Appendix: Configuring StateMod Data Sets for the StateMod GUI

7.03.01, 2011-08-31

1. Overview

This appendix describes how to configure StateMod data set files for optimal use with the StateMod GUI. Data sets prepared within Colorado's Decision Support Systems (CDSS) traditionally have been constructed using data management interface (DMI) programs such as TSTool and StateDMI, rather than interactive edits with the StateMod GUI. This facilitates efficient, automated, regeneration of data sets. However, once a data set is constructed, the StateMod GUI is useful for viewing data and results, as well as performing minor edits to data. Consequently, the DMI utilities are useful for data set construction and maintenance, and the GUI is useful for presentation and visual inspection.

The remainder of this appendix describes common configuration tasks that must be performed in order for a StateMod data set to be properly handled by the GUI. It is recommended that these tasks are completed by modelers so that data sets available on the CDSS website are fully configured for users.

These notes were reviewed/tested using the CDSS_WhiteR_09 data set, with some examples from other data sets. Why does the naming convention for data sets seem to change every time (distribution filename, main folder, sometimes data file base name)? Is a standards document being followed?

2. File Naming Conventions

The StateMod GUI does not have file naming restrictions for StateMod data files. However, older versions of StateMod may not recognize files that are in a folder that includes spaces anywhere in the path or has folder/file names greater than X characters. Are there still issues? This used to be an issue with the compiler settings in Lahey.

StateMod data set file naming conventions should follow the guidelines described in the StateDMI **Introduction** chapter (is this the best place for guidelines and are they correct in the StateDMI documentation)?

3. Response File

The StateMod "response file" (**.rsp*) contains a list of all data files used in the data set. The response file typically is edited by hand when creating a new data set using a copy of a response file from an existing data set. Changes also can be made in the StateMod GUI. If a response file has not been configured for the GUI, then the GUI will be limited in its ability to display and edit data files. The format of the response file is `Property = Value`. The following example illustrates GUI properties (note that because a StateMod data set my have multiple response files, the following inserts should be made in each response file):

```
#-----------------------------------------------------------------------
# Used by the StateMod GUI but not by StateMod the model...
Network                                 = ..\network\wm2009.net
GeographicInformation                   = wm2009.gvp
StateModExecutable                      = ..\bin\StateMod-12.29.24.exe
# ... end StateMod GUI items
#-----------------------------------------------------------------------
```

The StateMod software will ignore these properties. Details about the files referenced by the above properties are discussed in the following sections.

4. Network File

Section 3 provided an example of the response file `Network` property to specify the generalized network file. The generalized network file is an XML file that describes the connectivity of the model nodes, as well as providing some layout information for the network editor. This file is not used directly by StateMod but must be consistent with StateMod data files in order for the GUI to properly present information. The standard approach for creating a StateMod data set is to edit the network in StateDMI and generate associated model data files using automated data processing. For example, the list of diversion stations is extracted from the network and is used to query HydroBase for data. Once a data set has "graduated" for use in the StateMod GUI, any changes to the StateMod data set also must have consistent edits to the generalized network. The StateMod GUI enforces such actions through its edit features. For example, when adding a diversion station, the user is asked for the downstream node – the GUI then inserts the node into the network between existing nodes (or above the headwater node in a reach if appropriate). Consequently, the integrity of the network is maintained.

Although the StateMod software does not used the generalized network file, it must be included in the response file to allow the GUI to properly edit the data set, and to allow visualization of model features in the model network editor.

