Utility Network Migration Using Python

Mark Cederholm
UniSource Energy Services
Flagstaff, Arizona

Esri Developer Summit 2019

It’s never too soon to start!

- Study the data model (UPDM)
- Map your data
- Test your migration
- Learn ArcGIS Pro workflows
- Identify the gaps
- *Report bugs*

Utility Network Resources

- What is a utility network?

http://pro.arcgis.com/en/pro-app/help/data/utility-network/

- Gas Utility Network Configuration:

http://solutions.arcgis.com/gas/help/gas-utility-network-configuration/

- Utility Network Package Tools:

http://solutions.arcgis.com/utilities/help/utility-network-automation/

Pilot Project Goals:

1. Test migration of existing gas data to UPDM and Utility Network
2. Data model setup, network configuration, data loading, and initial cleanup 100% Python scripted

~ Scripting acts as documentation ~

Utility and Pipeline Data Model (2018):

https://community.esri.com/docs/DOC-11209-updm-2018-edition

Software Versions

- ArcGIS Enterprise 10.6.1
  [plus UN patch]
- ArcGIS Pro 2.3.1*
- untools 2.3.1**

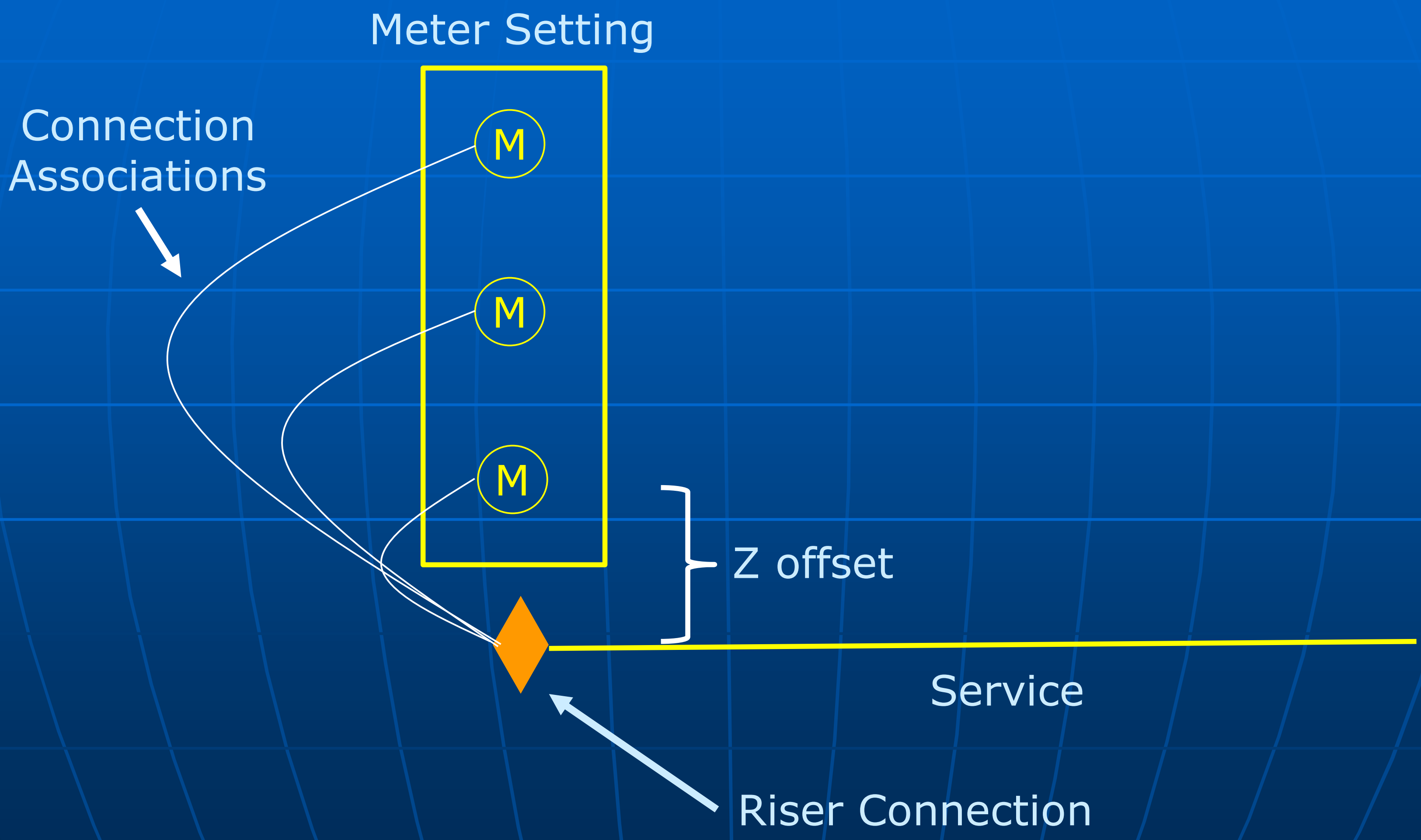
*Does not fix 2.3.0 EnableNetworkTopology bug

**Fixes 2.3.0 AssetPackageToUtilityNetwork bug

Why populate an asset package?

- Easier to modify subnetwork definitions and reload test data
- Apply Asset Package tool has some useful QA checks
- *Performance*

Problem: multiple customer meters



Add connection point asset types:

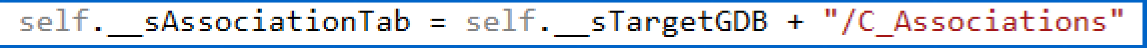
```
sDomainName = "Asset Type Pipeline Junction Connection Point"
arcpy.AddCodedValueToDomain_management(sTargetGDB, sDomainName, 802, "Riser Connection")
arcpy.AddCodedValueToDomain_management(sTargetGDB, sDomainName, 803, "System Connection")
```

Add connectivity rule (B_Rules):

