2016

GIS Data Preparation & Integration Tools Documentation

COLORADO WEST REGION DATA INTEGRATION GROUP

HEATHER WIDLUND, GIS COORDINATOR, SAN MIGUEL COUNTY, CO

HEATHERW@SANMIGUELCOUNTYCO.GOV

HTTPS://GITHUB.COM/HWIDLUND/INTEGRATION

CONTENTS

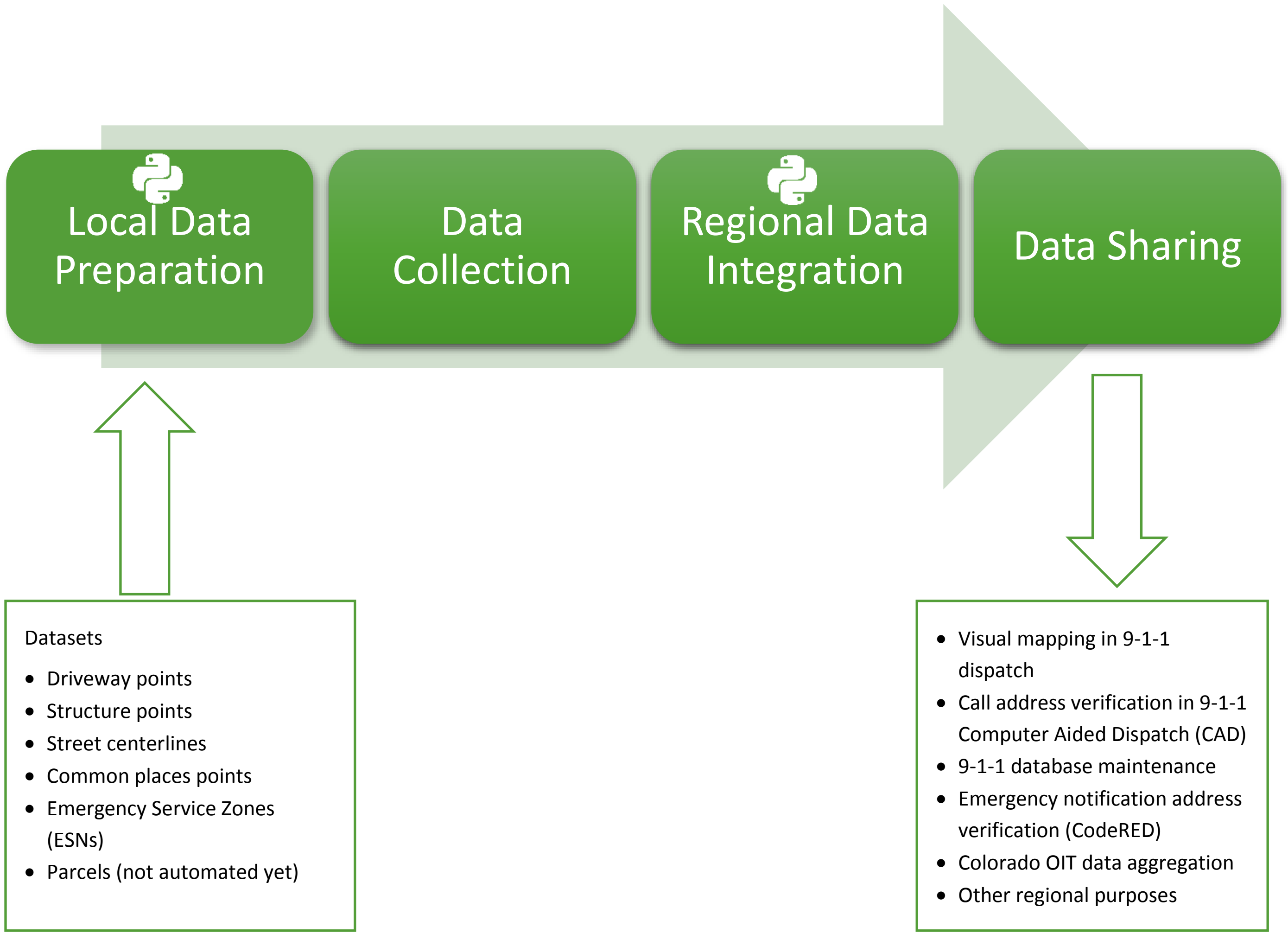
PROJECT DESCRIPTION

BACKGROUND

A group of several counties and cities in Southwest Colorado integrate data for emergency preparedness and response purposes. Although data schemas have been agreed to, most agencies are required to preprocess their data in advance of integration to comply. This project addresses the common extraction and translation needs between source and integrated data. This documentation describes the data preparation at the source agencies and the integration of the prepared data. Scripts and functions are described in detail in a separate document.

AUTOMATED TOOLS

A set of automated tools has been developed using Python, the arcpy library, and ArcGIS script tools, with the goal of increasing the frequency and completeness of data updates. Automation has been applied to two stages of the process: data preparation at the source agencies, and integration of the prepared data. The data are still collected manually after preparation (currently using a shared Google Drive location), and distributed manually after integration.



Datasets

- Driveway points
- Structure points
- Street centerlines
- Common places points
- Emergency Service Zones (ESNs)
- Parcels (not automated yet)

- Visual mapping in 9-1-1 dispatch
- Call address verification in 9-1-1 Computer Aided Dispatch (CAD)
- 9-1-1 database maintenance
- Emergency notification address verification (CodeRED)
- Colorado OIT data aggregation
- Other regional purposes

QUICK REFERENCE GUIDE

UPDATING YOUR DATA

- If you have any configuration changes to make, see below guide on how to do so.
- Close any open instances of ArcGIS Desktop applications & Python IDEs.
- Double-click on the `batchrunupdate.bat` file. (It may be named using your agency code.)
- If you receive an error message, check the `warnlog.log` file for errors, or the `debuglog.log` for details. The log files are located in the logs folder of the deployment package and can be opened in Notepad.
