Set and Forget

Automated Service Authoring

Insightful solutions.
Empowering advice.

NCTIR - North Canterbury Transport Infrastructure Recovery

abley

The Plan

- Overview
- Data Movement
- Service Creation
- Scenario 1 – Design
- Scenario 2 – UAV Imagery

abley

Questions for you

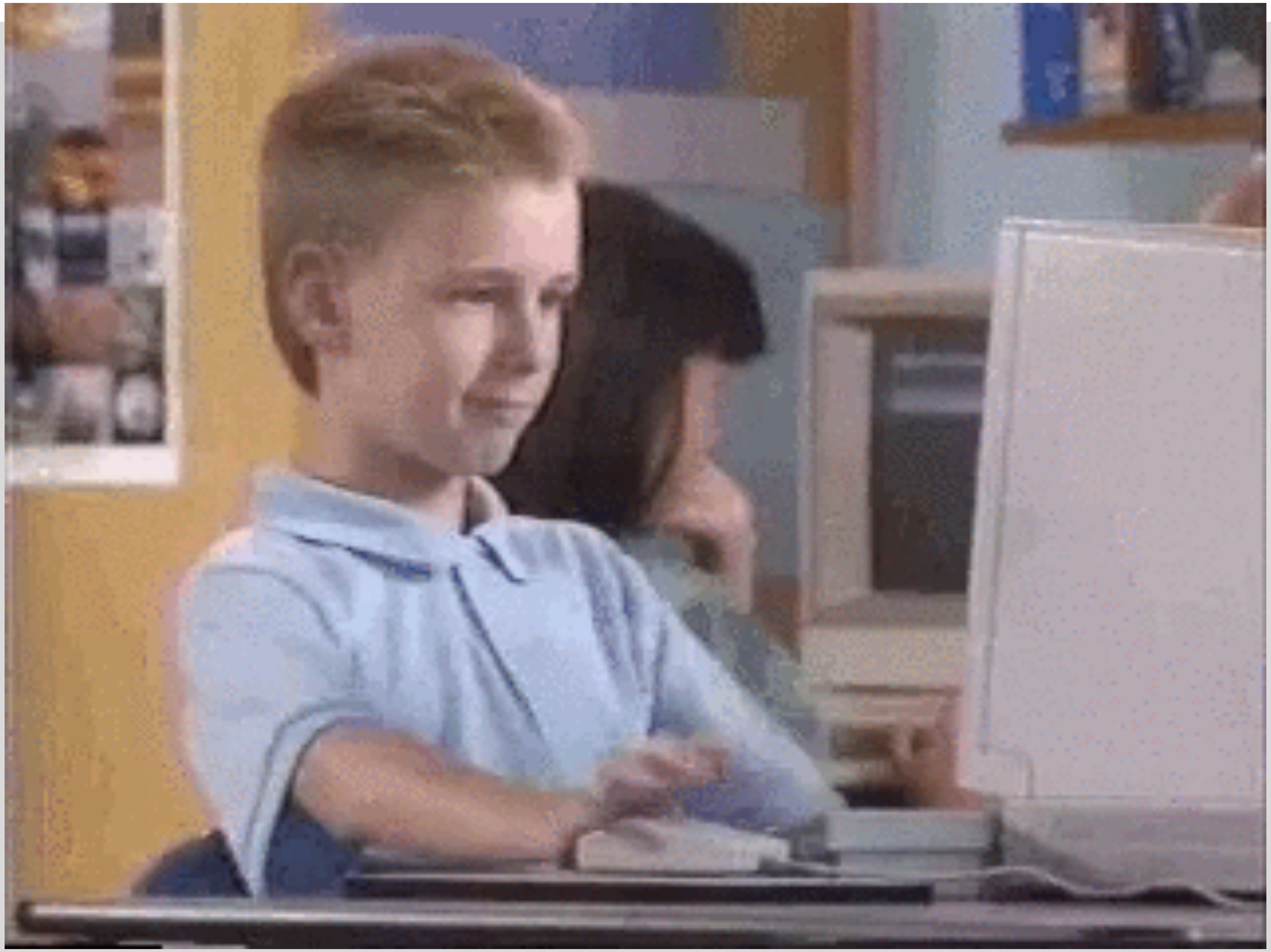
- How many of you have used ArcGIS Enterprise (Server)?
- Are you familiar with imagery, caching and ImageServer?
- Do you have workflows that would benefit from automated service publishing?
- Have you edited .sddrafts?

Technologies Used

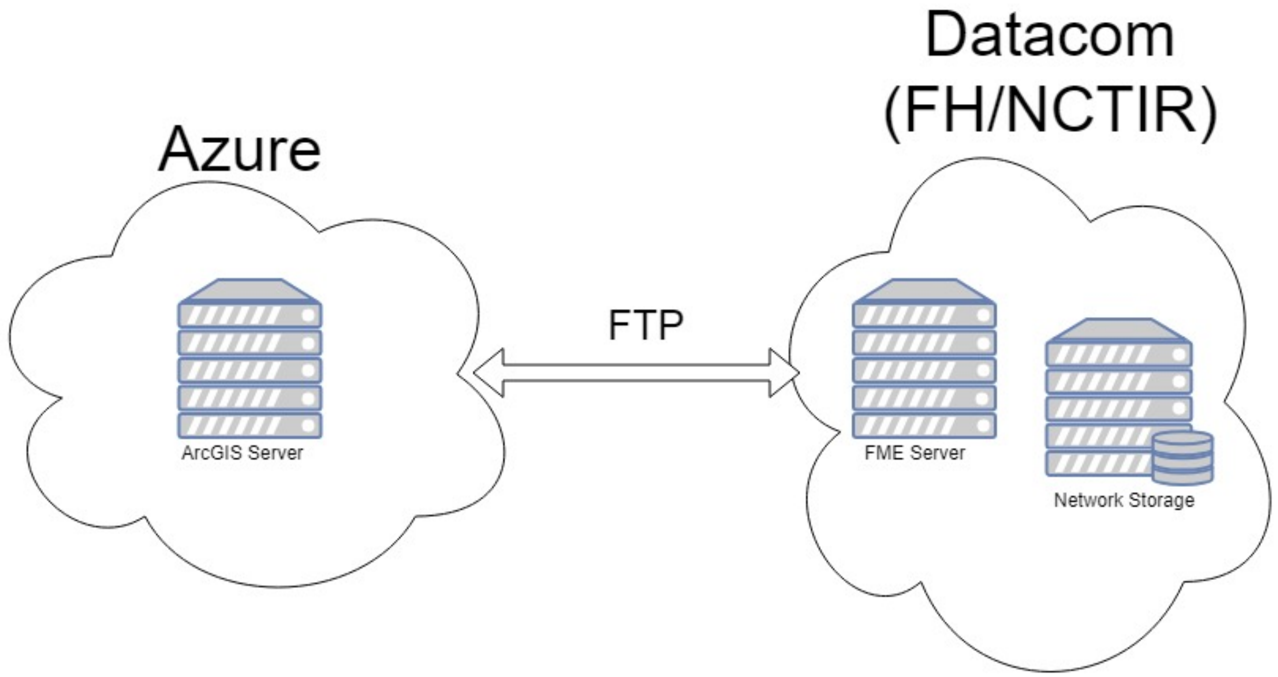
- ArcGIS Enterprise (Server)
- FME
- Python (Arcpy)
- XML

At first I was like

Then I was like



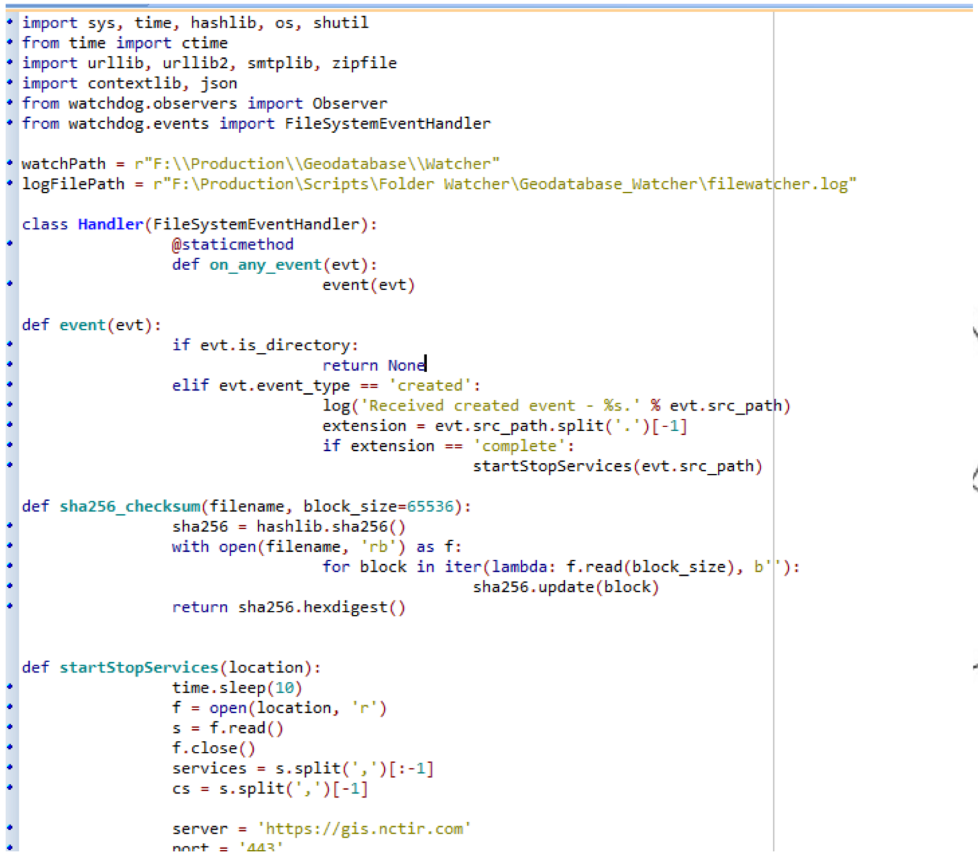
Problem(s) Overview



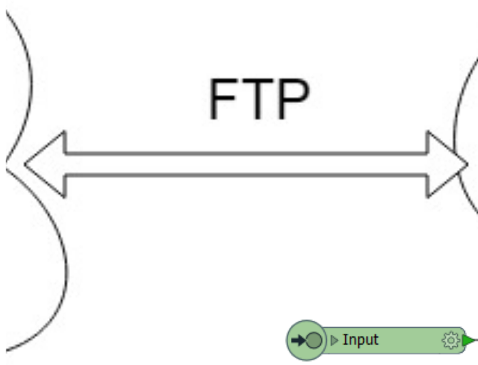
- Needed automation
- Irregular data deliveries
- Different infrastructure locations
- Processes are silo-ed