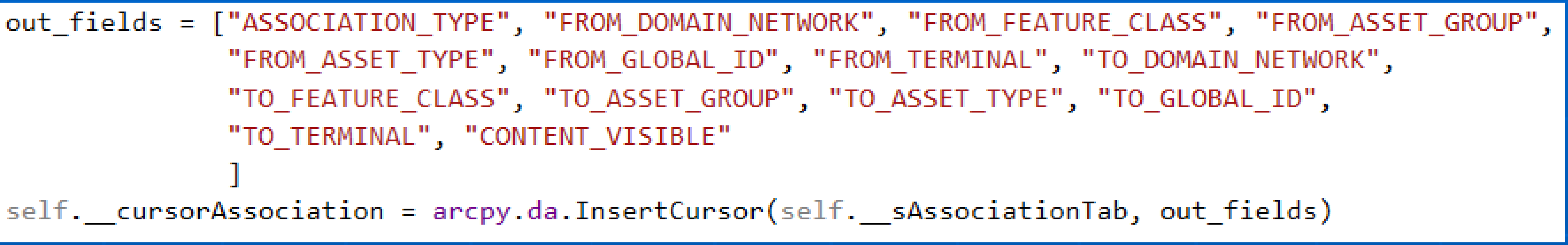
```
Fields = ["rule_type", "from_domain_network", "from_feature_class", "from_asset_group", "from_asset_type",
          "from_terminal", "to_domain_network", "to_feature_class", "to_asset_group", "to_asset_type"]
cursor = arcpy.da.InsertCursor("B_Rules", Fields)
```

```
row = ("Junction Junction Connectivity", 1, "Device", "Meter", "Customer",
       None, 1, "Junction", "Connection Point", "*")
cursor.insertRow(row)
```

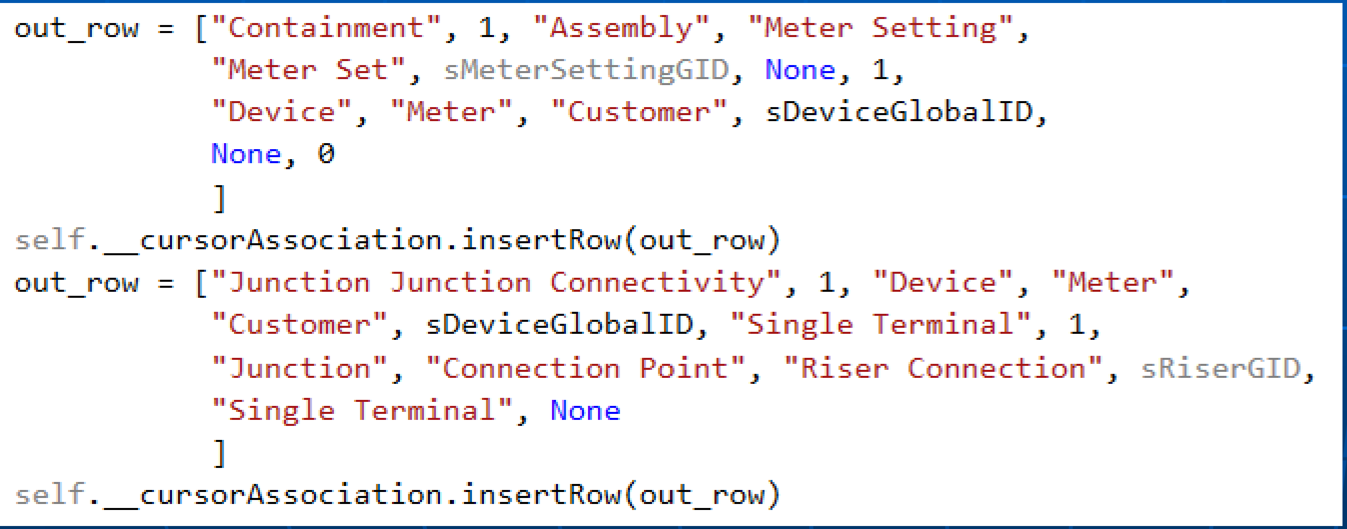
Add association records (C_Associations):



```
self.__sAssociationTab = self.__sTargetGDB + "/C_Associations"
```



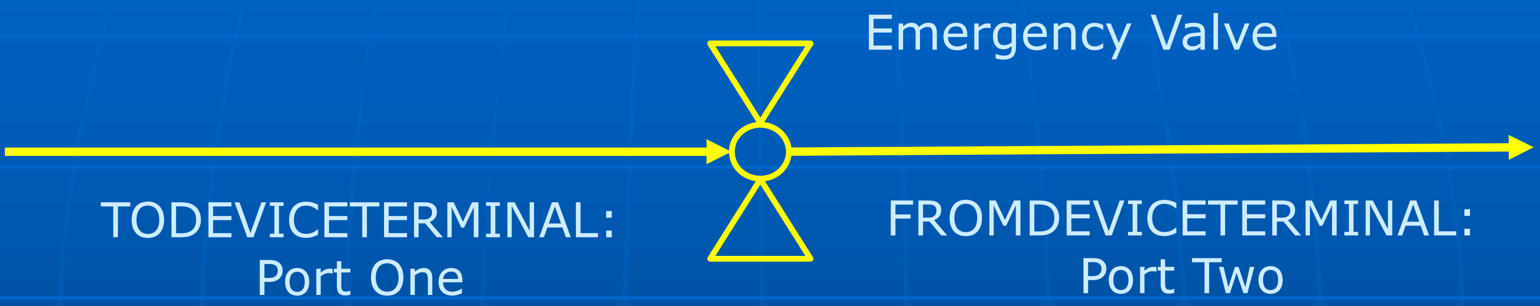
```
out_fields = ["ASSOCIATION_TYPE", "FROM_DOMAIN_NETWORK", "FROM_FEATURE_CLASS", "FROM_ASSET_GROUP",
              "FROM_ASSET_TYPE", "FROM_GLOBAL_ID", "FROM_TERMINAL", "TO_DOMAIN_NETWORK",
              "TO_FEATURE_CLASS", "TO_ASSET_GROUP", "TO_ASSET_TYPE", "TO_GLOBAL_ID",
              "TO_TERMINAL", "CONTENT_VISIBLE"
              ]
self.__cursorAssociation = arcpy.da.InsertCursor(self.__sAssociationTab, out_fields)
```



