Migrating a Century’s Worth of Easements into GIS

by

Jason Ramsey
About APS

• Since 1886, APS has served Arizona’s growing energy needs.

• Provides safe dependable electricity for 1.1 million customers in 11 of Arizona’s 15 counties.

• Nearly 6,700 APS employees
Easements

A Strip of Land 15.0 feet in width lying within Section 30, Township 7 North, Range 3 East of the Glo and Salt River Meridian, Maricopa County, Arizona, said strip lying 7.5 feet on each side of the following described centerline:

COMMENCING at a point on the North line 2958.0 feet west of the Northeast corner of said Section 30, said point being the POINT OF BEGINNING of said centerline;
Thence to a point 2398.0 feet west of the East line and 63.0 feet south of the North line of said Section 30;
Thence to a point 2099.0 feet west of the East line and 200.0 feet south of the North line of said Section 30;
Thence to a point 2079.0 feet west of the East line and 1147.0 feet south of the North line of said Section 30;
Thence to a point 2613.0 feet east of the West line and 1341.0 feet south of the North line of said Section 30;
Thence to a point 2611.0 feet east of the West line and 2625.0 feet north of the South line of said Section 30, said point is also the POINT OF TERMINUS of said centerline.

Containing 0.07 acres, more or less.
Easements

• Past 5 years Land Services has been populating the Land Service Database (SQL) with archived Easements
• When possible land agents have assigned point coordinates to easements in database
• Currently 91,000 unique records in database
Work flow before GIS migration

- Land Agent would query database by TRSQ and interpret legal descriptions for relevance.
Migration of Easements into GIS
Problem

How to create a GIS point layer from a non-GIS database?

• Considerations
  – Inconsistent, non-standardized data entry
  – No fields specific to locational data
    • stored in multiple fields not intended for location ("contract_num","comments", etc)
  – All records tied to E-Docs (scanned easement document)
  – Multiple types of spatial information(lat/long, street address, parcel number, TRSQ(township, range, section, quarter)
Requirements

• Cannot disrupt traditional data-entry by non-GIS people
• Allows non-GIS people ability to edit locational information tied to the database records
• Must be automatically updated on a regular scheduled basis
Solution

```python
from pandas import DataFrame, read_csv
import numpy as np
import pandas as pd
import arcpy
import geopy
from geopy import Nominatim
from tkinter import *
import tkinter.filedialog
import geopandas as gp
import os
from shapely.geometry import Point
from geopandas import sjoin

## select csv and load into Pandas dataframe
root = Tk()
root.withdraw()
Location= tkFileDialog.askopenfilename(filetypes = [('csv files', '*.csv'),title='Select a csv')
df = pd.read_csv(Location,usecols=['tbl_id','apn','contract_num','site_address','township','range','section'],
print df.shape[0]
### drop duplicate tbl_id records and concatenate all columns
df2=df.reset_index().drop_duplicates('tbl_id',take_last=True).set_index('tbl_id').sort()
df3 = df2.apply(lambda x: x.join(map(str, row)), axis=1)
### Find all the points
reExpressionPoints='\d\d[.]\d[/\d]+'.replace('\','').replace('.','').replace('D','D').replace('0','0').replace('d','d')
foundPoints=df3.str.findall(reExpressionPoints,re.MULTILINE).apply(pd.Series).stack()
foundPoints.index.name='tbl_id'
foundPointsDF=foundPoints.to_frame('foundPointsDF')
### Calculate pointCount for all tbl_id records
foundPointsDF['indexCount']=foundPointsDF.index
foundPointsDF['PointCount'] = foundPointsDF['indexCount'].apply(len)
del foundPointsDF['indexCount']
foundPointsDF.columns=['PointCoord','PointCount']
print foundPointsDF.shape[0]
### isolate records with only one point
Point1=foundPointsDF[foundPointsDF.PointCount==1]
### create point dataframe from foundPointsDF to append points from other sources (APN, geocoding, IRSQ)
MultiSourcePoint_DF=foundPointsDF['PointCoord'].to_frame('PointCoord')

```
```
Key python modules

• Pandas
  – Fast manipulation of big data in a dataframe format
• Geopandas
  – Adds simple geoprocessing to pandas dataframe
• Geopy
  – Geocoding addresses
• Arcpy
  – Importing points layer into geodatabase
Key elements

• Regular expressions

Expression for point coordinates = '3\d+.\d+|W+-1\d+/\d+/|1\d+.\d+/W+3\d+.\d+'

>>>['33.34,-100.345', '-111.345 34.234556']

Expression for finding parcel numbers = r’ \w+-\w-\w+\w+’


Expression for finding addresses= r'\b\d{1,6}(?:s+.+[0-9]{5})\b’

Results

• Script takes approximately 20 minutes to run
• >140,000 points created
Work flow before GIS migration

- Land Agent would query database by TRSQ and interpret legal descriptions for relevance
Work flow after GIS migration

• Land Agent navigates to area of interest and hyperlinks to database records and documents
Questions?

Jason.ramsey@aps.com