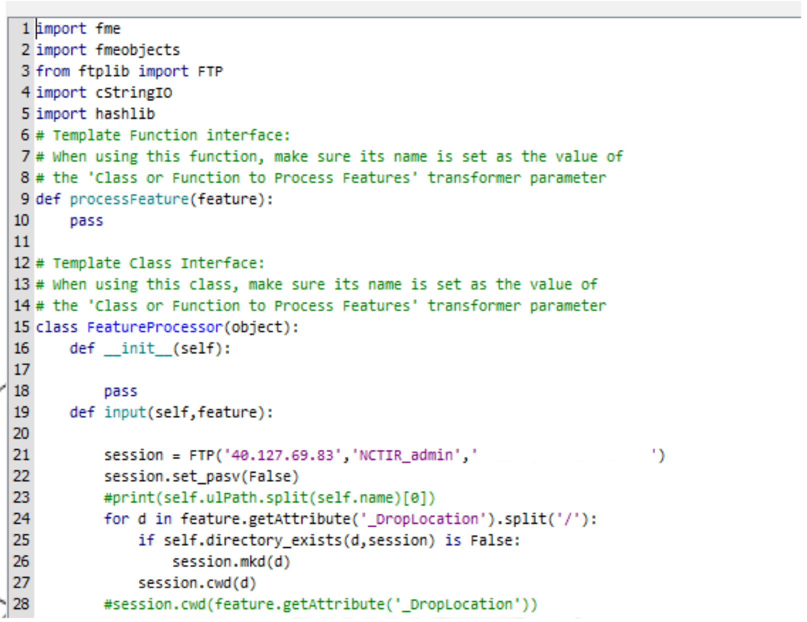
abley

Data Transfer

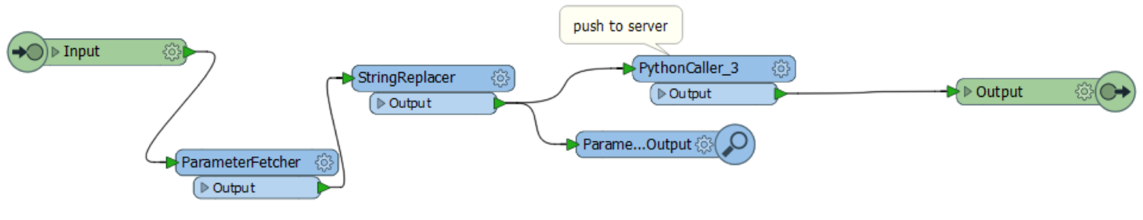


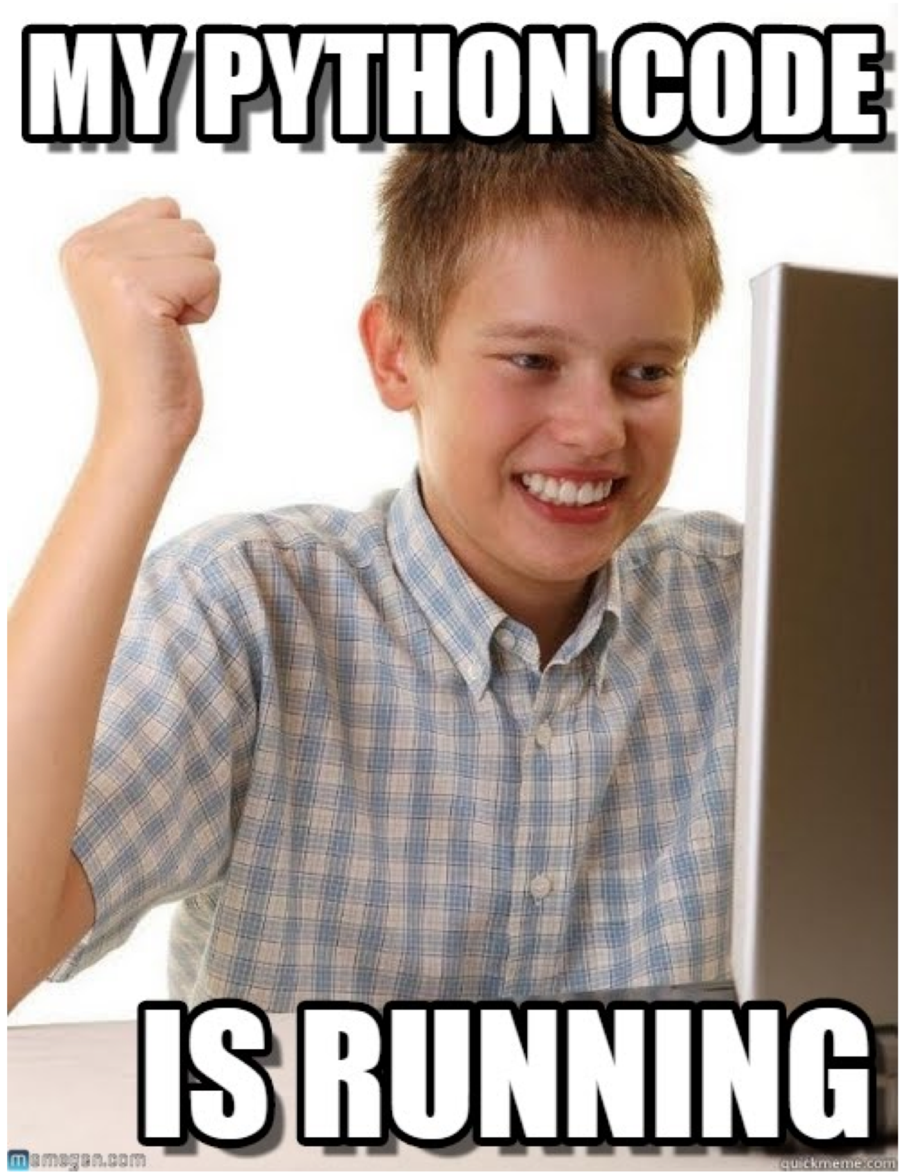
```
import sys, time, hashlib, os, shutil
from time import ctime
import urllib, urllib2, smtplib, zipfile
import contextlib, json
from watchdog.observers import Observer
from watchdog.events import FileSystemEventHandler

watchPath = r"F:\\Production\\Geodatabase\\Watcher"
logFilePath = r"F:\Production\Scripts\Folder Watcher\Geodatabase_Watcher\filewatcher.log"

class Handler(FileSystemEventHandler):
                @staticmethod
                def on_any_event(evt):
                                event(evt)

def event(evt):
                if evt.is_directory:
                                return None
                elif evt.event_type == 'created':
                                log('Received created event - %s.' % evt.src_path)
                                extension = evt.src_path.split('.')[-1]
                                if extension == 'complete':
                                                startStopServices(evt.src_path)

def sha256_checksum(filename, block_size=65536):
                sha256 = hashlib.sha256()
                with open(filename, 'rb') as f:
                                for block in iter(lambda: f.read(block_size), b''):
                                                sha256.update(block)
                return sha256.hexdigest()


def startStopServices(location):
                time.sleep(10)
                f = open(location, 'r')
                s = f.read()
                f.close()
                services = s.split(',')[:-1]
                cs = s.split(',')[-1]

                server = 'https://gis.nctir.com'
                port = '443'
```





```
import fme
import fmeobjects
from ftplib import FTP
import cStringIO
import hashlib
# Template Function interface:
# When using this function, make sure its name is set as the value of
# the 'Class or Function to Process Features' transformer parameter
def processFeature(feature):
    pass

# Template Class Interface:
# When using this class, make sure its name is set as the value of
# the 'Class or Function to Process Features' transformer parameter
class FeatureProcessor(object):
    def __init__(self):

        pass
    def input(self,feature):

        session = FTP('40.127.69.83','NCTIR_admin','[redacted]')
        session.set_pasv(False)
        #print(self.ulPath.split(self.name)[0])
        for d in feature.getAttribute('_DropLocation').split('/'):
            if self.directory_exists(d,session) is False:
                session.mkd(d)
            session.cwd(d)
        #session.cwd(feature.getAttribute('_DropLocation'))
```



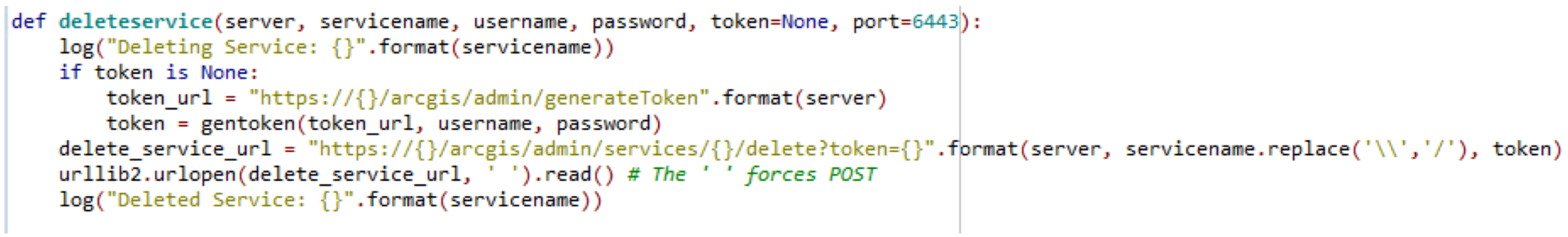

MY PYTHON CODE
IS RUNNING
memegen.com
quickmeme.com

Service Creation

- Stop and Delete old service
- Create service definition draft (.sddraft)
- Analyse .sddraft
- Create service definition (.sd)
- Use .sd to publish service

Service Creation

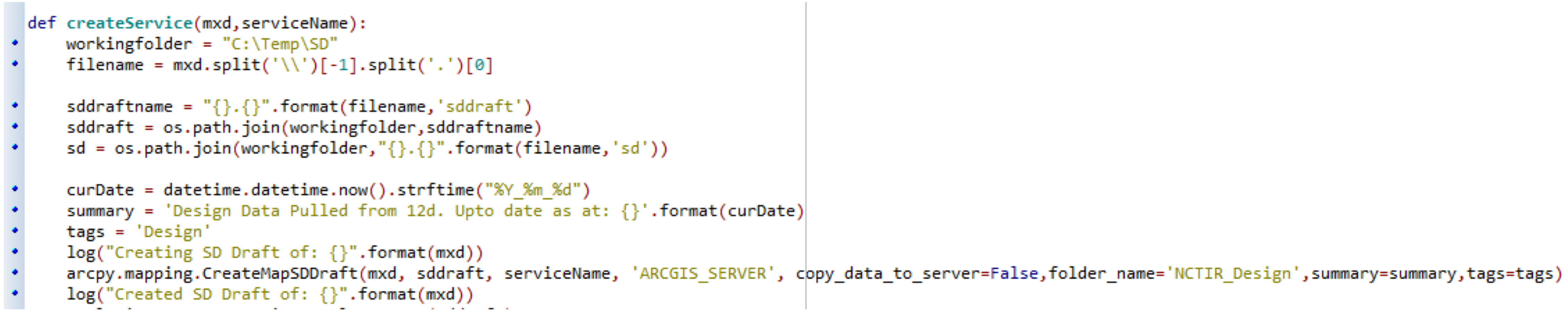
- Stop and Delete old service
- Create service definition draft (.sddraft)
- Analyse .sddraft
- Create service definition (.sd)
- Use .sd to publish service



```python
def deleteservice(server, servicename, username, password, token=None, port=6443):
    log("Deleting Service: {}".format(servicename))
    if token is None:
        token_url = "https://{}/arcgis/admin/generateToken".format(server)
        token = gentoken(token_url, username, password)
    delete_service_url = "https://{}/arcgis/admin/services/{}/delete?token={}".format(server, servicename.replace('\\','/'), token)
    urllib2.urlopen(delete_service_url, ' ').read() # The ' ' forces POST
    log("Deleted Service: {}".format(servicename))
```

Service Creation

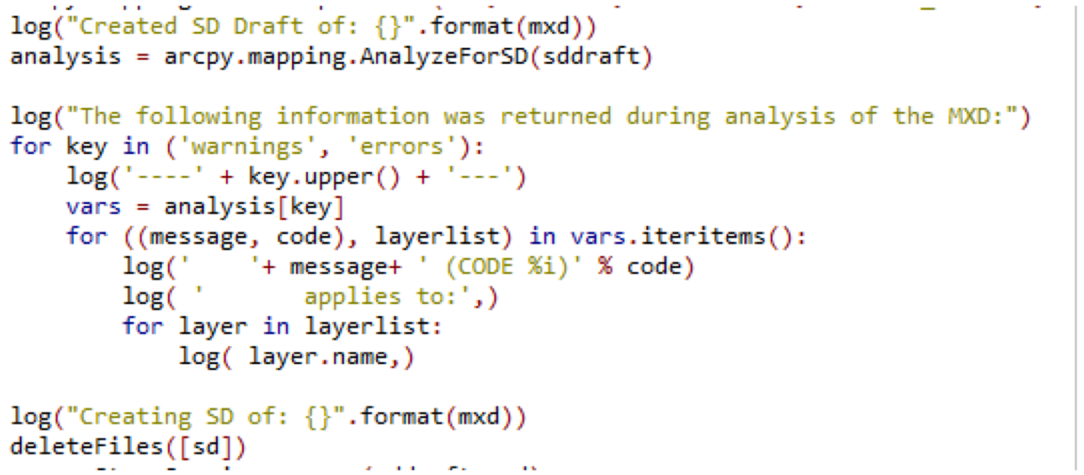
- Stop and Delete old service
- Create service definition draft (.sddraft)
- Analyse .sddraft
- Create service definition (.sd)
- Use .sd to publish service



```
def createService(mxd,serviceName):
    workingfolder = "C:\Temp\SD"
    filename = mxd.split('\\')[-1].split('.')[0]

    sddraftname = "{}.{}".format(filename,'sddraft')
    sddraft = os.path.join(workingfolder,sddraftname)
    sd = os.path.join(workingfolder,"{}.{}".format(filename,'sd'))

    curDate = datetime.datetime.now().strftime("%Y_%m_%d")
    summary = 'Design Data Pulled from 12d. Upto date as at: {}'.format(curDate)
    tags = 'Design'
    log("Creating SD Draft of: {}".format(mxd))
    arcpy.mapping.CreateMapSDDraft(mxd, sddraft, serviceName, 'ARCGIS_SERVER', copy_data_to_server=False,folder_name='NCTIR_Design',summary=summary,tags=tags)
    log("Created SD Draft of: {}".format(mxd))
```

Service Creation

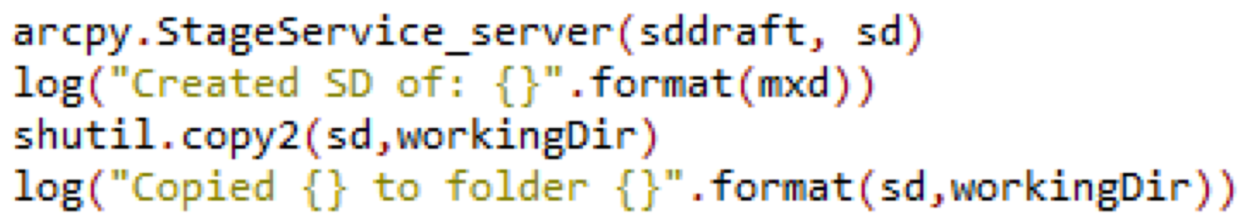
- Stop and Delete old service
- Create service definition draft (.sddraft)
- Analyse .sddraft
- Create service definition (.sd)
- Use .sd to publish service



```
log("Created SD Draft of: {}".format(mxd))
analysis = arcpy.mapping.AnalyzeForSD(sddraft)

log("The following information was returned during analysis of the MXD:")
for key in ('warnings', 'errors'):
    log('----' + key.upper() + '---')
    vars = analysis[key]
    for ((message, code), layerlist) in vars.iteritems():
        log('    '+ message+ ' (CODE %i)' % code)
        log( '       applies to:',)
        for layer in layerlist:
            log( layer.name,)

log("Creating SD of: {}".format(mxd))
deleteFiles([sd])
```