- If the tool does not run (disappears quickly with no message), make sure there aren't any open instances of ArcCatalog or ArcMap creating schema locks.
- Navigate to the outputdata folder.
    - If the folder doesn't contain a new zip file, check the logs for errors, fix, and re-run.
    - Otherwise, upload the new zip file to Google Drive in the Consolidated Source Data folder.
    - Log the date of upload on the Consolidated GDB tab of the Data Update Log.

UPDATING CONFIGURATION PARAMETERS

Updating dataset locations

- Open the Integration Toolbox.
- For each dataset that has configuration changes, open the script tool associated with the dataset (i.e. "Configure Road Layer" for the Roads dataset).
- Use the GUI to browse to the new dataset locations.
- Fill in the rest of the parameters.
- Run the tool.
- *If the **outputdata folder** (not the data source) location has changed*, also edit the `batchrunupdate.bat` file to reflect the new full path to it. See section below on "Updating location of config, logs, outputdata or scripts folders"

Updating source data field names

- Open the Integration Toolbox.
- For each dataset that has configuration changes, open the script tool associated with the dataset (i.e. "Configure Road Layer" for the Roads dataset).
- Use the GUI to choose new values for the field names in the associated boxes.
- Fill in the rest of the parameters.
- Run the tool.

Updating schema data field names

If an integration data schema is updated, certain values in various scripts and script tools must be changed. Please notify the author.

UPDATING OTHER PARAMETERS

Updating location of config, logs, outputdata or scripts folders

If you have moved the configuration deployment package, this must be reflected in the `batchrunupdate.bat` file.

- Edit the `batchrunupdate.bat` file by right-clicking and choosing “edit”.
- Edit the appropriate parameter(s): paths to the locations of these folders and/or the runupdate.py script. (See sample batch file below.)
- Save and close.

Updating ArcGIS Desktop version (i.e. 10.3 to 10.4)

If you have updated your ArcGIS Desktop version, your Python.exe will be located in a new folder and this must be reflected in the `batchrunupdates.bat` file.