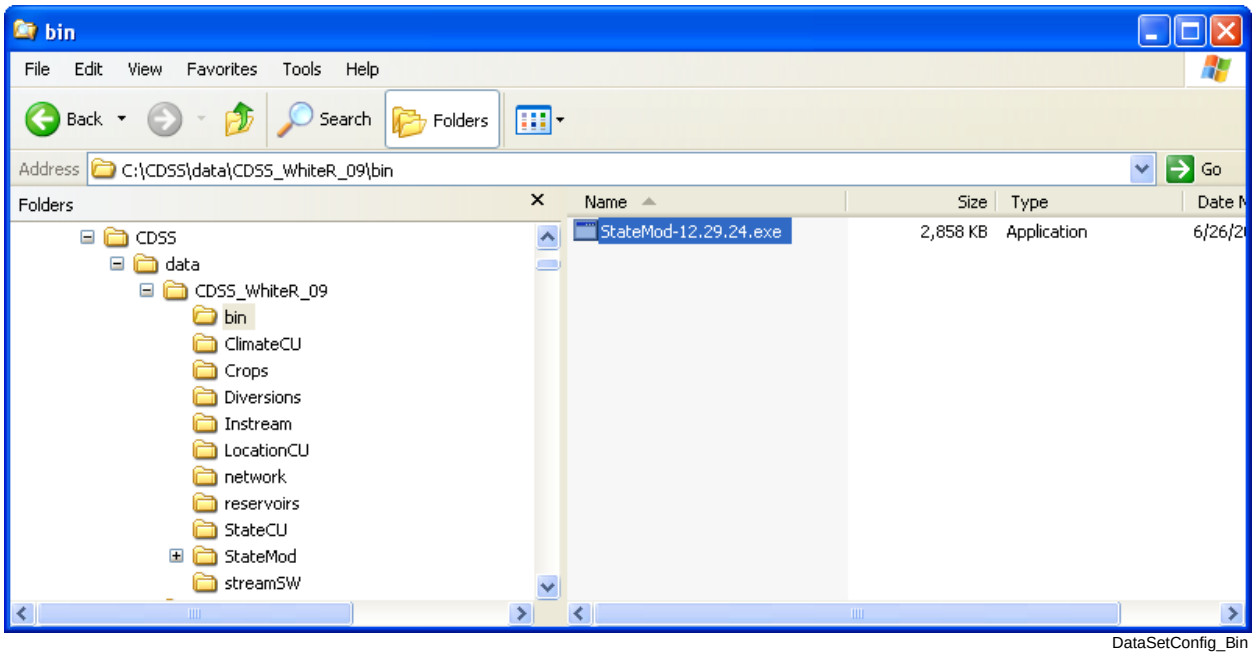
5. Location of StateMod Executable Program

The StateMod executable program can be run on the command line independent of the StateMod GUI, in which case the StateMod program is located in the current folder, the path to the program must be specified on the command line, or the program is found using the `PATH` environment variable.

The StateMod GUI runs the StateMod executable program and displays the standard output (program progress) in the GUI. The location of the StateMod executable program when run from the GUI is determined as follows:

1. Run the StateMod program specified by the `StateModExecutable` property in the response file (see the example in **Section 3**). The location can be relative to the response file folder or can be absolute.
2. Run the StateMod program specified by the `StateModExecutablePath` property in the *system\StateModGUI.cfg* file, located under the StateMod GUI software installation. A recent StateMod executable program will be distributed with the StateMod GUI and will be used by default.

In order to isolate the impacts of the StateMod software version on a data set, it is recommended that a copy of the executable program that is used with a data set is saved with the data set, for example in a *bin* folder parallel to other data set folders ("bin" means binary and is common folder name for executable software):



DataSetConfig_Bin

Example of Configuring StateMod Executable Location for Data Set

This practice isolates the StateMod software used with a data set from other software changes on the system and ensures that the data set, if archived or transported to another computer, will include a compatible version of the StateMod program. Once a location for the StateMod executable is determined, its location can be specified for the GUI as described above.