Service Creation

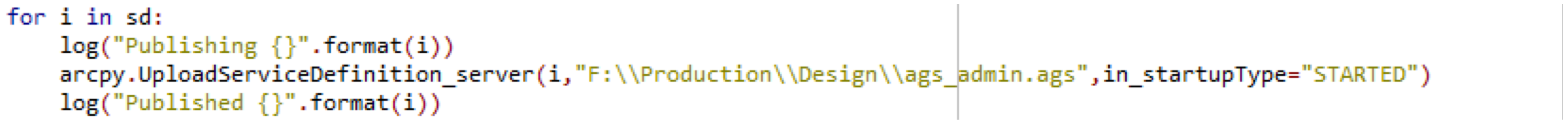
- Stop and Delete old service
- Create service definition draft (.sddraft)
- Analyse .sddraft
- Create service definition (.sd)
- Use .sd to publish service



```
arcpy.StageService_server(sddraft, sd)
log("Created SD of: {}".format(mxd))
shutil.copy2(sd,workingDir)
log("Copied {} to folder {}".format(sd,workingDir))
```

Service Creation

- Stop and Delete old service
- Create service definition draft (.sddraft)
- Analyse .sddraft
- Create service definition (.sd)
- Use .sd to publish service

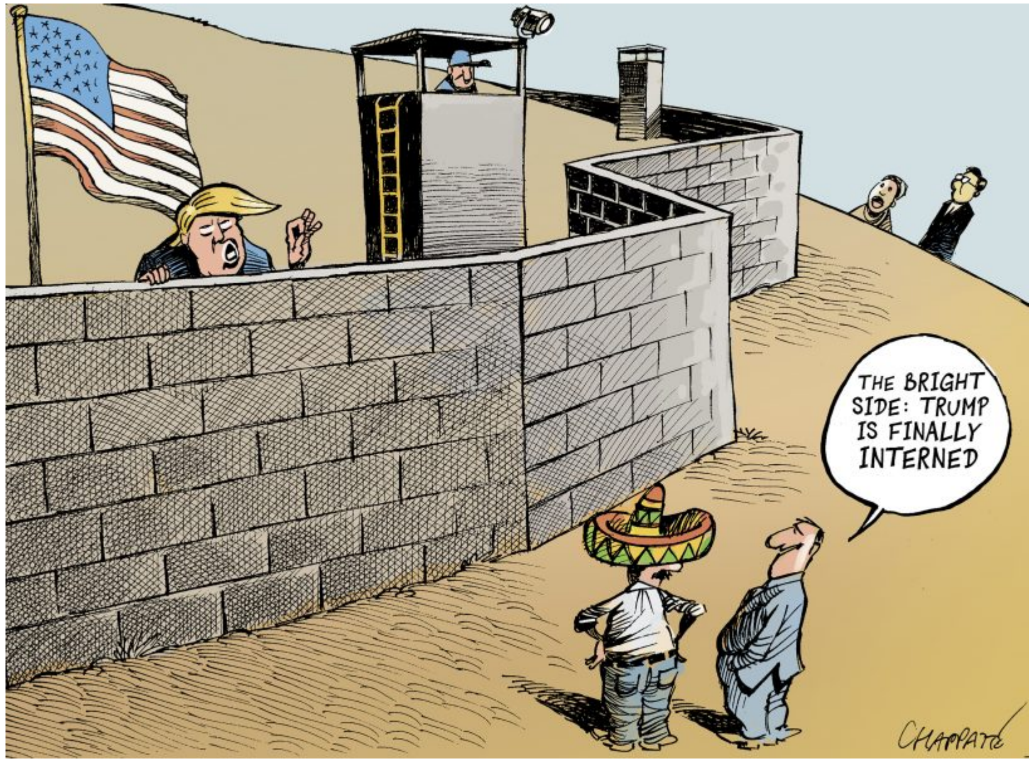


```
for i in sd:
    log("Publishing {}".format(i))
    arcpy.UploadServiceDefinition_server(i,"F:\\Production\\Design\\ags_admin.ags",in_startupType="STARTED")
    log("Published {}".format(i))
```

Scenario 1

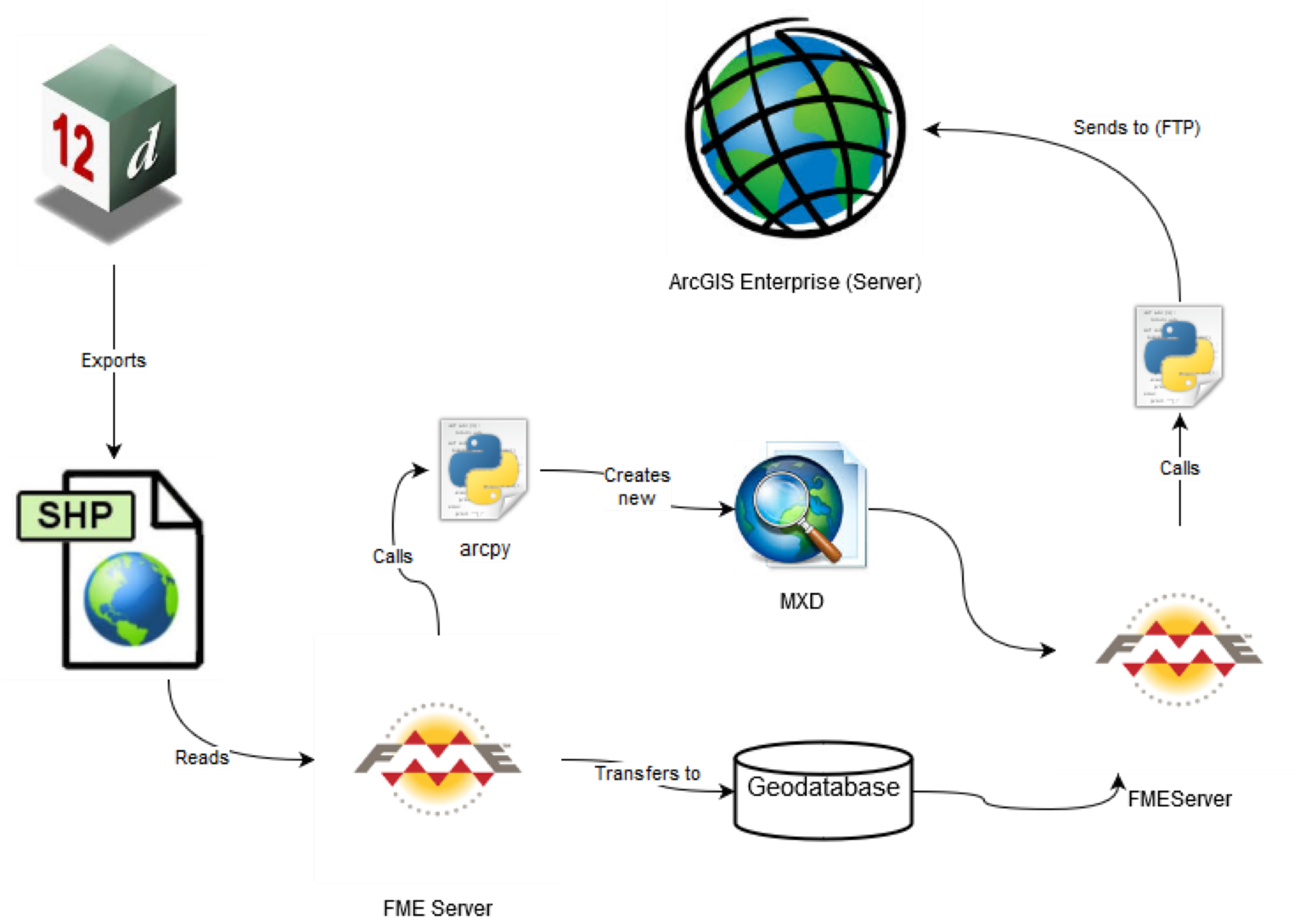
12D Design

The Problems



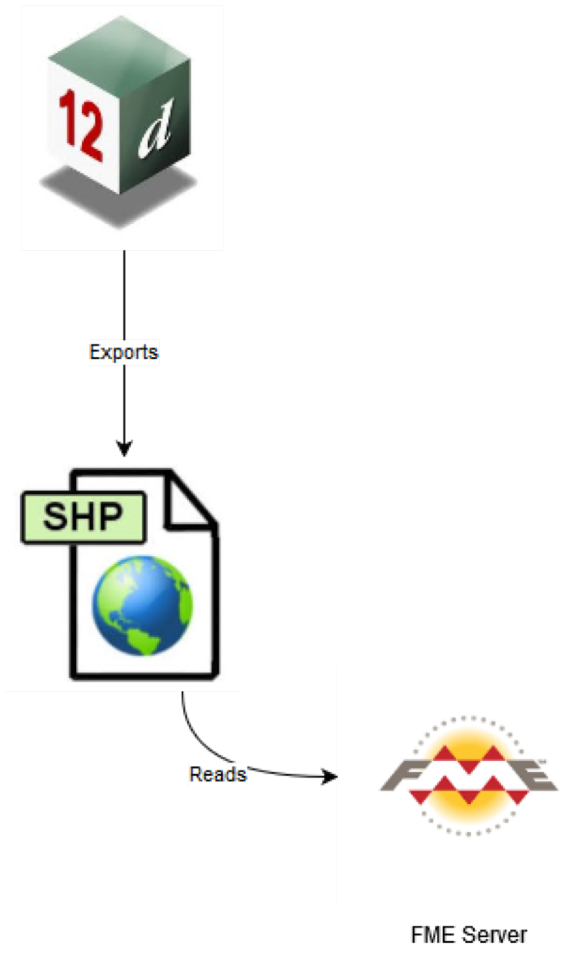
- Designers working in 12D
- No visibility of neighboring projects
- Exports in SHP
- Wanted to view it online with the same symbology

12 Design

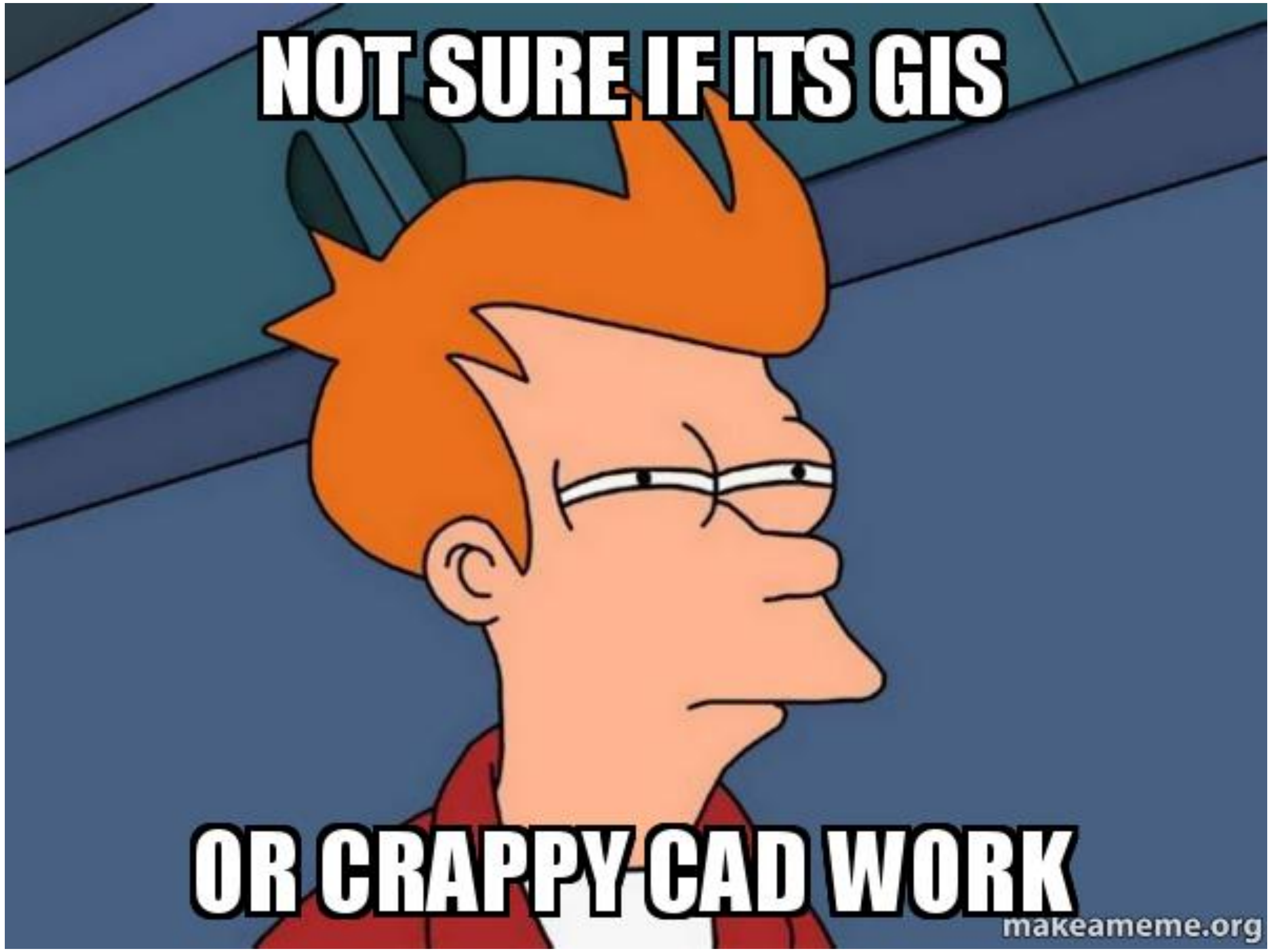


Exports
SHP
Reads
FME Server
Calls
arcpy
Creates new
MXD
Transfers to
Geodatabase
FMEServer
Calls
Sends to (FTP)
ArcGIS Enterprise (Server)