- Locate the Python.exe and note the new path.
- Edit the `batchrunupdate.bat` file by right-clicking and choosing “edit”.
- Change the first parameter in the script, i.e. `C:\Python27\ArcGIS10.3\Python.exe` to the new ArcGIS version location, i.e. `C:\Python27\ArcGIS10.4\Python.exe`
- Save and close.

INTEGRATING DATA

- Aggregate the agency zip files which contain new data into one directory.
- Create an empty directory for the script to unzip the zips to.
- Run the Integrate Data script tool in the Integration Tools toolbox (see Part 2).

PART 1: SOURCE DATA PREPARATION

The functions in the **preparation scripts** are designed to extract applicable data from each data source (agency) and translate into datasets which match the agreed-upon data schemas. The outcome of the preparation stage is a zipped file geodatabase containing all data layers in the integration schema for a single agency.

DEPLOYMENT PACKAGE

A directory structure is deployed in each agency, consisting of four directories, an ArcGIS Toolbox, and a batch file. This location should be permanent and accessible by the machine which will execute the scripts.

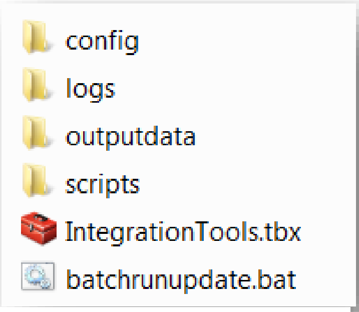

config
logs
outputdata
scripts
IntegrationTools.tbx
batchrunupdate.bat

Figure 1 Deployment package

The *config* folder will contain configuration files (created by the ArcGIS Integration Tools toolbox); the log files will be housed in the *logs* folder; the provided blueprint geodatabase for the source agency is located in the *outputdata* folder, and the scripts are in the *scripts* folder.

REQUIREMENTS

- A computer to execute scripts which has Python 2.7.x+ and ArcGIS 10.2+ installed.
- Deployment package provided by author and housed in a permanent and accessible location.
- Source data.
- ArcGIS applications and Python IDEs should be closed during processing.

SCHEMATIC OF DATA PREPARATION

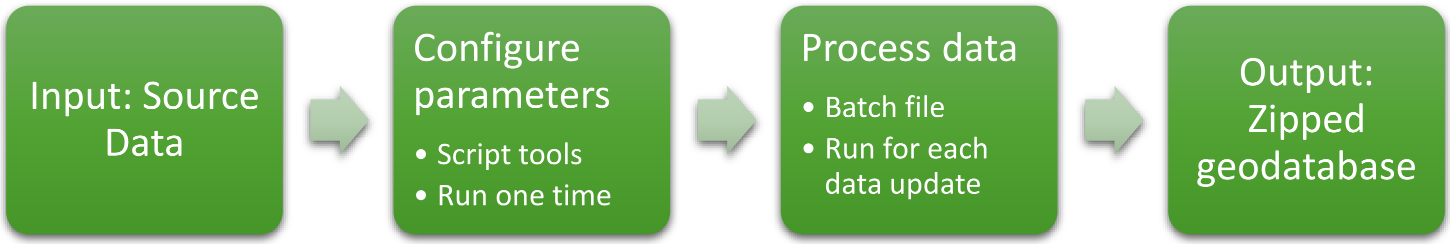


Figure 2 Data preparation

CONFIGURATION SCRIPT TOOLS

The first step is configuration. The Integration Tools toolbox script tools set up parameters for the preparation processing. The toolbox contains a script tool for each feature class that will be updated. Each tool is associated with a configuration script file, as shown in Figure 3. Running the script tool creates a configuration file (such as “structures.ini”) based on the parameters entered in the tool. These parameters define what data will be processed and how.