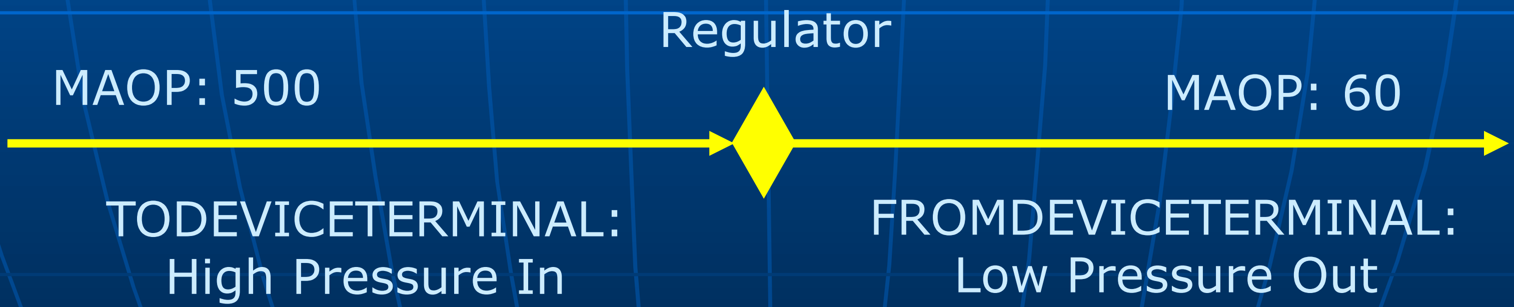
```
out_row = ["Containment", 1, "Assembly", "Meter Setting",
           "Meter Set", sMeterSettingGID, None, 1,
           "Device", "Meter", "Customer", sDeviceGlobalID,
           None, 0
           ]
self.__cursorAssociation.insertRow(out_row)
out_row = ["Junction Junction Connectivity", 1, "Device", "Meter",
           "Customer", sDeviceGlobalID, "Single Terminal", 1,
           "Junction", "Connection Point", "Riser Connection", sRiserGID,
           "Single Terminal", None
           ]
self.__cursorAssociation.insertRow(out_row)
```

Problem: terminal assignment

Devices with terminal configurations must have terminals assigned to connecting lines:



Directional terminals must be assigned appropriately:

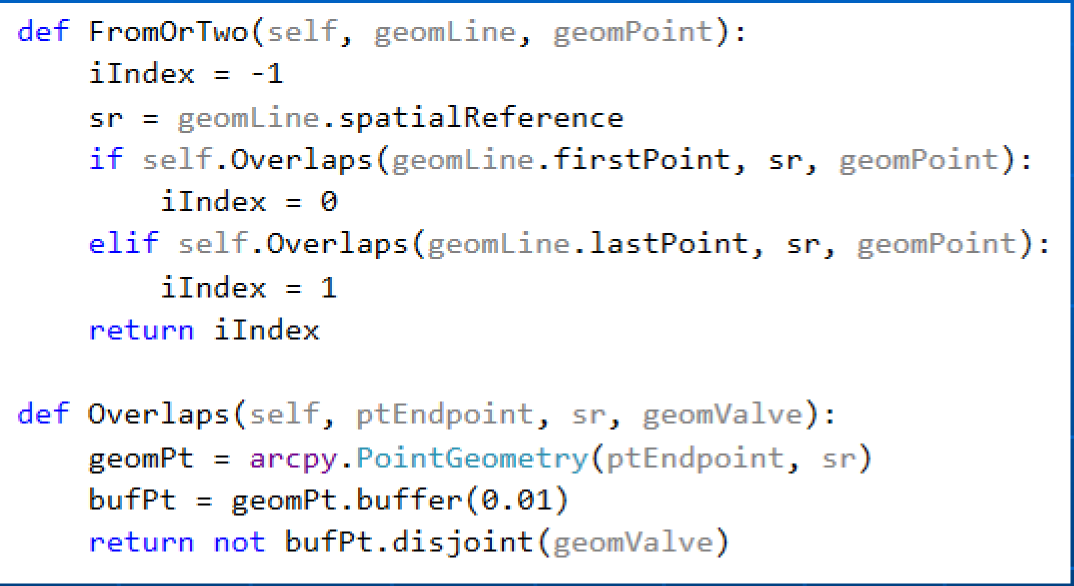

Regulator
MAOP: 500
MAOP: 60
TODEVICETERMINAL: High Pressure In
FROMDEVICETERMINAL: Low Pressure Out

1:44:40 PM

Terminal Assignment

- SpatialJoin_analysis to get connected line attributes
- Determine terminals to be assigned to connected lines
- For each connected line, check if device on FROM or TO endpoint

arcpy Geometry.overlaps method doesn’t work:



```python
def FromOrTwo(self, geomLine, geomPoint):
    iIndex = -1
    sr = geomLine.spatialReference
    if self.Overlaps(geomLine.firstPoint, sr, geomPoint):
        iIndex = 0
    elif self.Overlaps(geomLine.lastPoint, sr, geomPoint):
        iIndex = 1
    return iIndex

def Overlaps(self, ptEndpoint, sr, geomValve):
    geomPt = arcpy.PointGeometry(ptEndpoint, sr)
    bufPt = geomPt.buffer(0.01)
    return not bufPt.disjoint(geomValve)
```

Why subnetworks?

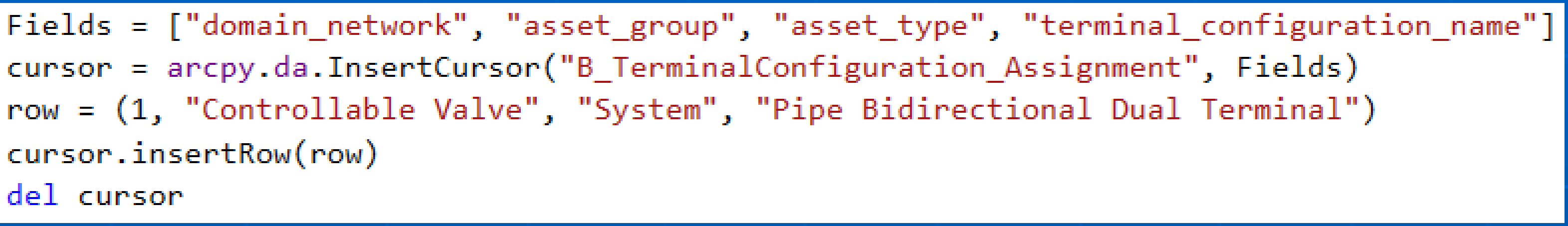
- Populate summary attributes on subnet lines
- Populate attributes on features
- Subnet lines do not need to be generated for all tiers