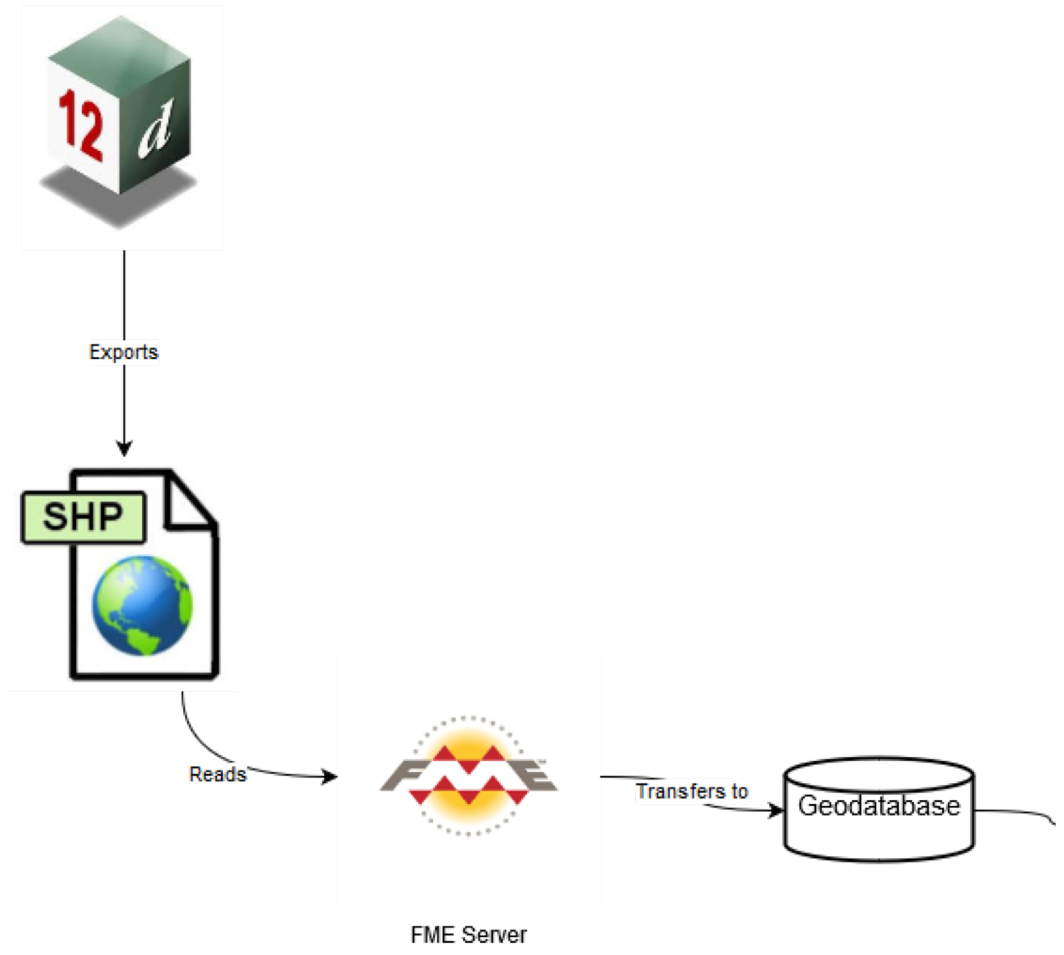
12 Design



- Designers export on demand
- Run FME process nightly

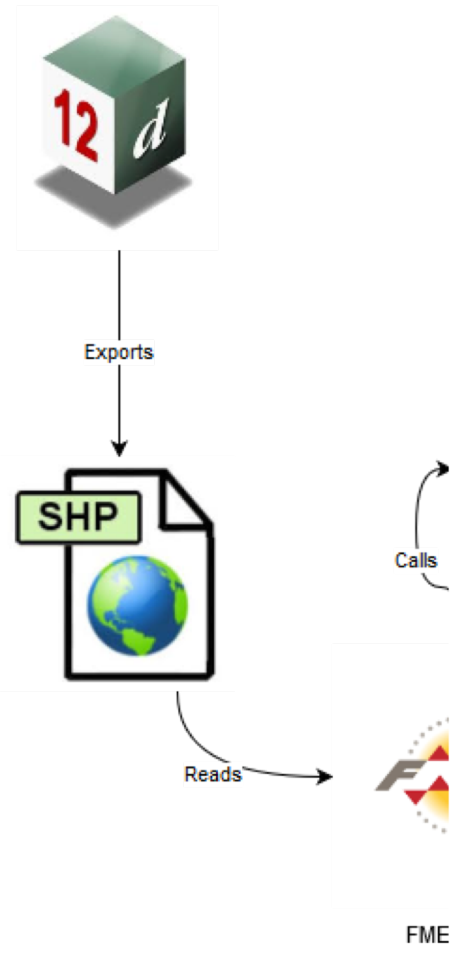

NOT SURE IF ITS GIS
OR CRAPPY CAD WORK
makeameme.org

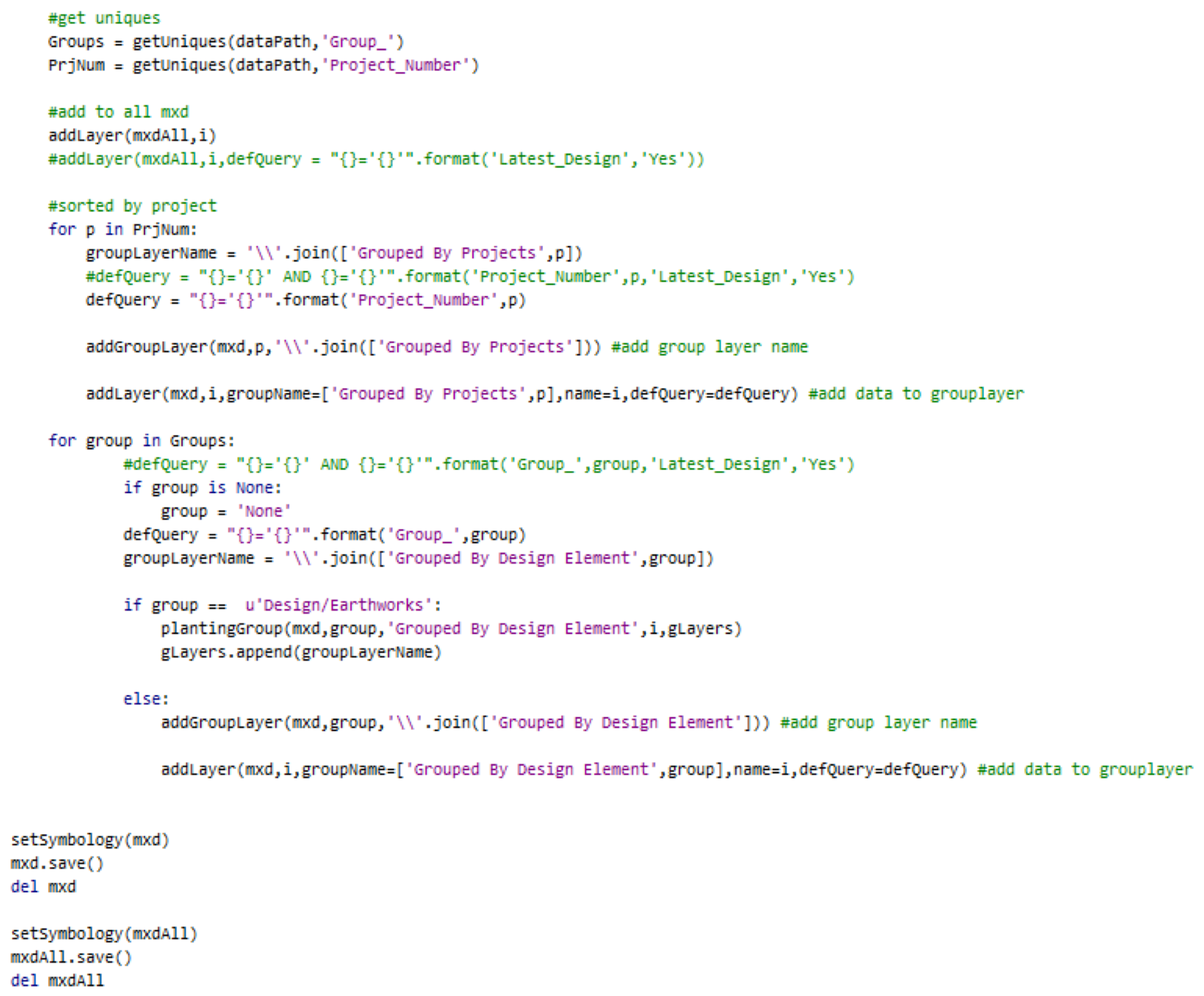
12 Design



- Moved to GDB

12 Des





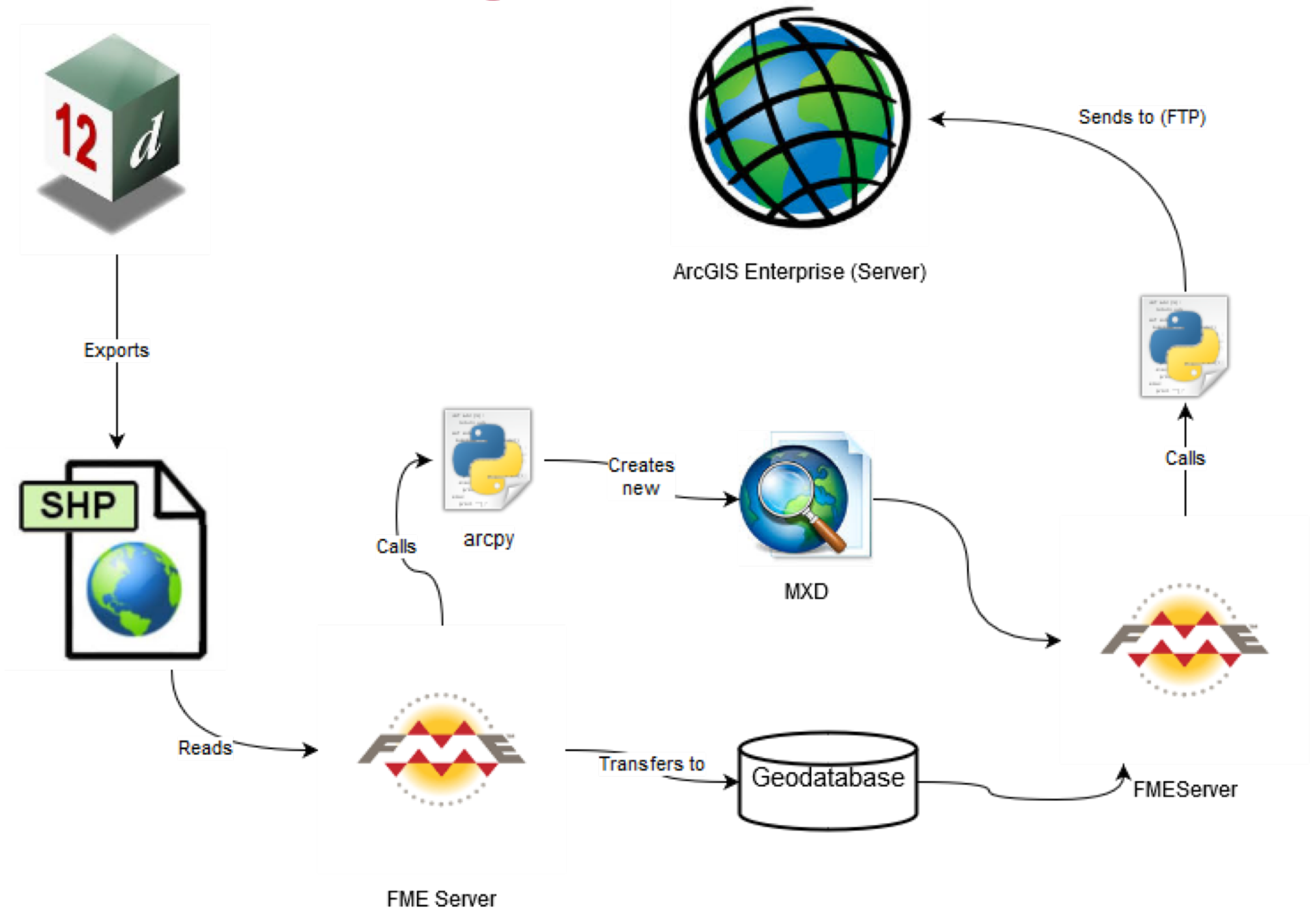
```
    #get uniques
    Groups = getUniques(dataPath,'Group_')
    PrjNum = getUniques(dataPath,'Project_Number')

    #add to all mxd
    addLayer(mxdAll,i)
    #addLayer(mxdAll,i,defQuery = "{}='{}'".format('Latest_Design','Yes'))

    #sorted by project
    for p in PrjNum:
        groupLayerName = '\\'.join(['Grouped By Projects',p])
        #defQuery = "{}='{}' AND {}='{}'".format('Project_Number',p,'Latest_Design','Yes')
        defQuery = "{}='{}'".format('Project_Number',p)

        addGroupLayer(mxd,p,'\\'.join(['Grouped By Projects'])) #add group layer name

        addLayer(mxd,i,groupName=['Grouped By Projects',p],name=i,defQuery=defQuery) #add data to grouplayer

    for group in Groups:
            #defQuery = "{}='{}' AND {}='{}'".format('Group_',group,'Latest_Design','Yes')
            if group is None:
                group = 'None'
            defQuery = "{}='{}'".format('Group_',group)
            groupLayerName = '\\'.join(['Grouped By Design Element',group])

            if group ==  u'Design/Earthworks':
                plantingGroup(mxd,group,'Grouped By Design Element',i,gLayers)
                gLayers.append(groupLayerName)

            else:
                addGroupLayer(mxd,group,'\\'.join(['Grouped By Design Element'])) #add group layer name

                addLayer(mxd,i,groupName=['Grouped By Design Element',group],name=i,defQuery=defQuery) #add data to grouplayer


setSymbology(mxd)
mxd.save()
del mxd

setSymbology(mxdAll)
mxdAll.save()
del mxdAll
```