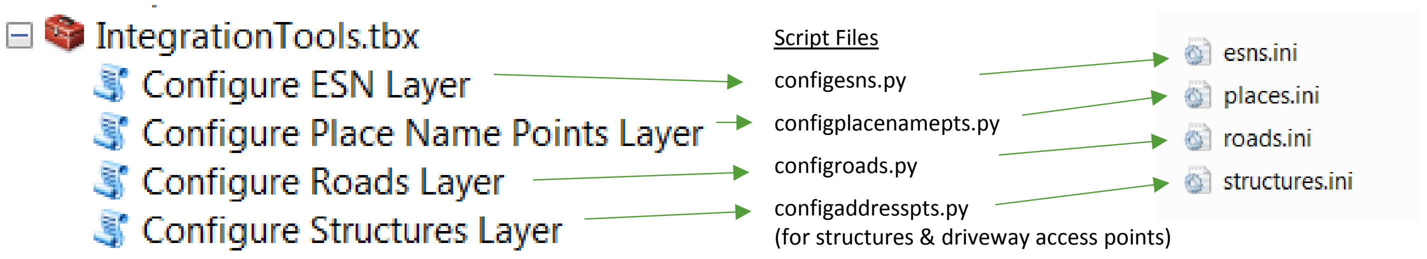


Figure 3 Script tools matched to configuration script file names & *.ini files

The parameters in each tool include:

- **configuration file path and name** (file will be overwritten or created if it doesn’t exist; should be named something like places.ini and must reside in the config folder of the deployment package),
- **data source, target geodatabase and target feature class**,
- **extra query information** (“Additional SQL”, if needed),
- **datum transformation**, and
- **field mapping** of schema fields to source fields (see following page).

There are defaults filled in for field map values that match the integrated schema. Mismatches with the source data fields will be indicated with red “x”s (see Figure 6). In this case, either the default text can be deleted if there is no match in the source data, or the correct field name from the source data chosen from the drop-down list.

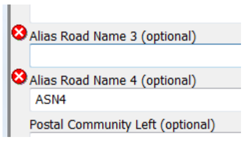

Alias Road Name 3 (optional)
Alias Road Name 4 (optional)
ASN4
Postal Community Left (optional)