6. Specifying the Location of StateMod Files

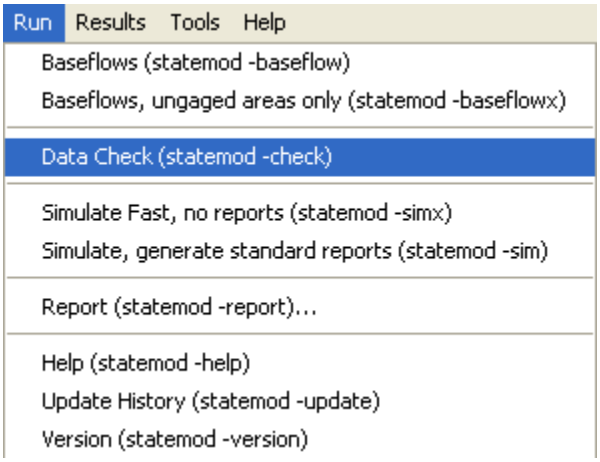
The response file (**.rsp*) contains locations of files that comprise a data set. The location of files generally can be specified relative to the response file. For example, the following line indicates that the control file is in the same folder as the response file:

```
# cm2009.rsp
#
# StateMod Response File
#
# Type                                    Name
# ________________________________________ ____________________________________________
Control                                  = wm2009.ctl
```

The locations of files in other folders also generally can be specified relative to the response file. For example, the following illustrates how data related to the consumptive use model could be specified as residing in the *StateCU* folder that is on the same level as the *StateMod* folder:

```
# cm2009.rsp
#
# StateMod Response File
#
# Type                                    Name
# ________________________________________ ____________________________________________
IrrigationPractice_Yearly                = ..\StateCU\wm2009.ipy
ConsumptiveWaterRequirement_Monthly      = ..\StateCU\wm2009.iwr
StateCU_Structure                        = ..\StateCU\wm2009.str
```

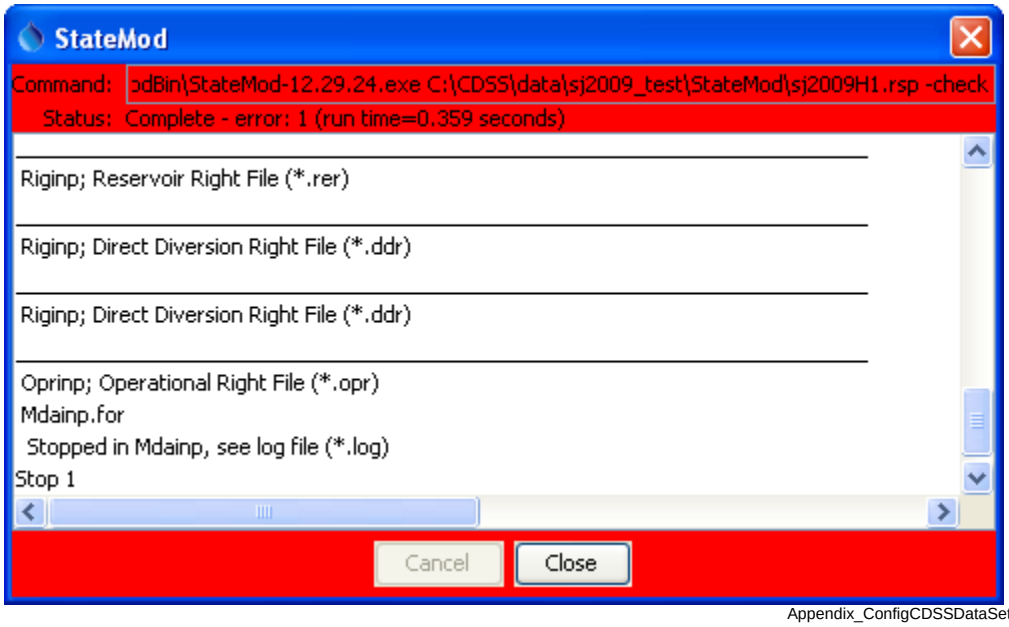
Using relative path locations allows the data set to be self-contained and transportable to other computers that may have different top-level folder organizations. StateMod determines the final file location by concatenating the response file folder with the filename in the response file. This works as expected when SateMod is run from the command line in the same folder as the response file. However, if StateMod is run from the GUI (which itself starts in a different folder than the response file) or StateMod is run from the command line in a folder that is different from the response file, an error may occur because StateMod does not properly concatenate relative paths for all files. The following sequence of actions illustrates detecting a file location problem and implementing a work-around, using the San Juan StateMod data set. After loading a data set in the StateMod GUI, the StateMod data check can be executed as follows:



Appendix_ConfigCDSSDataSet_RspPathError1

Running the StateMod Data Check Module

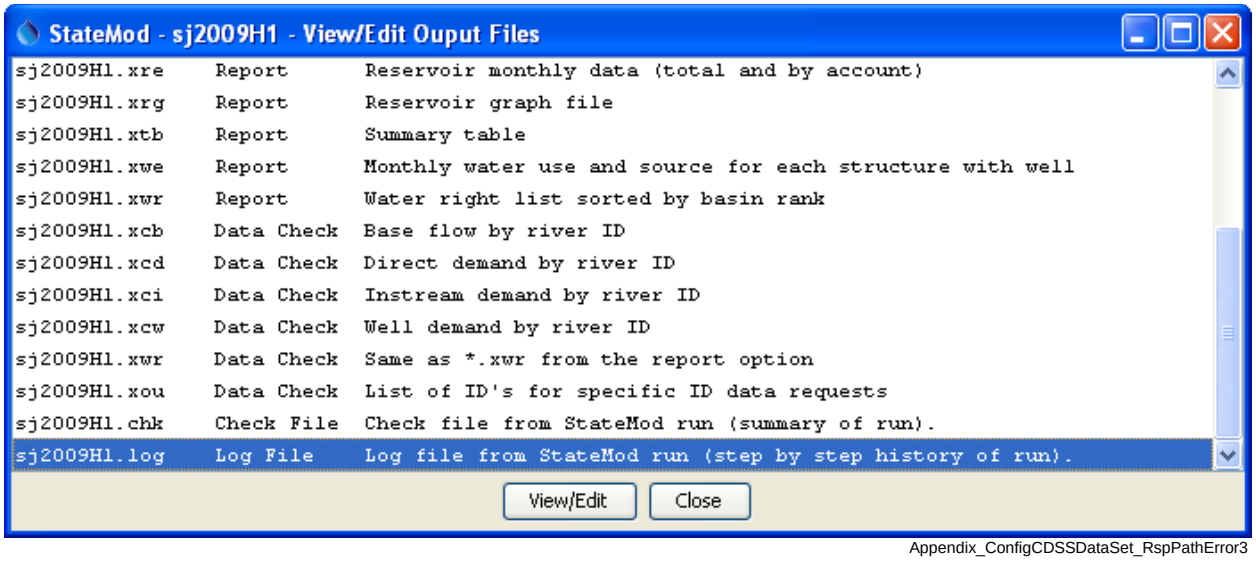
An error may be indicated, as shown below:



Appendix_ConfigCDSSDataSet_RspPathError2

StateMod Data Check Error – StateMod Found an Error

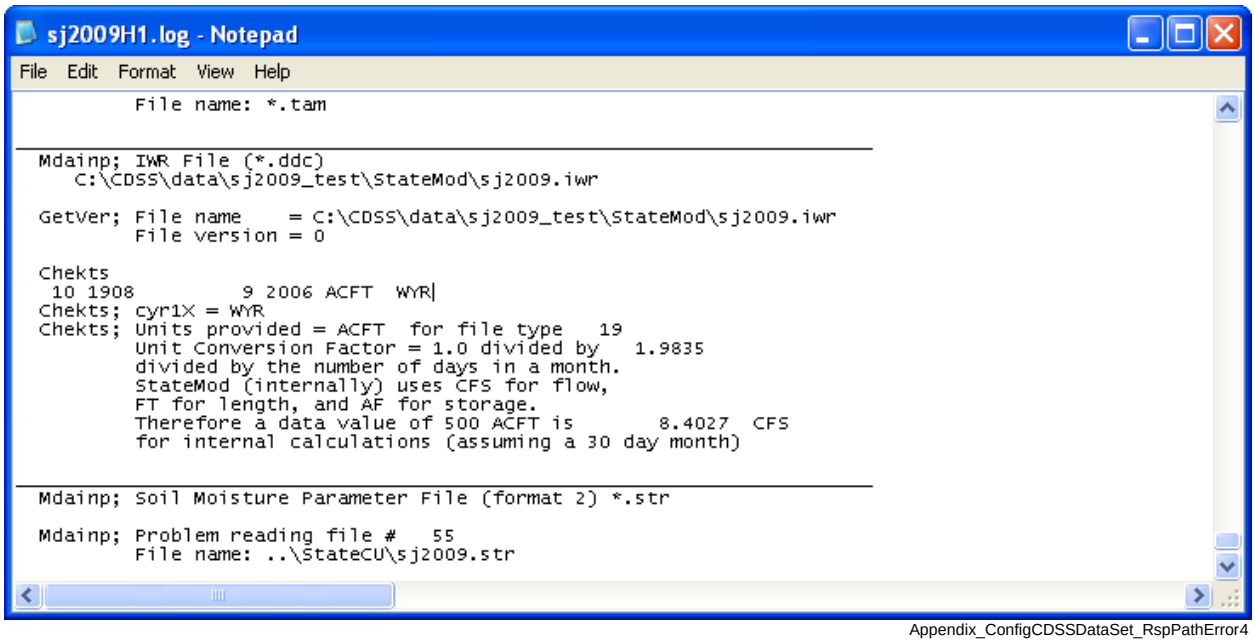
More detailed information about the error will be listed in the StateMod log file, which can be accessed in the GUI using the ***Results…Output File*** menu item (then scroll the file list, select the log file corresponding to the response file, and click on the ***View/Edit*** button) :



Appendix_ConfigCDSSDataSet_RspPathError3

StateMod Data Check Error – Selecting the StateMod Log File to View

The following figure illustrates the contents of the log file. In this case, the Notepad program is used to view the file on Windows and the file has been scrolled to the bottom to show the error.



Appendix_ConfigCDSSDataSet_RspPathError4

StateMod Data Check Error – Viewing the StateMod Log File Contents

The last line indicates that there is a problem reading the StateCU *sj2009.str* file, which has been specified as a relative path in the response file using *..\StateCU\sj2009.iwr*. The warning message is not very descriptive. However, a clue to the problem is that the *sj2009.iwr* file listed near the top of the above figure shows the file name as an absolute path using a combination of the response file folder and the **.iwr* filename in the response file. The StateMod code properly handles a relative path for the **.iwr*

file but not the **.str* file. This is a limitation in StateMod that needs to be fixed in order to allow relative paths for all files to be used when running StateMod from the GUI.

Work-arounds for this limitation include:

1. copy the file from its current location to the data set *StateMod* folder and edit the response file so that a simple filename (without ..\, etc.) is used
    a. CON: multiple copies of a file may result in confusion about which file is the most recent
    b. CON: multiple copies of a file results in a file organization that does not adhere to data set standards
2. use an absolute path in the response file
    a. CON: using an absolute path will make the data set more difficult to transport to other machines or file system locations
    b. CON: an absolute path may result in a specific file unintentionally being used for multiple copies of the data set

Ideally, the StateMod limitation can be resolved in the near future and this section of the documentation can be removed.

7. Spatial Data Files

The StateMod model software does not require spatial data when running a simulation and in fact StateMod data files do not contain fields for spatial data. Instead, spatial information (such as area/precipitation information and network upstream/downstream relationships) are provided by preprocessing data with tools such as StateDMI and TSTool. For example, area/precipitation information is used to create the proration factors in the **.rib* file and upstream/downstream relationships are included in the **.net* and **.rin* files as "downstream ID" for each node.

Spatial data are useful for visualization in the StateMod GUI. Currently, the StateMod GUI does not require that spatial data are specified for a StateMod data set; however, map visualization will be missing or limited without the configuration of spatial data.