MXD

plicated

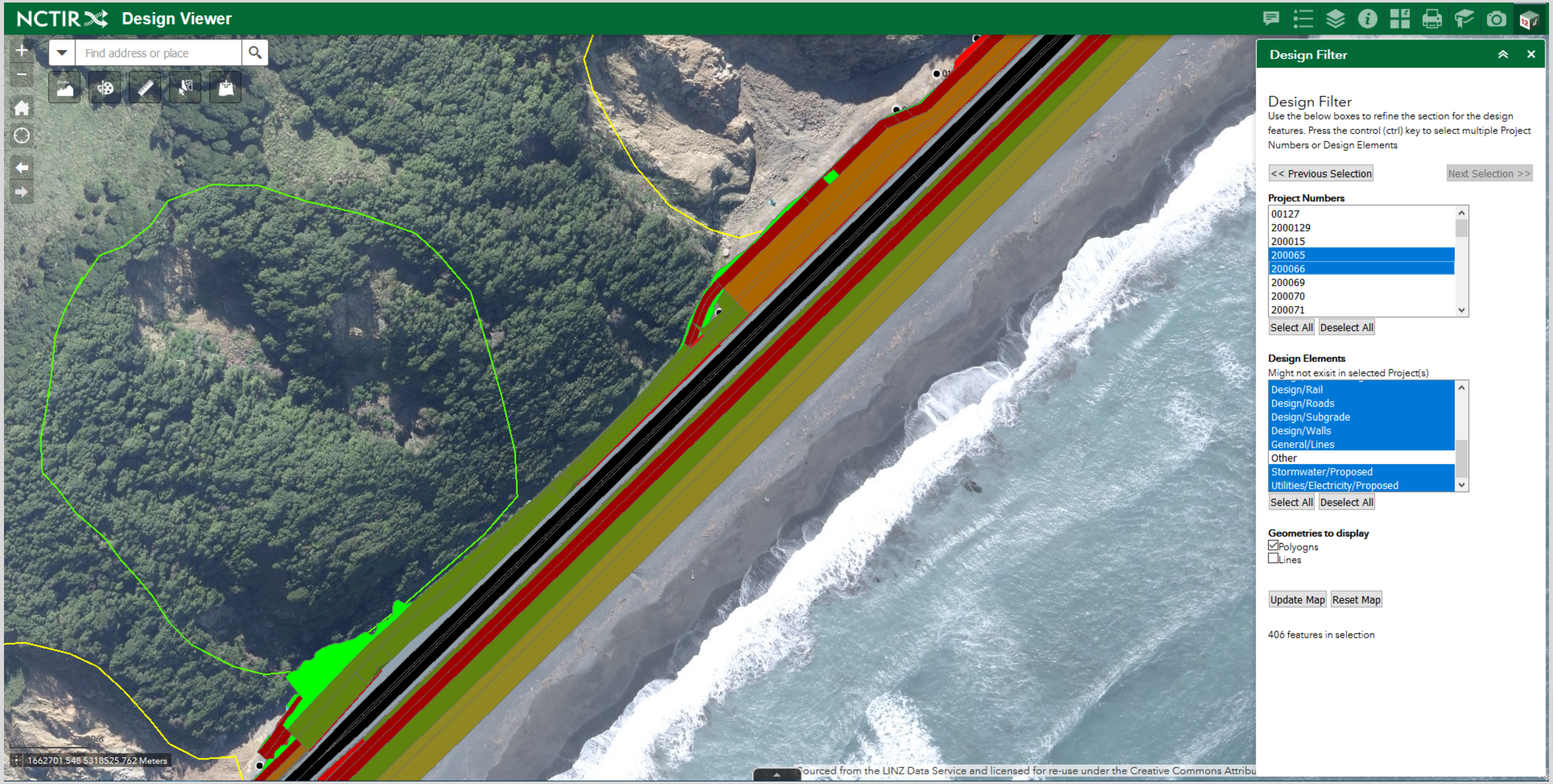
ably

12 Design



- Zipped up and off to the server!

Final Result


NCTIR Design Viewer
Find address or place
1662701.548 5318525.762 Meters
Sourced from the LINZ Data Service and licensed for re-use under the Creative Commons Attribu

Design Filter

Design Filter
Use the below boxes to refine the section for the design features. Press the control (ctrl) key to select multiple Project Numbers or Design Elements

<< Previous Selection   Next Selection >>

Project Numbers
00127
2000129
200015
200065
200066
200069
200070
200071

Select All Deselect All

Design Elements
Might not exisit in selected Project(s)
Design/Rail
Design/Roads
Design/Subgrade
Design/Walls
General/Lines
Other
Stormwater/Proposed
Utilities/Electricity/Proposed

Select All Deselect All

Geometries to display
☑ Polyogns
☐ Lines

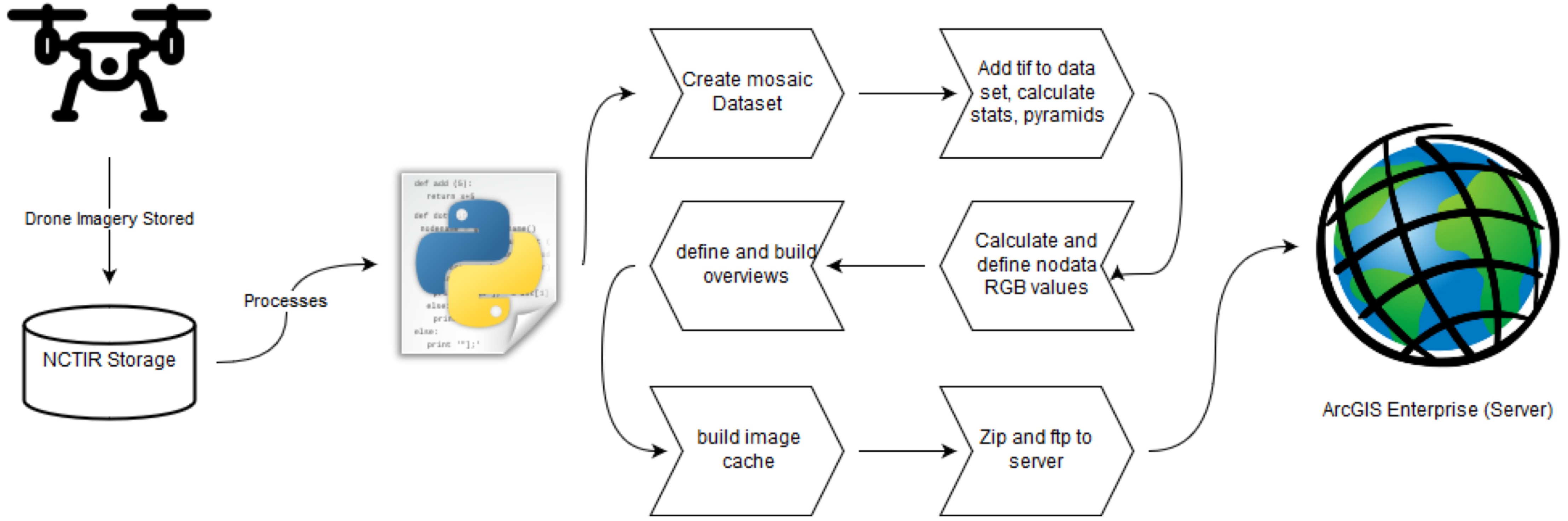
Update Map Reset Map

406 features in selection

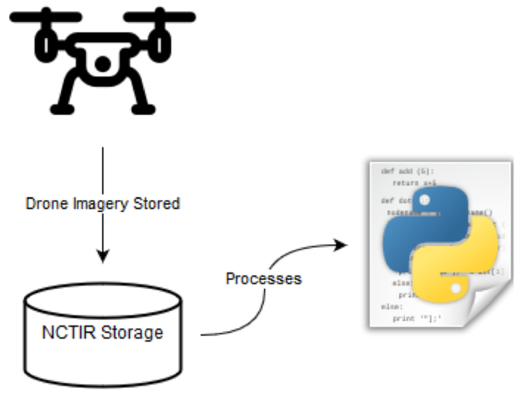
Scenario 2

UAV Surveys

UAV Surveys

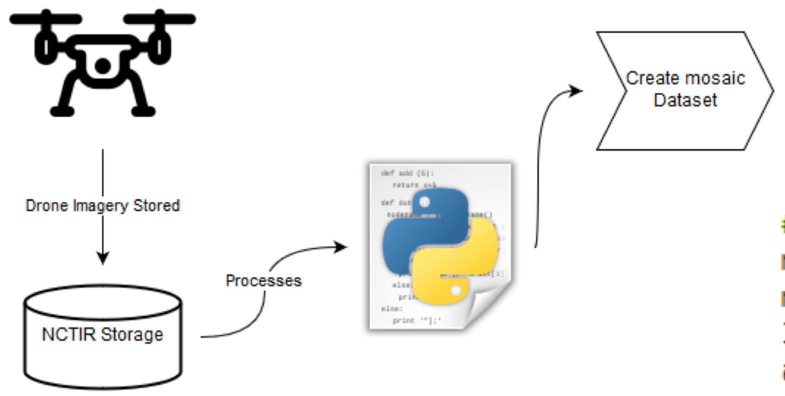

Drone Imagery Stored
NCTIR Storage
Processes
Create mosaic Dataset
Add tif to data set, calculate stats, pyramids
Calculate and define nodata RGB values
define and build overviews
build image cache
Zip and ftp to server
ArcGIS Enterprise (Server)