Each subnetwork must start with a device acting as a controller

Tier controllers

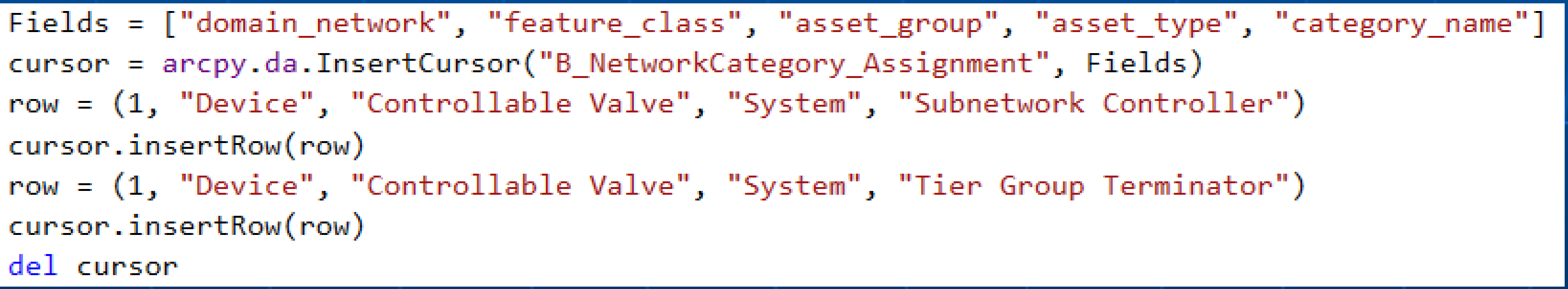
Tier	UPDM	Pilot (initial)
Transmission System	Custody Transfer Meter	System Valve
Transmission Pressure	Compressor, Pump, Flow Valve	Regulator
Distribution System	Custody Transfer Meter	System Valve
Distribution Pressure	Regulator	Regulator
Distribution Isolation	Emergency Valve	Emergency Valve

Add terminal configuration (B_TerminalConfiguration_Assignment):



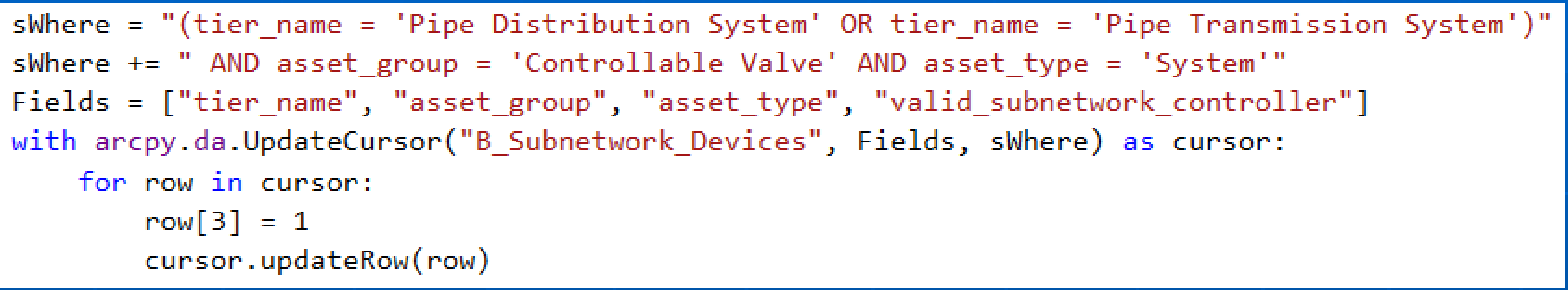
```
Fields = ["domain_network", "asset_group", "asset_type", "terminal_configuration_name"]
cursor = arcpy.da.InsertCursor("B_TerminalConfiguration_Assignment", Fields)
row = (1, "Controllable Valve", "System", "Pipe Bidirectional Dual Terminal")
cursor.insertRow(row)
del cursor
```

Add network categories (B_NetworkCategory_Assignment):

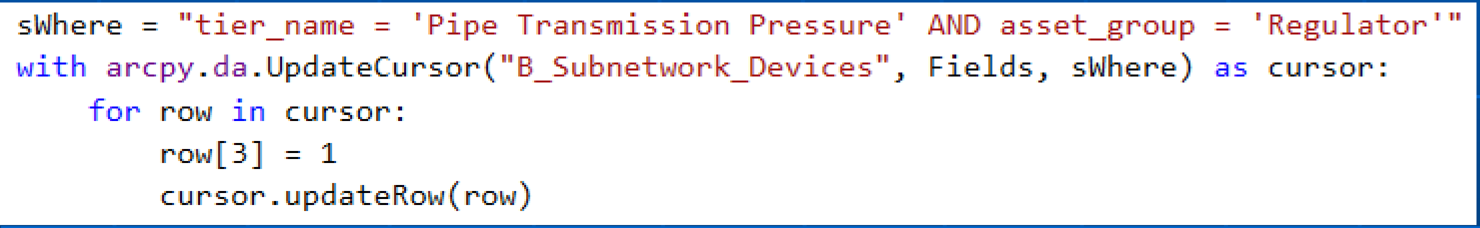


```
Fields = ["domain_network", "feature_class", "asset_group", "asset_type", "category_name"]
cursor = arcpy.da.InsertCursor("B_NetworkCategory_Assignment", Fields)
row = (1, "Device", "Controllable Valve", "System", "Subnetwork Controller")
cursor.insertRow(row)
row = (1, "Device", "Controllable Valve", "System", "Tier Group Terminator")
cursor.insertRow(row)
del cursor
```

Update subnetwork controllers (B_Subnetwork_Devices):