The StateMod GUI uses a text properties map definition file that was developed early in CDSS efforts. This map file format and the mapping software do not require licenses such as ArcGIS. However, the mapping tools provide limited geoprocessing capabilities and support limited data formats. In particular, ESRI shapefiles that are commonly used in CDSS are supported; however, background imagery and web services are not. Specify the location of the map file using the `GeographicInformation` property in the response file (see example in **Section 3**).

There are a number of challenges to using spatial data with StateMod data sets, including:

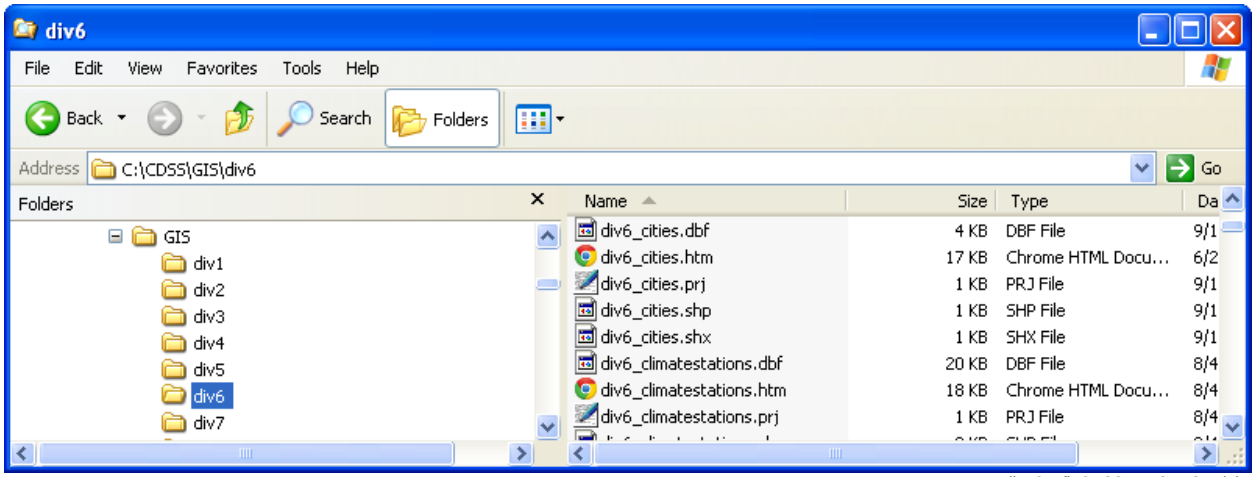
1. CDSS spatial data files available on the CDSS website (*http://cdss.state.co.us/DNN/GIS/tabid/67/Default.aspx*) typically include all features for a data type. For example, the diversion structures layer for a division include all headgates, no matter whether they are included in a data set, and also may include abandoned structures. Consequently, displaying all of the locations in a data set would be overwhelming.
2. Structures/stations that are used in a data set may have inaccurate or missing coordinates. Although correcting the locations in a data set may need to occur, ideally quality control of the data should occur before locations are even used in a data set.

3. It is typical that a data set will include nodes for modeling purposes that do not correspond to physical locations. In these cases, adding the locations to "official" spatial data may be confusing.
4. Mapping software such as ERSI tools can be expensive, complicated, and difficult to maintain, in particular as software versions and data formats change.
5. Although tools to process spatial data generally are available to modelers, some of the above issues require resources that have not been available on modeling projects; consequently, generating data sets with completely defined spatial data has not been a priority.

The compromise that has been implemented in the StateMod GUI is to provide a simple, optional map interface that uses simple data files. Increasingly, it is desirable to move from "optional" to "mandatory" for map visualization. The following information seeks to help preparers of StateMod data sets quickly implement spatial data so that data sets are indeed spatially integrated. Additional software development is needed to fully-integrate spatial data with StateMod data; however, the configuration described below is a step towards this eventuality. Software is only part of the issue – data and modeling standards such as those described in this document must be followed.