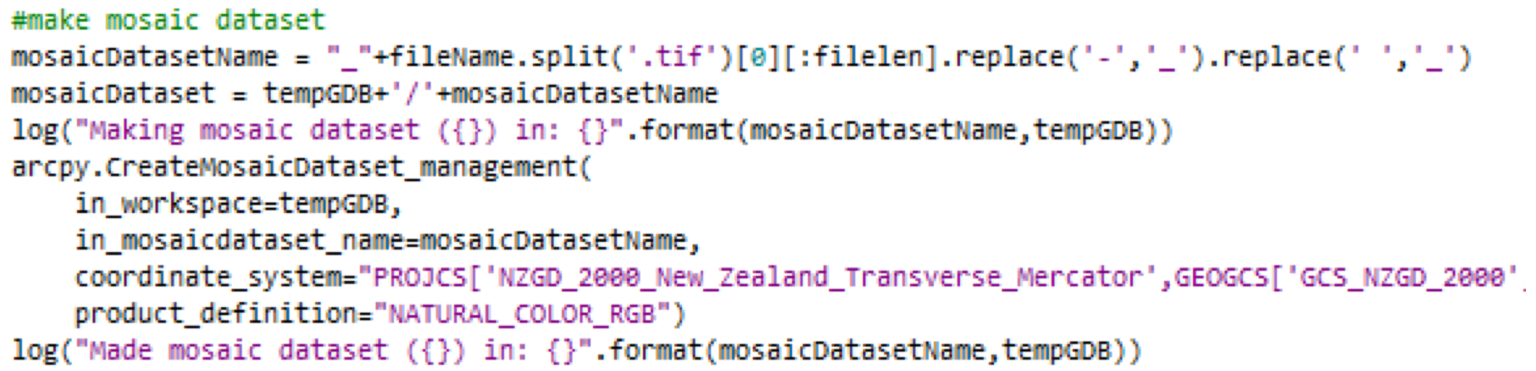
UAV Surveys



- .tifs stored on network
- Python script picks up .tif

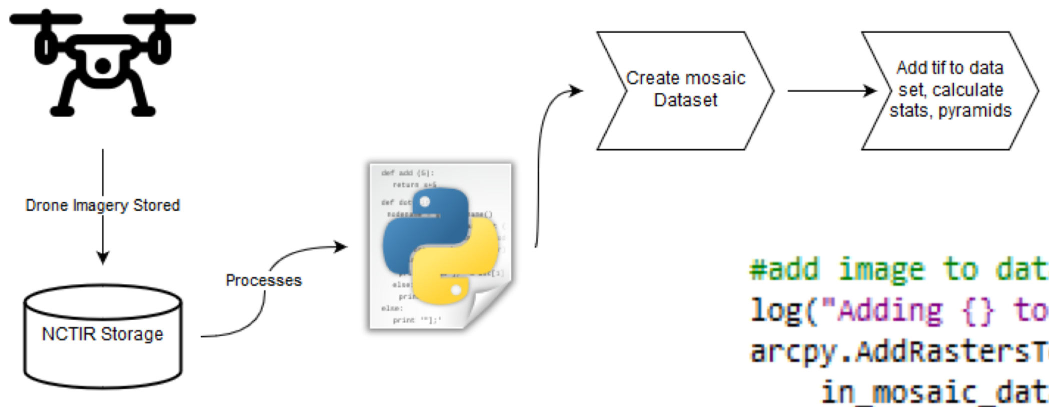
UAV Surveys

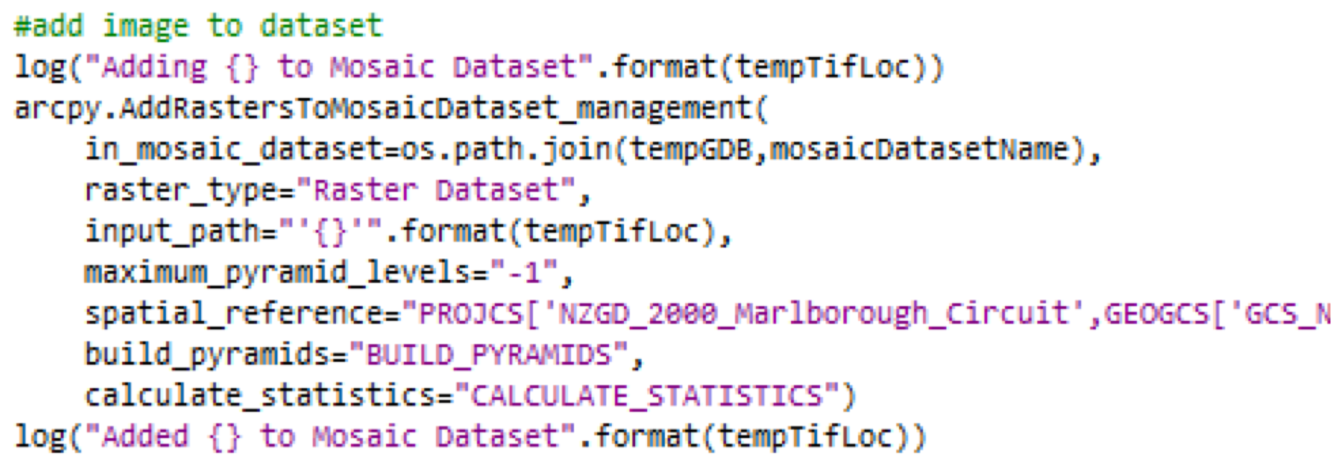




```
#make mosaic dataset
mosaicDatasetName = "_"+fileName.split('.tif')[0][:filelen].replace('-','_').replace(' ','_')
mosaicDataset = tempGDB+'/'+mosaicDatasetName
log("Making mosaic dataset ({}) in: {}".format(mosaicDatasetName,tempGDB))
arcpy.CreateMosaicDataset_management(
    in_workspace=tempGDB,
    in_mosaicdataset_name=mosaicDatasetName,
    coordinate_system="PROJCS['NZGD_2000_New_Zealand_Transverse_Mercator',GEOGCS['GCS_NZGD_2000'
    product_definition="NATURAL_COLOR_RGB")
log("Made mosaic dataset ({}) in: {}".format(mosaicDatasetName,tempGDB))
```

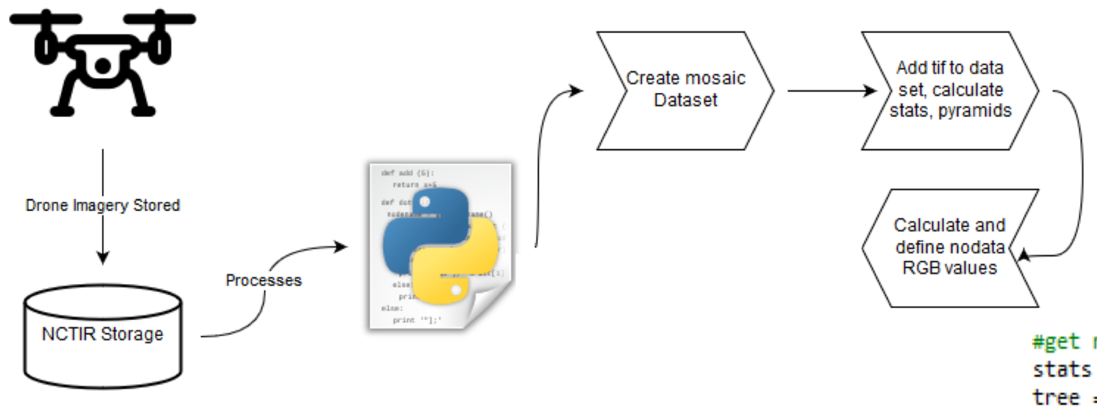
UAV Surveys

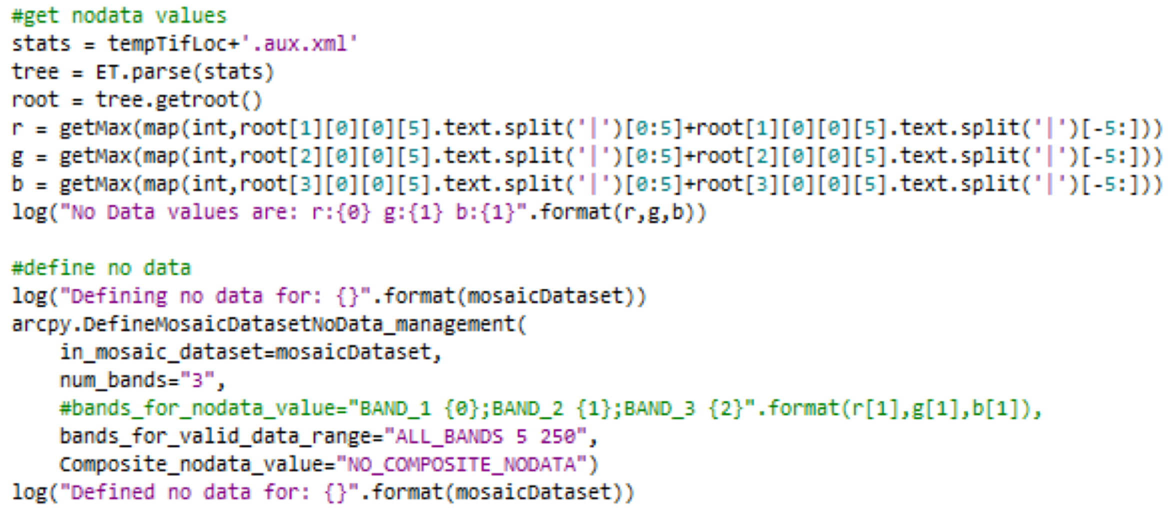




```
#add image to dataset
log("Adding {} to Mosaic Dataset".format(tempTifLoc))
arcpy.AddRastersToMosaicDataset_management(
    in_mosaic_dataset=os.path.join(tempGDB,mosaicDatasetName),
    raster_type="Raster Dataset",
    input_path="'{}'".format(tempTifLoc),
    maximum_pyramid_levels="-1",
    spatial_reference="PROJCS['NZGD_2000_Marlborough_Circuit',GEOGCS['GCS_N
    build_pyramids="BUILD_PYRAMIDS",
    calculate_statistics="CALCULATE_STATISTICS")
log("Added {} to Mosaic Dataset".format(tempTifLoc))
```

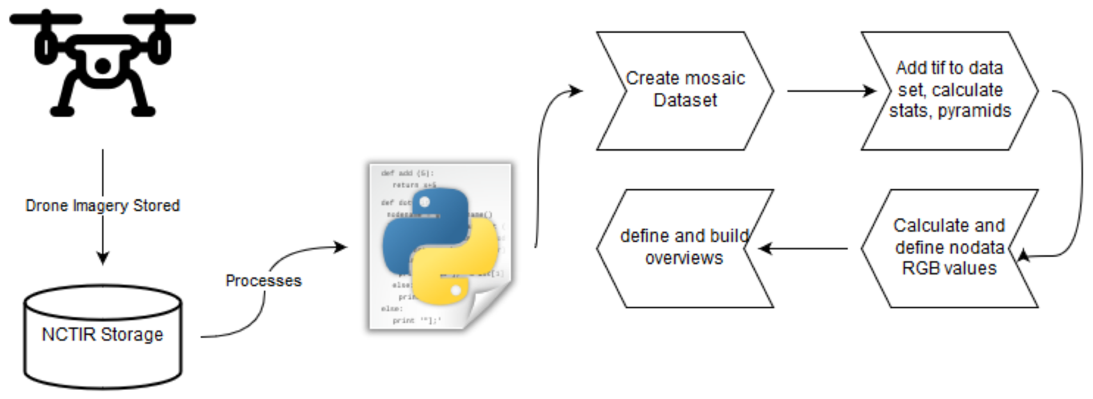
UAV Surveys

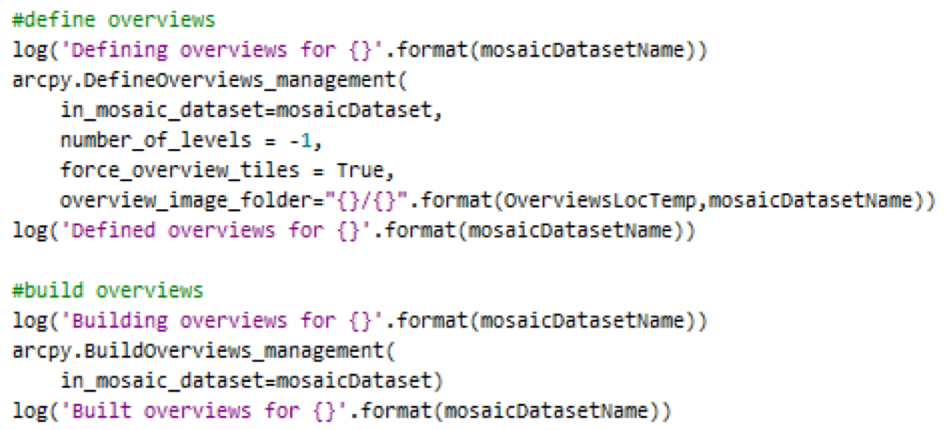

Drone Imagery Stored
NCTIR Storage
Processes
Create mosaic Dataset
Add tif to data set, calculate stats, pyramids
Calculate and define nodata RGB values



```
#get nodata values
stats = tempTifLoc+'.aux.xml'
tree = ET.parse(stats)
root = tree.getroot()
r = getMax(map(int,root[1][0][0][5].text.split('|')[0:5]+root[1][0][0][5].text.split('|')[-5:]))
g = getMax(map(int,root[2][0][0][5].text.split('|')[0:5]+root[2][0][0][5].text.split('|')[-5:]))
b = getMax(map(int,root[3][0][0][5].text.split('|')[0:5]+root[3][0][0][5].text.split('|')[-5:]))
log("No Data values are: r:{0} g:{1} b:{1}".format(r,g,b))

#define no data
log("Defining no data for: {}".format(mosaicDataset))
arcpy.DefineMosaicDatasetNoData_management(
    in_mosaic_dataset=mosaicDataset,
    num_bands="3",
    #bands_for_nodata_value="BAND_1 {0};BAND_2 {1};BAND_3 {2}".format(r[1],g[1],b[1]),
    bands_for_valid_data_range="ALL_BANDS 5 250",
    Composite_nodata_value="NO_COMPOSITE_NODATA")
log("Defined no data for: {}".format(mosaicDataset))
```