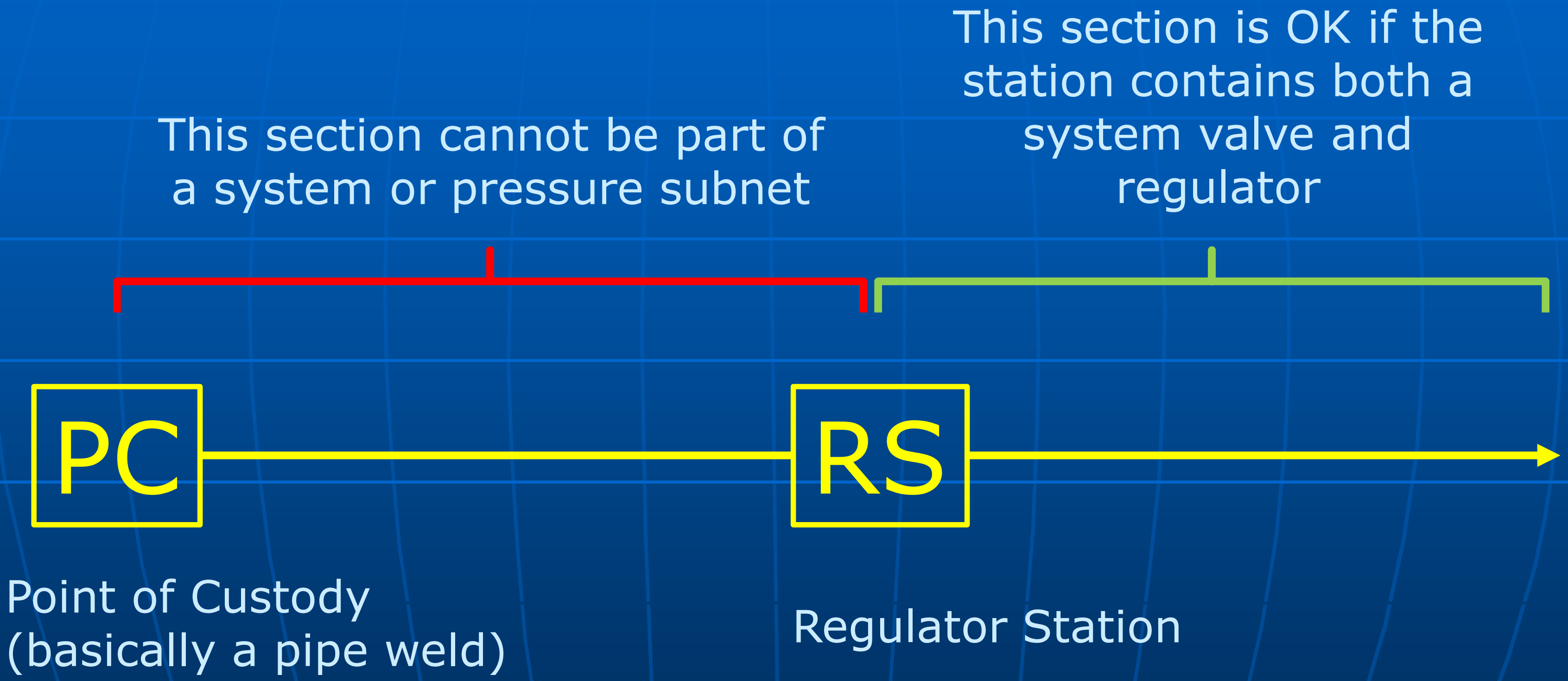


```
sWhere = "(tier_name = 'Pipe Distribution System' OR tier_name = 'Pipe Transmission System')"
sWhere += " AND asset_group = 'Controllable Valve' AND asset_type = 'System'"
Fields = ["tier_name", "asset_group", "asset_type", "valid_subnetwork_controller"]
with arcpy.da.UpdateCursor("B_Subnetwork_Devices", Fields, sWhere) as cursor:
    for row in cursor:
        row[3] = 1
        cursor.updateRow(row)
```



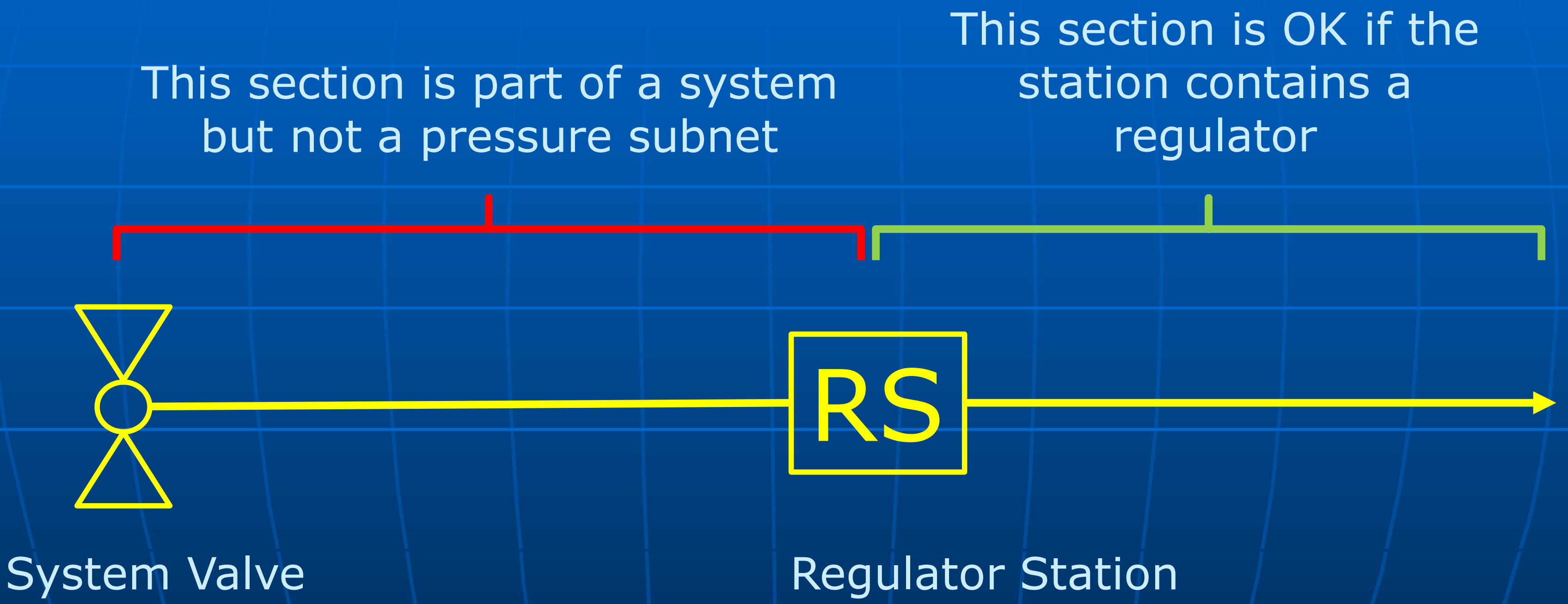
```
sWhere = "tier_name = 'Pipe Transmission Pressure' AND asset_group = 'Regulator'"
with arcpy.da.UpdateCursor("B_Subnetwork_Devices", Fields, sWhere) as cursor:
    for row in cursor:
        row[3] = 1
        cursor.updateRow(row)
```

Problem: most of our systems do not start with devices!