Figure 4 Mismatched fields

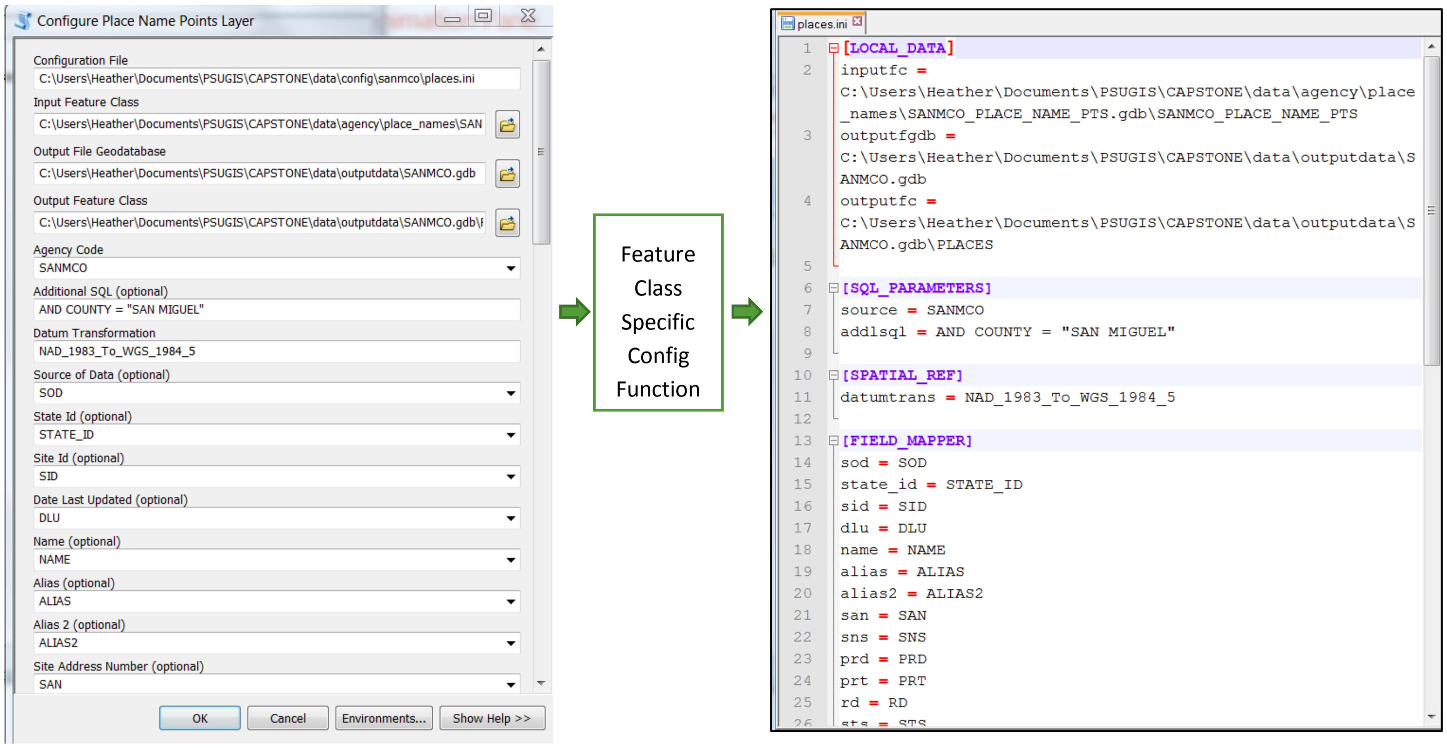
NOTE

If any of the input parameters change (e.g. moved or renamed data, changed source data field names), simply run the script tool for the applicable layer again to reset the parameters.

NOTE

*If changes are made to the **schema field names** or if schema fields are **added or deleted**, various values must be manually edited in each feature class specific configuration file (i.e. configroads.py). Also, default values in the configuration script tools’ properties must be changed or parameters added or deleted.*

CONFIGURATION SCRIPT TOOLS EXAMPLE


Configure Place Name Points Layer
OK
Cancel
Environments...
Show Help >>
Feature Class Specific Config Function
places.ini

Configuration File
C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\config\sanmco\places.ini
Input Feature Class
C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\agency\place_names\SAN
Output File Geodatabase
C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\outputdata\SANMCO.gdb
Output Feature Class
C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\outputdata\SANMCO.gdb\
Agency Code
SANMCO
Additional SQL (optional)
AND COUNTY = "SAN MIGUEL"
Datum Transformation
NAD_1983_To_WGS_1984_5
Source of Data (optional)
SOD
State Id (optional)
STATE_ID
Site Id (optional)
SID
Date Last Updated (optional)
DLU
Name (optional)
NAME
Alias (optional)
ALIAS
Alias 2 (optional)
ALIAS2
Site Address Number (optional)
SAN

```
[LOCAL_DATA]
inputfc = C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\agency\place_names\SANMCO_PLACE_NAME_PTS.gdb\SANMCO_PLACE_NAME_PTS
outputfgdb = C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\outputdata\SANMCO.gdb
outputfc = C:\Users\Heather\Documents\PSUGIS\CAPSTONE\data\outputdata\SANMCO.gdb\PLACES

[SQL_PARAMETERS]
source = SANMCO
addlsql = AND COUNTY = "SAN MIGUEL"

[SPATIAL_REF]
datumtrans = NAD_1983_To_WGS_1984_5

[FIELD_MAPPER]
sod = SOD
state_id = STATE_ID
sid = SID
dlu = DLU
name = NAME
alias = ALIAS
alias2 = ALIAS2
san = SAN
sns = SNS
prd = PRD
prt = PRT
rd = RD
sts = STS
```

Figure 5 Screenshot of example configuration script tool and the *.ini file as written (opened in Notepad++)

Parameters are read in as values, then written out as `[section]` & `key=value` pairs in the associated *.ini file. For field mapping, "key" is the actual schema field name, and "value" is the field name that matches it in the source data. The feature class specific configuration function assigns the pair by the order in which they are read.

BATCH FILE

Description

A batch file is used to initiate running the preparation scripts. This file has been provided to the agencies, in some cases with multiple scripts being executed in succession to perform pre- or post-processing of the source data. The file provides very basic error checking by analyzing the script exit code for runupdate.py. If the script exits with a code other than 0 (a 0 indicates success), the black command window will remain open with a message to check log files. If successful, the command window will close after completion.