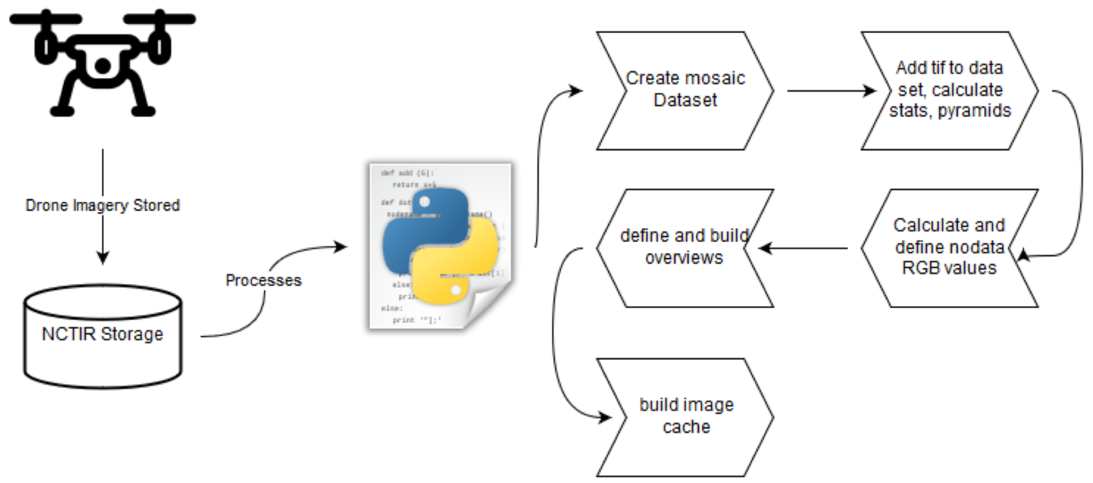
UAV Surveys


Drone Imagery Stored
NCTIR Storage
Processes
Create mosaic Dataset
Add tif to data set, calculate stats, pyramids
Calculate and define nodata RGB values
define and build overviews

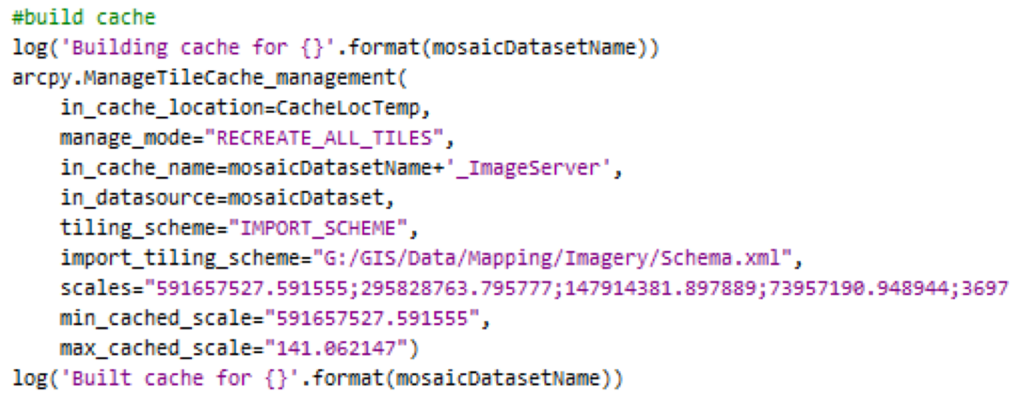


```
#define overviews
log('Defining overviews for {}'.format(mosaicDatasetName))
arcpy.DefineOverviews_management(
    in_mosaic_dataset=mosaicDataset,
    number_of_levels = -1,
    force_overview_tiles = True,
    overview_image_folder="{}/{}".format(OverviewsLocTemp,mosaicDatasetName))
log('Defined overviews for {}'.format(mosaicDatasetName))

#build overviews
log('Building overviews for {}'.format(mosaicDatasetName))
arcpy.BuildOverviews_management(
    in_mosaic_dataset=mosaicDataset)
log('Built overviews for {}'.format(mosaicDatasetName))
```

UAV Surveys

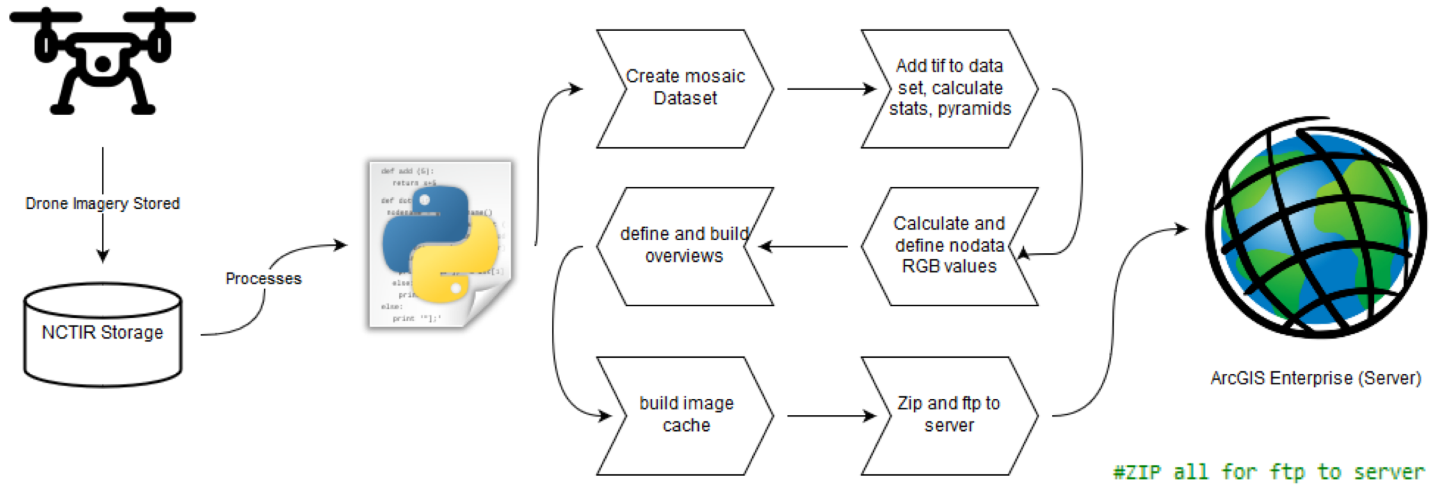


Drone Imagery Stored
NCTIR Storage
Processes
Create mosaic Dataset
Add tif to data set, calculate stats, pyramids
Calculate and define nodata RGB values
define and build overviews
build image cache



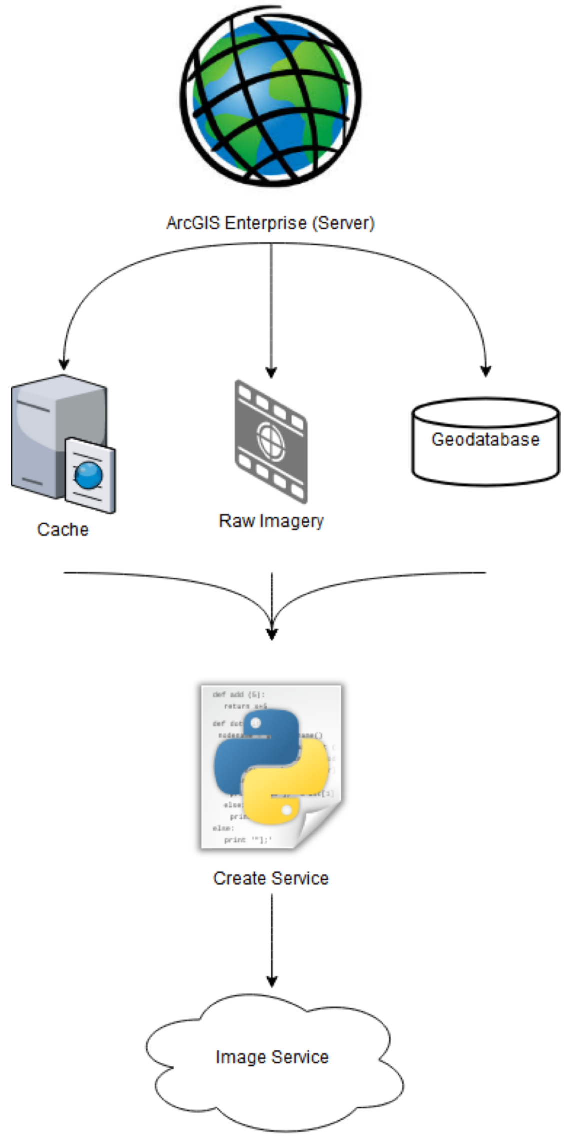
```
#build cache
log('Building cache for {}'.format(mosaicDatasetName))
arcpy.ManageTileCache_management(
    in_cache_location=CacheLocTemp,
    manage_mode="RECREATE_ALL_TILES",
    in_cache_name=mosaicDatasetName+'_ImageServer',
    in_datasource=mosaicDataset,
    tiling_scheme="IMPORT_SCHEME",
    import_tiling_scheme="G:/GIS/Data/Mapping/Imagery/Schema.xml",
    scales="591657527.591555;295828763.795777;147914381.897889;73957190.948944;3697
    min_cached_scale="591657527.591555",
    max_cached_scale="141.062147")
log('Built cache for {}'.format(mosaicDatasetName))
```

UAV Surveys

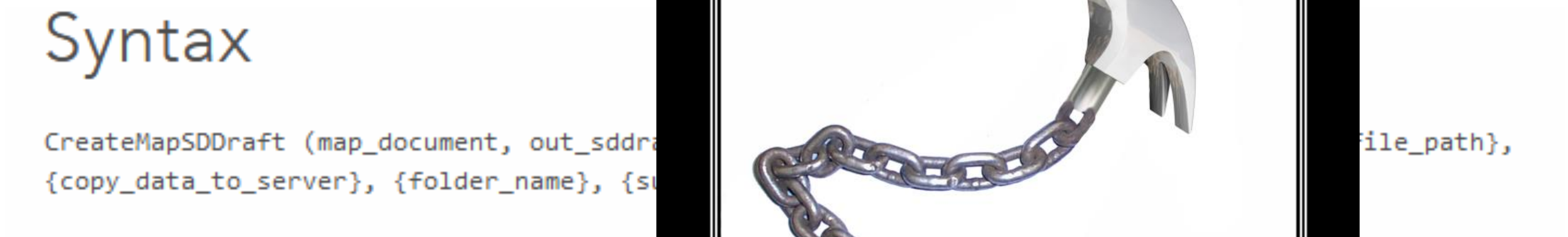

Drone Imagery Stored
NCTIR Storage
Processes
Create mosaic Dataset
Add tif to data set, calculate stats, pyramids
Calculate and define nodata RGB values
define and build overviews
build image cache
Zip and ftp to server
ArcGIS Enterprise (Server)

```
#ZIP all for ftp to server
log('Zipping all files')
with zipfile.ZipFile(mosaicDatasetName+'.zip', 'w', zipfile.ZIP_DEFLATED, allowZip64 = True) as myzip:
    addToZip(os.path.join(CacheLocTemp,mosaicDatasetName+'_ImageServer'),myzip,'cache')
    addToZip(os.path.join(OverviewsLocTemp,mosaicDatasetName),myzip,'overview')
    myzip.write(tempTifLoc, os.path.basename(tempTifLoc))
    addToZip(tempGDB,myzip)
log('Zipped all files')

#ftp to server
log('Moving {} to server via FTP'.format(mosaicDatasetName+'.zip'))
toFtp(mosaicDatasetName+'.zip')
log('Moved{} to server via FTP'.format(mosaicDatasetName+'.zip'))
```

UAV Surveys


ArcGIS Enterprise (Server)
Cache
Raw Imagery
Geodatabase
Create Service
Image Service

UAV Surveys



Syntax

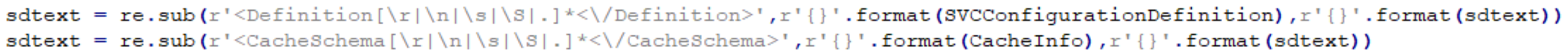
```
CreateMapSDDraft (map_document, out_sddr                                         ile_path},
{copy_data_to_server}, {folder_name}, {s
```

What is a .sddraft?