This section cannot be part of a system or pressure subnet
This section is OK if the station contains both a system valve and regulator
PC
RS
Point of Custody (basically a pipe weld)
Regulator Station

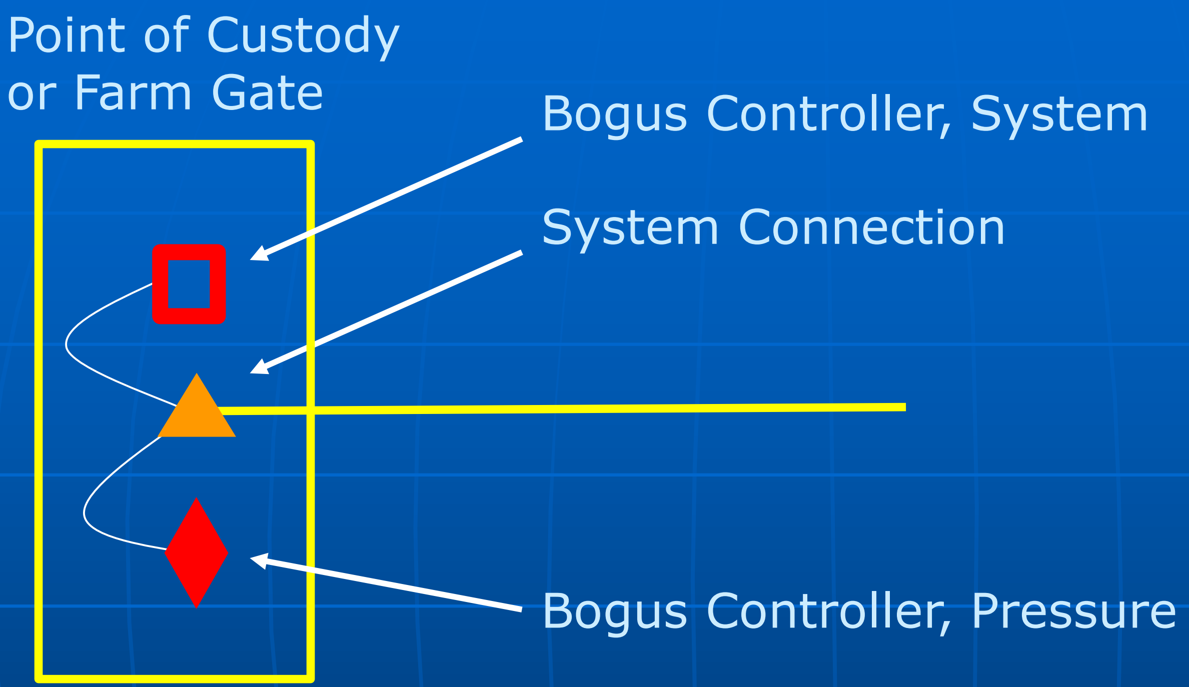
1:44:40 PM

Two of our systems (and more in the future) start with a system valve

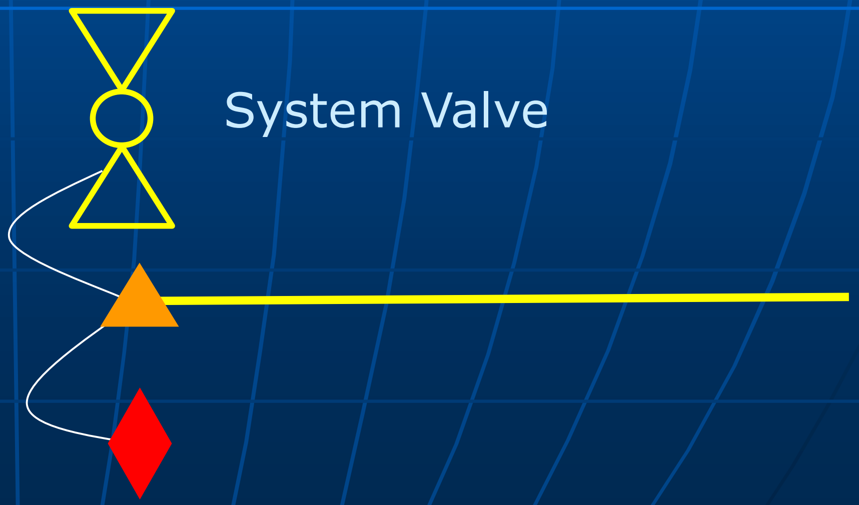

This section is part of a system but not a pressure subnet
This section is OK if the station contains a regulator
RS
System Valve
Regulator Station

1:44:40 PM

New device asset group: “Bogus Controller”