NOTE

If the parameters listed in the batch file change, then edit the batch file to reflect the changes. These parameters include the ***location of the Python.exe****, the* ***path & name*** *of script to run, the location of the* ***config folder****, the* ***logs folder*** *and the* ***output data folder****.*

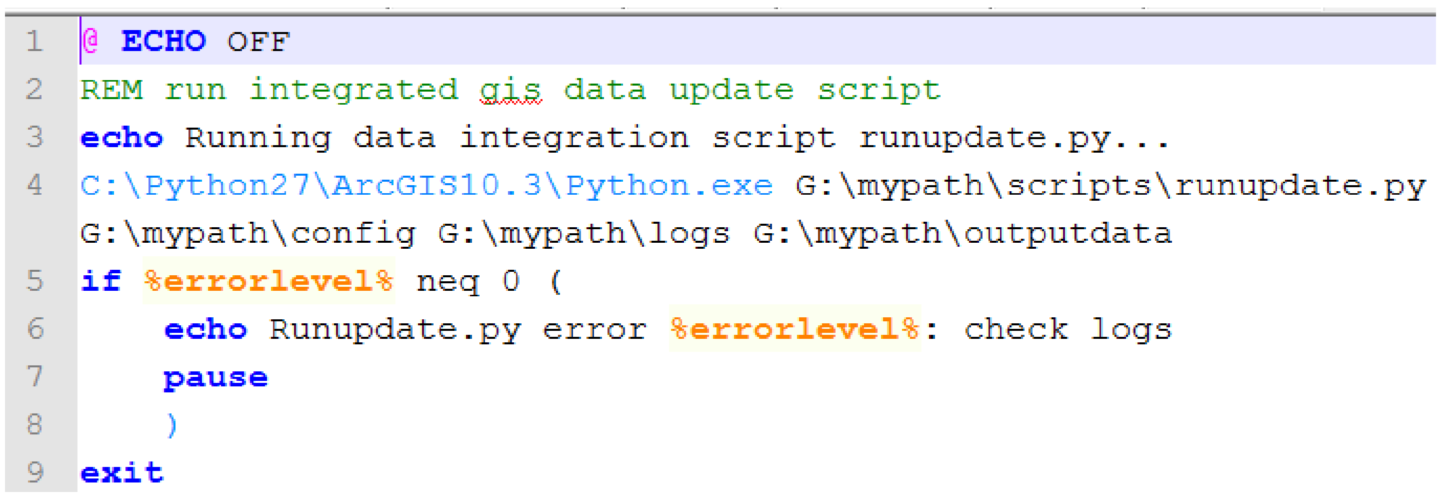
How to edit batch file

- Right-click on file and choose "Edit" OR,
- Right-click and choose "Open With..." Notepad or Notepad++

How to run batch file

- Close any open instances of ArcGIS Desktop and Python IDEs
- Double-click on the *.bat file from Windows Explorer, OR
- Set up to run as a task in Windows Task Scheduler

Example batch file with error messaging



```
@ ECHO OFF
REM run integrated gis data update script
echo Running data integration script runupdate.py...
C:\Python27\ArcGIS10.3\Python.exe G:\mypath\scripts\runupdate.py G:\mypath\config G:\mypath\logs G:\mypath\outputdata
if %errorlevel% neq 0 (
    echo Runupdate.py error %errorlevel%: check logs
    pause
    )
exit
```

Example of running two scripts in succession in a batch file

```
@ ECHO OFF
REM run GUNNCO integrated gis data update script; first run updates, then calc SIDs
echo Running data integration scripts...
C:\Python27\ArcGIS10.3\Python.exe C:\mypath\gunnco_integration\scripts\runupdate.py C:\mypath\gunnco_integration\config C:\mypath\gunnco_integration\logs C:\mypath\gunnco_integration\outputdata
if %errorlevel% neq 0 (
    echo Runupdate.py error %errorlevel%: check logs
    pause
    )

C:\Python27\ArcGIS10.3\Python.exe C:\mypath\gunnco_integration\scripts\gunnco_sid_calc.py C:\mypath\gunnco_integration\outputdata C:\mypath\gunnco_integration\logs
if %errorlevel% neq 0 (
    echo Gunnco_sid_calc.py error %errorlevel%: check logs
    pause
    )
exit
```

In this example, the data processing script (runupdate.py) is called first, and then a separate script is called which populates a unique ID field in the prepared data (SID).