XML

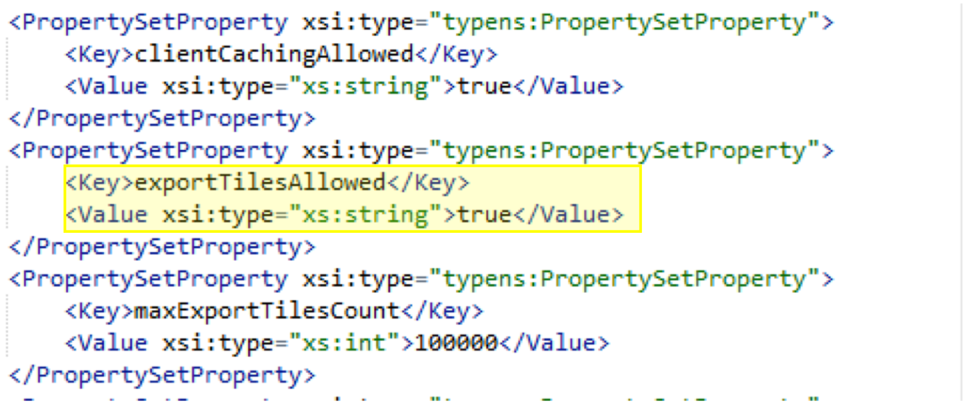
Edit the XML

- Regex to the rescue
- Made a ‘template’ service in ArcMap
- Replace the required sections

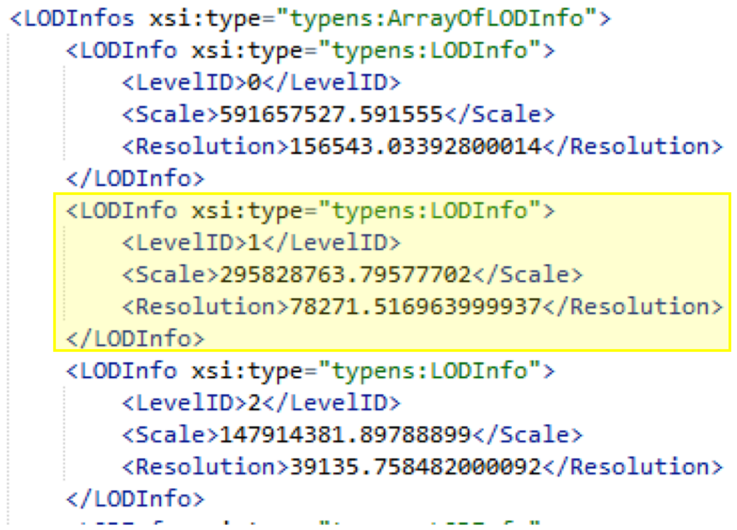


```
sdtext = re.sub(r'<Definition[\r|\n|\s|\S|.]*<\/Definition>',r'{}'.format(SVCConfigurationDefinition),r'{}'.format(sdtext))
sdtext = re.sub(r'<CacheSchema[\r|\n|\s|\S|.]*<\/CacheSchema>',r'{}'.format(CacheInfo),r'{}'.format(sdtext))
```

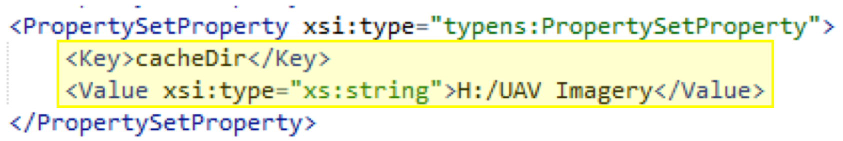
Edit the XML



```
<PropertySetProperty xsi:type="typens:PropertySetProperty">
    <Key>clientCachingAllowed</Key>
    <Value xsi:type="xs:string">true</Value>
</PropertySetProperty>
<PropertySetProperty xsi:type="typens:PropertySetProperty">
    <Key>exportTilesAllowed</Key>
    <Value xsi:type="xs:string">true</Value>
</PropertySetProperty>
<PropertySetProperty xsi:type="typens:PropertySetProperty">
    <Key>maxExportTilesCount</Key>
    <Value xsi:type="xs:int">100000</Value>
</PropertySetProperty>
```

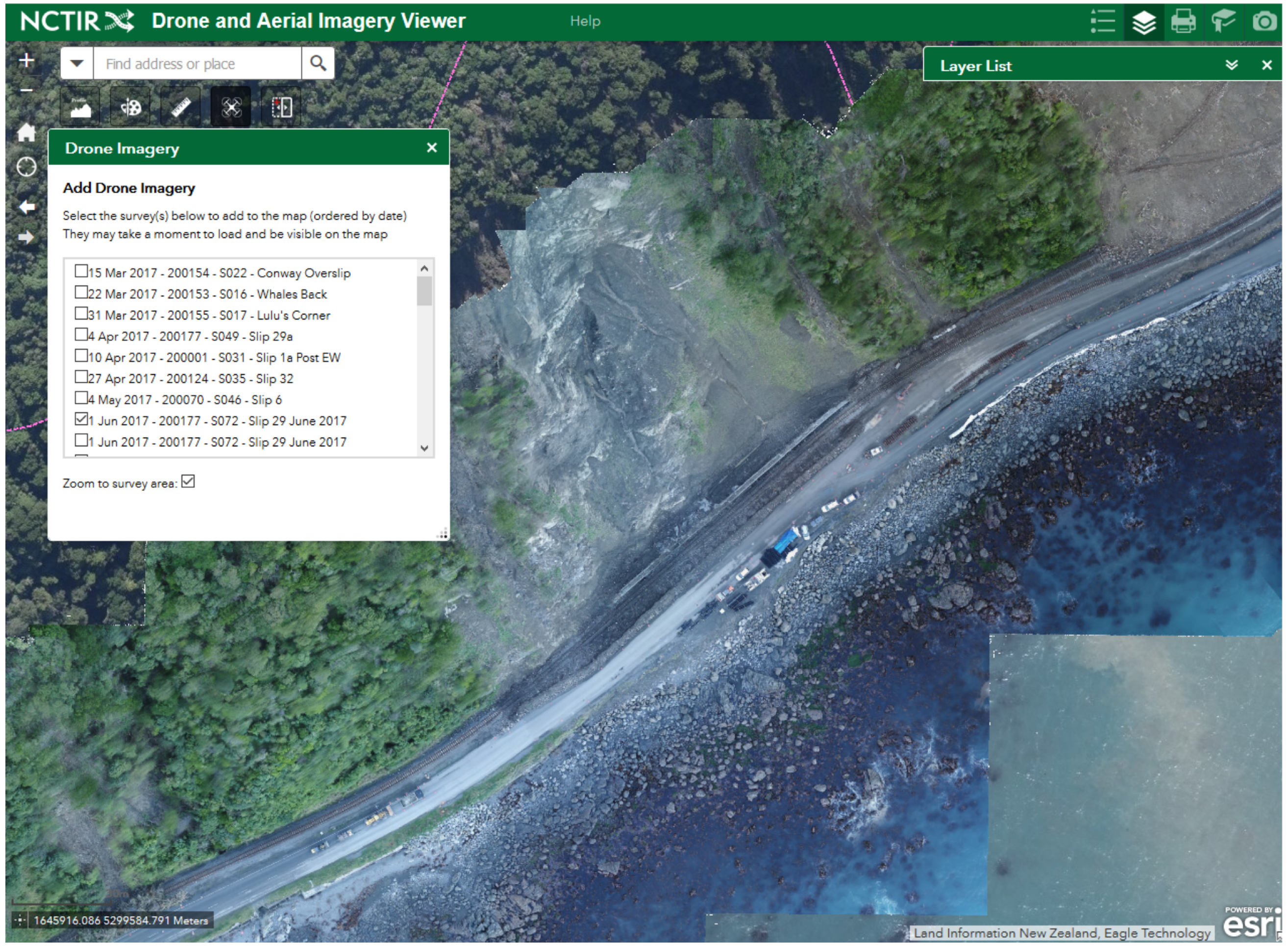


```
<LODInfos xsi:type="typens:ArrayOfLODInfo">
    <LODInfo xsi:type="typens:LODInfo">
        <LevelID>0</LevelID>
        <Scale>591657527.591555</Scale>
        <Resolution>156543.03392800014</Resolution>
    </LODInfo>
    <LODInfo xsi:type="typens:LODInfo">
        <LevelID>1</LevelID>
        <Scale>295828763.79577702</Scale>
        <Resolution>78271.516963999937</Resolution>
    </LODInfo>
    <LODInfo xsi:type="typens:LODInfo">
        <LevelID>2</LevelID>
        <Scale>147914381.89788899</Scale>
        <Resolution>39135.758482000092</Resolution>
    </LODInfo>
```



```
<PropertySetProperty xsi:type="typens:PropertySetProperty">
    <Key>cacheDir</Key>
    <Value xsi:type="xs:string">H:/UAV Imagery</Value>
</PropertySetProperty>
```

Result


NCTIR
Drone and Aerial Imagery Viewer
Help
Find address or place
Layer List
1645916.086 5299584.791 Meters
Land Information New Zealand, Eagle Technology
POWERED BY esri

Drone Imagery

Add Drone Imagery

Select the survey(s) below to add to the map (ordered by date)
They may take a moment to load and be visible on the map

- ☐ 15 Mar 2017 - 200154 - S022 - Conway Overslip
- ☐ 22 Mar 2017 - 200153 - S016 - Whales Back
- ☐ 31 Mar 2017 - 200155 - S017 - Lulu's Corner
- ☐ 4 Apr 2017 - 200177 - S049 - Slip 29a
- ☐ 10 Apr 2017 - 200001 - S031 - Slip 1a Post EW
- ☐ 27 Apr 2017 - 200124 - S035 - Slip 32
- ☐ 4 May 2017 - 200070 - S046 - Slip 6
- ☑ 1 Jun 2017 - 200177 - S072 - Slip 29 June 2017
- ☐ 1 Jun 2017 - 200177 - S072 - Slip 29 June 2017

Zoom to survey area: ☑

- NCTIR UAV Imagery/ 200001 S031 Slip 1A Post E (ImageServer)
- NCTIR UAV Imagery/ 200021 S419 Haupuku Stockp (ImageServer)
- NCTIR UAV Imagery/ 200022 S286 Tunnel 12 UAV (ImageServer)
- NCTIR UAV Imagery/ 200028 S353 SR14 Sep 2017 (ImageServer)
- NCTIR UAV Imagery/ 200065 S121 Slip 1B June 2 (ImageServer)
- NCTIR UAV Imagery/ 200065 S252 Slip 1b August (ImageServer)
- NCTIR UAV Imagery/ 200065 S308 Site 1 Sep 201 (ImageServer)
- NCTIR UAV Imagery/ 200066 S109 Slip 2 June 20 (ImageServer)
- NCTIR UAV Imagery/ 200066 S308 A1 Irongate to (ImageServer)
- NCTIR UAV Imagery/ 200066 S308 A2 Irongate to (ImageServer)
- NCTIR UAV Imagery/ 200066 S308 A3 Irongate to (ImageServer)
- NCTIR UAV Imagery/ 200070 S046 Slip 6 May 201 (ImageServer)
- NCTIR UAV Imagery/ 200071 S097 Slip 7 June 20 (ImageServer)
- NCTIR UAV Imagery/ 200071 S233 Slip 6 Aug 201 (ImageServer)
- NCTIR UAV Imagery/ 200071 S233 Slip 7 Aug 201 (ImageServer)
- NCTIR UAV Imagery/ 200124 S035 Raramai Tunnel (ImageServer)
- NCTIR UAV Imagery/ 200127 S628 Site SR14 Marc (ImageServer)
- NCTIR UAV Imagery/ 200127 S664 Peketa SR1 2 2 (ImageServer)
- NCTIR UAV Imagery/ 200128 S397 Paparoa Point (ImageServer)
- NCTIR UAV Imagery/ 200128 S582 The Hump Feb 2 (ImageServer)
- NCTIR UAV Imagery/ 200134 S243 Parititahi Tun (ImageServer)
- NCTIR UAV Imagery/ 200153 S016 Whales Back Di (ImageServer)
- NCTIR UAV Imagery/ 200154 S022 Conway Oversli (ImageServer)
- NCTIR UAV Imagery/ 200155 S017 Lulus Hair Pin (ImageServer)
- NCTIR UAV Imagery/ 200166 S573 Tunnel 18 Sout (ImageServer)
- NCTIR UAV Imagery/ 200177 S049 KKS Site 29a M (ImageServer)
- NCTIR UAV Imagery/ 200177 S072 Foreshore Slip (ImageServer)
- NCTIR UAV Imagery/ 200177 S072 Hillside Slip (ImageServer)
- NCTIR UAV Imagery/ 200179 S391 KR Bridge 129 (ImageServer)
- NCTIR UAV Imagery/ 200186 S149 Slip 5 June 20 (ImageServer)
- NCTIR UAV Imagery/ 200187 S348 29a and Hapura (ImageServer)
- NCTIR UAV Imagery/ 200194 S550 UAV and Topo a (ImageServer)
- NCTIR UAV Imagery/ 200197 S308 Site 1 to Iron (ImageServer)
- NCTIR UAV Imagery/ 200219 S448 Inland Road RP (ImageServer)
- NCTIR UAV Imagery/ 200219 S512 Displaced Gabi (ImageServer)
- NCTIR UAV Imagery/ 300003 S458 Okiwi Bay Oct (ImageServer)
- NCTIR UAV Imagery/ 300003 S458 Paparoa Point (ImageServer)
- NCTIR UAV Imagery/ 300003 S475 Rakautara Amen (ImageServer)
- NCTIR UAV Imagery/ 300004 S589 Haupuku Shared (ImageServer)
- NCTIR UAV Imagery/ 300004 S590 Site 2 to Nins (ImageServer)
- NCTIR UAV Imagery/ 300004 S661 Half Moon Bay (ImageServer)

Summary

- Automation (ArcPy) has saved time
- Provided greater support for other teams
- Simplified workflows

abley

WRITES 10 LINE PYTHON SCRIPT
ELIMINATES THE NEED FOR 5 JOBS
quickmeme.com

Thanks

Any Questions?

Set and Forget

Automated Service Authoring

Hamish Kingsbury PGDipGIS BSc
Senior Spatial Data Specialist

M +64 27 373 9475
E Hamish.Kingsbury@abley.com

ably