The StateMod GUI uses a software component called "GeoView" and configuration of the map uses a "GeoView Project" (**.gvp*) file, which is comparable to the ArcGIS map document; however, the **.gvp* file is human-readable text and is small. The **GeoView Mapping Tools** appendix provides details about how to use the mapping software components. The discussion below focuses on how to set up spatial data for the map.

GIS (spatial) data for CDSS are available on the CDSS website (*http://cdss.state.co.us/DNN/GIS/tabid/67/Default.aspx*) and files traditionally are installed in the *C:\CDSS\GIS* folder, with subfolders for each water division:

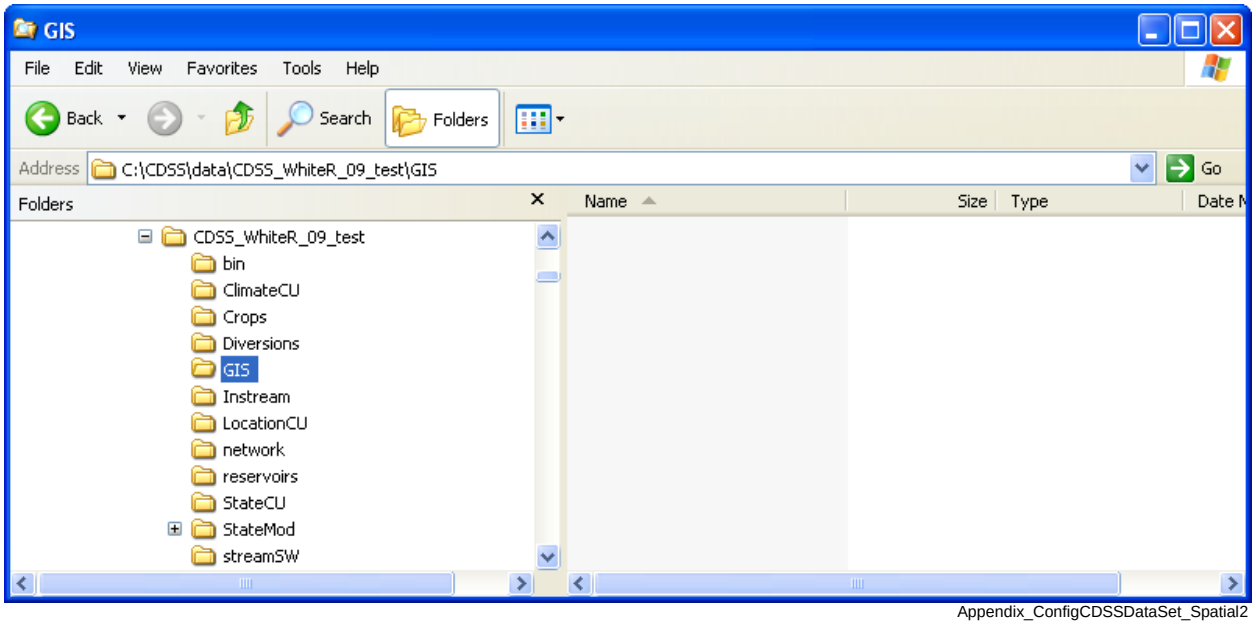


Appendix_ConfigCDSSDataSet_Spatial1

StateMod Spatial Data Configuration – Typical CDSS Spatial Data Folder Organization

This organization facilitates sharing of spatial data files (which can be large) in the same way that HydroBase is used as the master copy of CDSS data. However, similar to how HydroBase data are processed to create StateMod model files, it is useful to copy GIS files to a data set so that they can be "frozen" and are documented as part of the data set. To this end, it is recommended that a *GIS* folder be created in the data set folder, and spatial data files copied to the data set *GIS* folder. Because the *GIS* folder may be distributed with the data set, care should be taken to include only layers that are used in the