PART 2: REGIONAL DATA INTEGRATION

The functions in the **integration scripts** are designed to process the output from each agency's preparation (a zipped file geodatabase) and combine the data into seamless datasets. The outcome of the integration stage is a file geodatabase containing all data layers in the integration schema for all agencies in the group.

REQUIREMENTS

- A computer to execute scripts which has Python 2.7.x+ and ArcGIS 10.2+ installed.
- A directory of zipped geodatabases to be processed.
- An empty directory to unzip geodatabases to.
- A file geodatabase containing the current version of the integrated data.
- Other instances of ArcGIS applications & Python IDEs should be closed during processing.

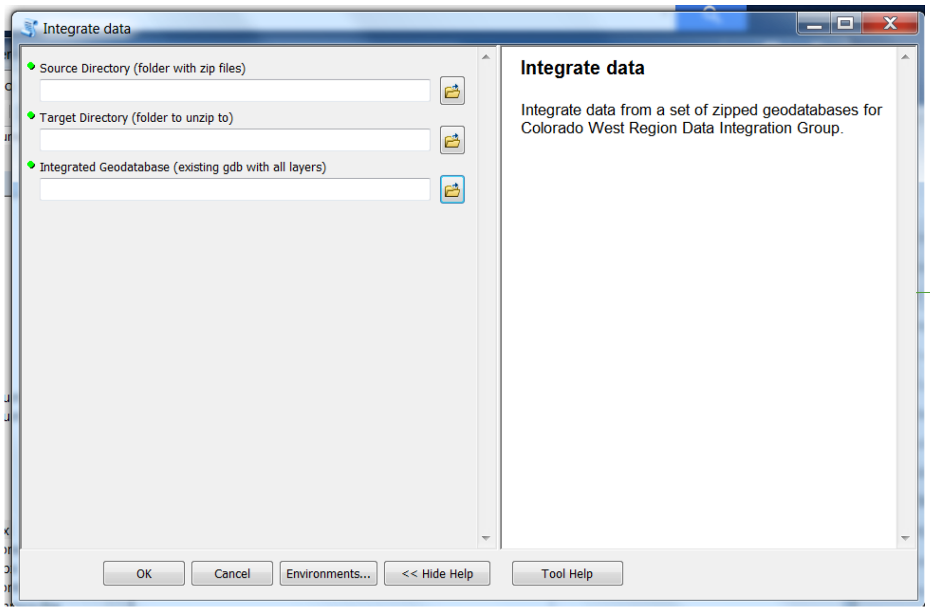
INTEGRATION SCRIPT TOOL

The integration of the preparation output (directory of zipped file geodatabases containing new data) is performed using a script tool. The tool requires an input directory where the zip file(s) are located, an output (empty) directory to unzip them to, and a copy of the existing file geodatabase containing integrated data. The new data will replace the data in the existing file geodatabase. (If only one agency's datasets need to be updated, place that single zip file in the source directory. The other data will remain untouched.)

The **IntegrateData** function used by the script tool cycles through all the zip files in the source directory, extracts them to a directory, deletes the old data in each feature class which match a query based on the new data, and appends the new data.

How to use:

- Close all other instances of ArcGIS Desktop and Python IDEs.
- In ArcCatalog, browse to the location of the Integration Toolbox and open the Integrate Data tool.
- Enter the three parameters in the tool and click "OK".



Integrate data

Source Directory (folder with zip files)

Target Directory (folder to unzip to)

Integrated Geodatabase (existing gdb with all layers)

Integrate data

Integrate data from a set of zipped geodatabases for Colorado West Region Data Integration Group.

Integratedata.py