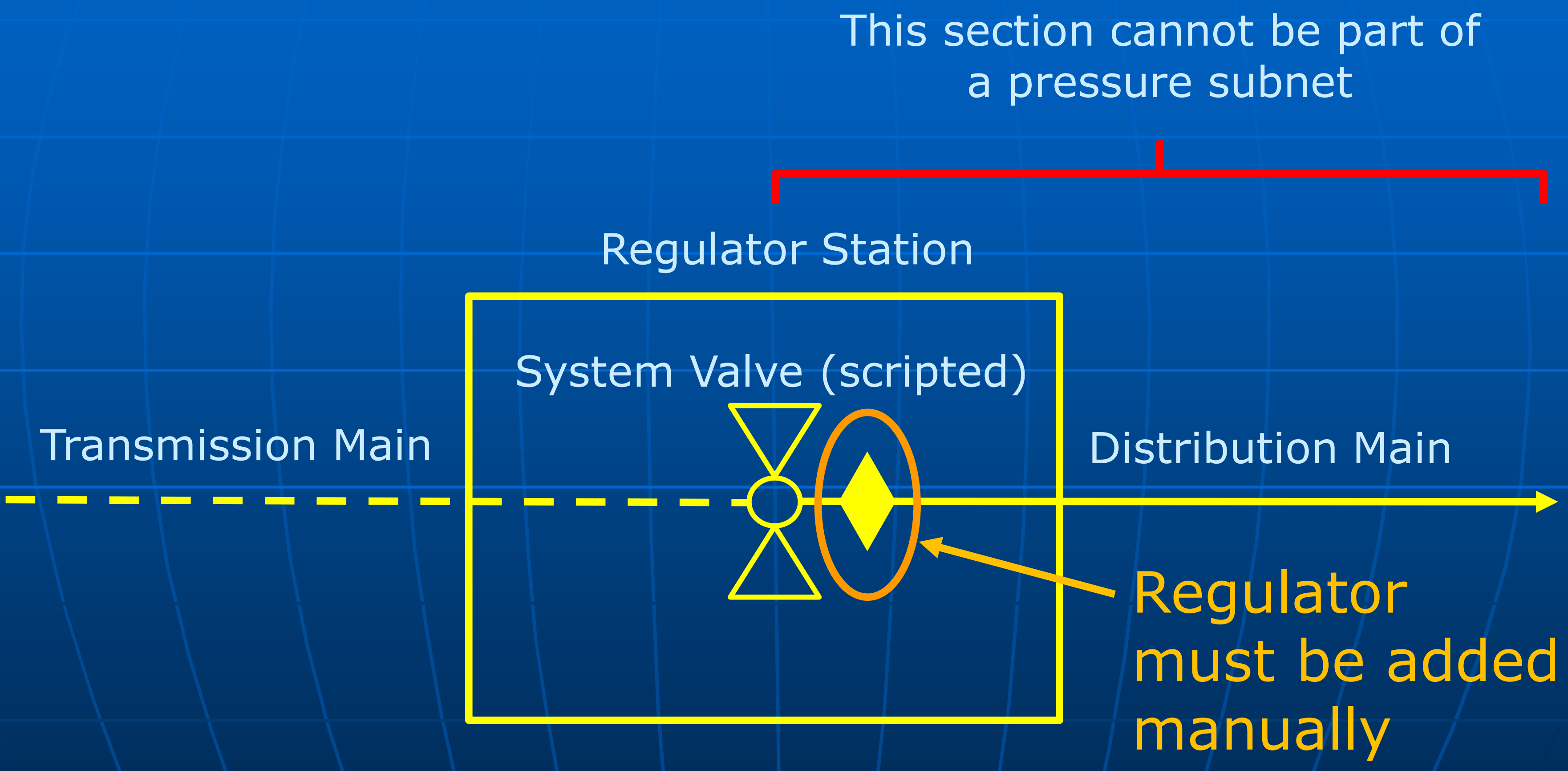


Where a valve exists:



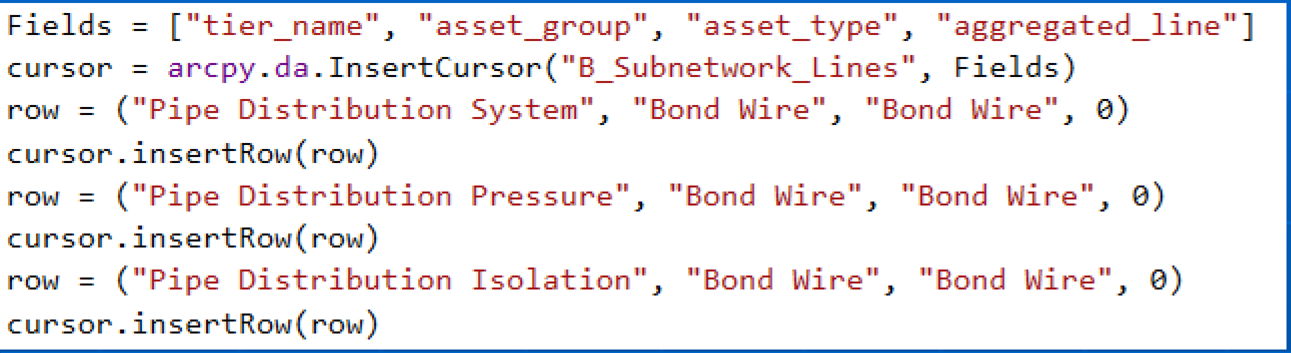
1:44:40 PM

Problem: regulator stations between transmission and distribution systems


This section cannot be part of a pressure subnet
Regulator Station
System Valve (scripted)
Transmission Main
Distribution Main
Regulator must be added manually

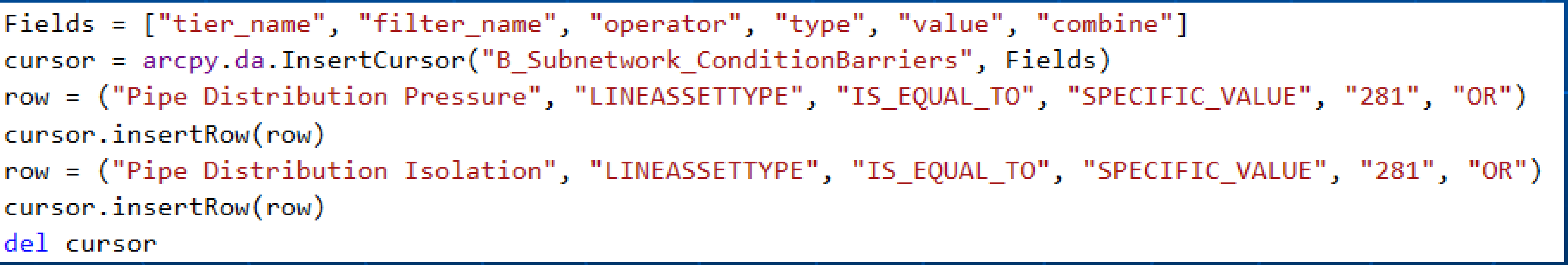
Problem: CP bond wire

Add to subnetworks (B_Subnetwork_Lines):



```
Fields = ["tier_name", "asset_group", "asset_type", "aggregated_line"]
cursor = arcpy.da.InsertCursor("B_Subnetwork_Lines", Fields)
row = ("Pipe Distribution System", "Bond Wire", "Bond Wire", 0)
cursor.insertRow(row)
row = ("Pipe Distribution Pressure", "Bond Wire", "Bond Wire", 0)
cursor.insertRow(row)
row = ("Pipe Distribution Isolation", "Bond Wire", "Bond Wire", 0)
cursor.insertRow(row)
```

