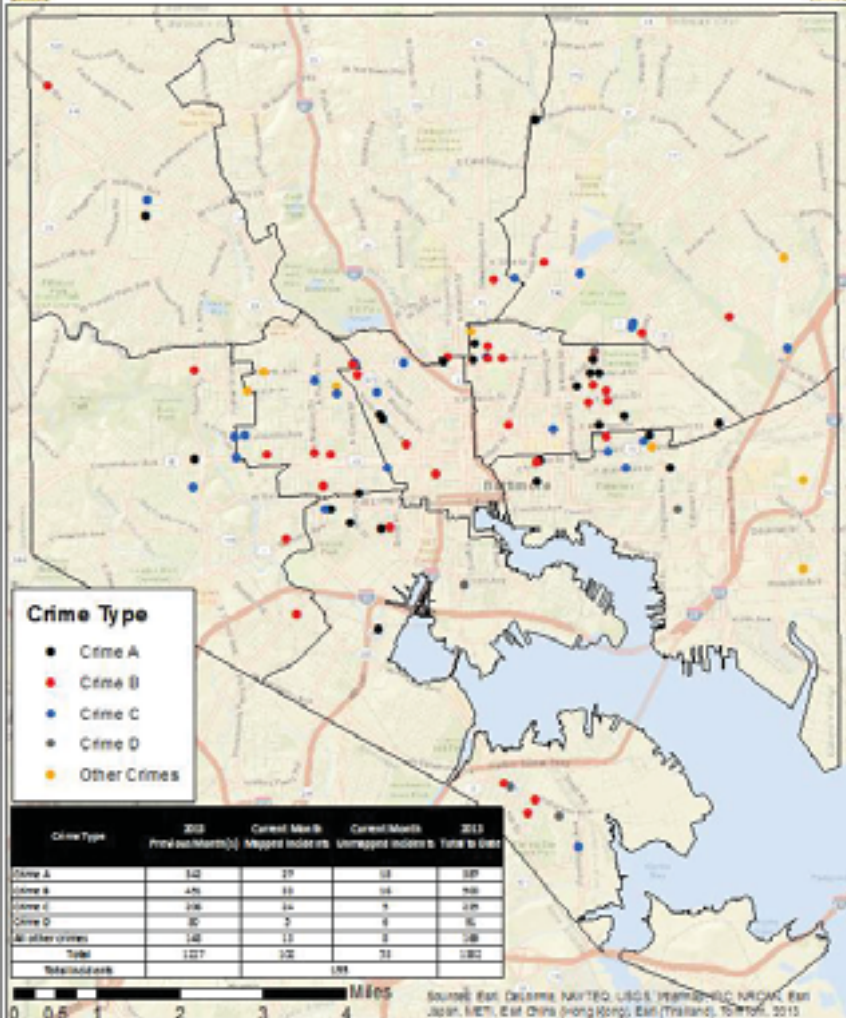
PROCESS: EXPORT TO DISTRIBUTION FRIENDLY FILE FORMATS

```
##Update and Export Map##  
#Refresh Map  
arcpy.RefreshActiveView()  
#Save Copy of MXD  
mxd.saveACopy(mxdOutput)  
#Export Map to PDF  
arcpy.mapping.ExportToPDF(mxd, r"U:\Maps\CrimeType_Locations_" + MapMonth + "2013.pdf")  
#Export Map to JPEG  
arcpy.mapping.ExportToJPEG(mxd, r"U:\Maps\CrimeType Locations " + MapMonth + "2013.jpg")
```

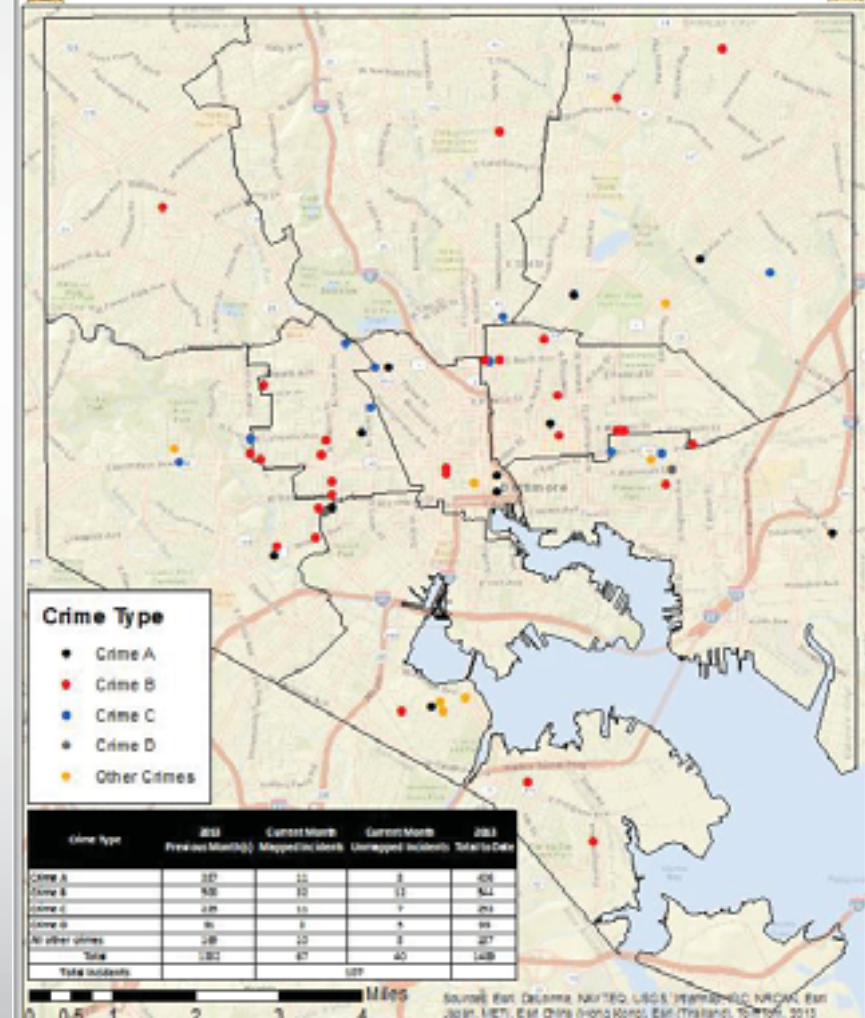
-  CrimeType_Locations_November2013.jpg
-  CrimeType_Locations_November2013.pdf
-  CrimeType_Locations_October2013.jpg
-  CrimeType_Locations_October2013.pdf

FINAL PRODUCT

Arrests by Crime Type - Baltimore City, October 2013



Arrests by Crime Type - Baltimore City, November 2013



OTHER ISSUES AND DECISIONS

- Issues
 - Data
 - Missing Arrest Locations
 - Invalid Addresses
 - Misspelled Crime Types
 - Layout
 - Bug that causes table not to display in ArcMap but appears in export
 - Python
 - Modules not included with Python need to be installed locally
- Decisions
 - No manual rematch of addresses
 - Records with valid geocoded addresses outside of Baltimore City would be removed from dataset
 - Only top 4 crimes would be shown separately, all the rest fall under Other

PROJECT OUTCOME

- Analysts....
 - With little or no GIS experience can generate maps...
 - Can create maps on demand...
 - Can use a standard template for use in reports...



THANK YOU!

Matt Sokol

Senior GIS Analyst

DoIT GIO

matthew.sokol@maryland.gov

Jessica Curtis

Intelligence Analyst

MCAC

jessica.sullivan@mcac.maryland.gov