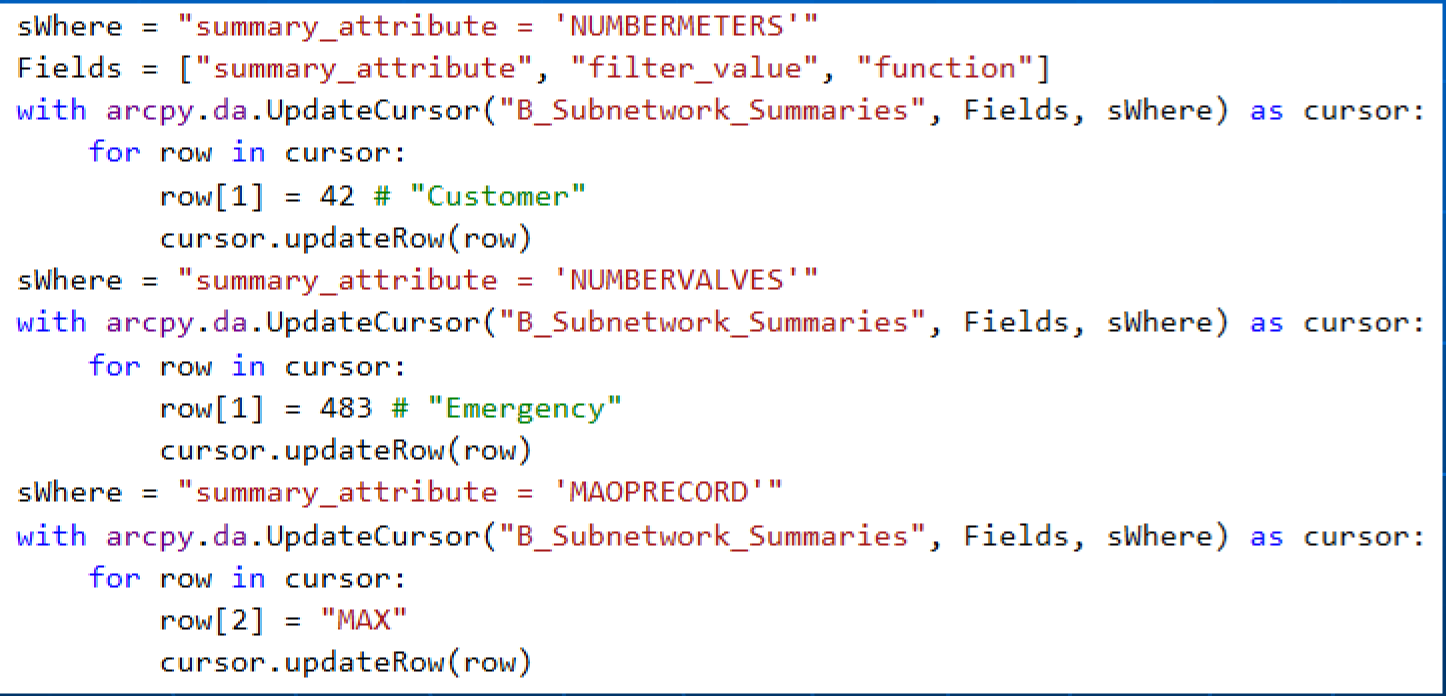
Add condition barriers (B_Subnetwork_ConditionBarriers):



```
Fields = ["tier_name", "filter_name", "operator", "type", "value", "combine"]
cursor = arcpy.da.InsertCursor("B_Subnetwork_ConditionBarriers", Fields)
row = ("Pipe Distribution Pressure", "LINEASSETTYPE", "IS_EQUAL_TO", "SPECIFIC_VALUE", "281", "OR")
cursor.insertRow(row)
row = ("Pipe Distribution Isolation", "LINEASSETTYPE", "IS_EQUAL_TO", "SPECIFIC_VALUE", "281", "OR")
cursor.insertRow(row)
del cursor
```

Problem: out-of-box summaries not working

Update subnetwork summaries (B_Subnetwork_Summaries):



```
sWhere = "summary_attribute = 'NUMBERMETERS'"
Fields = ["summary_attribute", "filter_value", "function"]
with arcpy.da.UpdateCursor("B_Subnetwork_Summaries", Fields, sWhere) as cursor:
    for row in cursor:
        row[1] = 42 # "Customer"
        cursor.updateRow(row)
sWhere = "summary_attribute = 'NUMBERVALVES'"
with arcpy.da.UpdateCursor("B_Subnetwork_Summaries", Fields, sWhere) as cursor:
    for row in cursor:
        row[1] = 483 # "Emergency"
        cursor.updateRow(row)
sWhere = "summary_attribute = 'MAOPRECORD'"
with arcpy.da.UpdateCursor("B_Subnetwork_Summaries", Fields, sWhere) as cursor:
    for row in cursor:
        row[2] = "MAX"
        cursor.updateRow(row)
```

Loading Data

- AssetPackageToUtilityNetwork enables Editor Tracking
- Workaround:
  1) DisableEditorTracking
  2) arcpy.env.preserveGlobalIds = True
  3) Append
  4) ImportSubnetworkControllers
  5) ImportAssociations

Initial Cleanup: Topology Errors

- Error 25: Stacked point features

  [Fixed using Python]
- Error 21: Duplicate vertices

  [Fixed using Python]
- Error 9: Flow valve terminals

  [Must be fixed manually]

Total Migrated Line Features

- Connected: 248319
- Not connected: 2074

Lessons Learned

- Break large tasks into smaller ones
- Reset the Python environment often
- Expect crashes
- RepairGeometry isn't good enough
- Manual cleanup <u>will</u> be needed

To Do:

- Snap unconnected features
- CP features and tier group
- ArcGIS Monitor

Future Considerations:

- Design & work management tools
- Synchronization tools?
- Data Reviewer
- Migrate mapping & reporting apps
- Off-line support [ArcGIS Runtime]

Questions?

- Mark Cederholm
  mcederholm@uesaz.com
- This presentation and sample code may be downloaded at:

http://www.pierssen.com/arcgis/python.htm