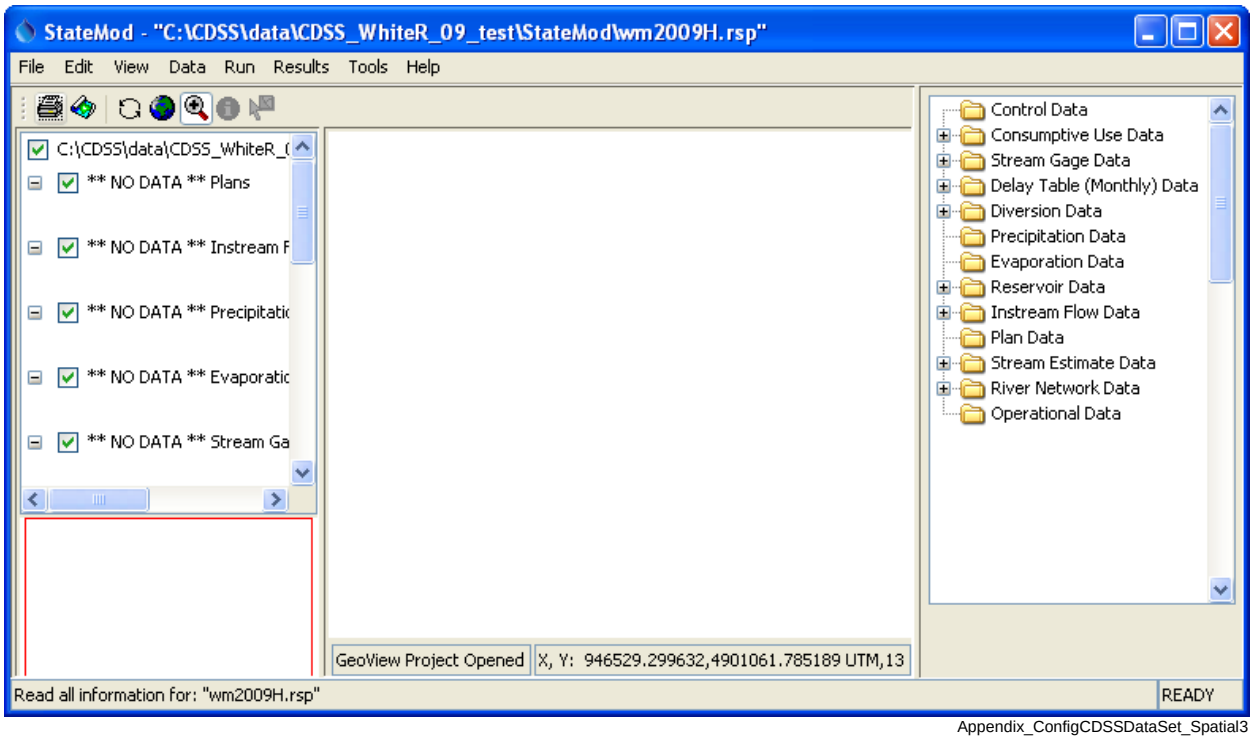
data set (in any way), in order to minimize the size of the data set distribution file. After creating the *GIS* folder, the folders for a data set will be similar to the following:



Appendix_ConfigCDSSDataSet_Spatial2

StateMod Spatial Data Configuration – Initial Spatial Data Folder Organization for Data Set

It is recommended that the **.gvp* file for the StateMod data set be saved in the *StateMod* folder (rather than the *GIS* folder), using a filename consistent with the response file. This ensures that the map is consistent with other StateMod files. If necessary, create more than one **.gvp* file consistent with various response files. Similarly, map configurations for StateCU or other tools can be stored in folders containing the files for those tools. Does this make sense? Or should the standard be to put the **.gvp* file in the GIS folder (if so, what is the naming convention to avoid conflict between different versions of the files)? To facilitate initialization of the **.gvp* file, copy the file from the example data set distributed with the StateMod GUI training materials (or a fully configured StateMod data set) and then edit the file with a text editor. The first step usually is to copy the **.gvp* file, indicate its presence in the response file (see **Section 3**), and try loading the data set in the StateMod GUI. An initial attempt will yield something like the following figure, which indicates that no data can be found. The labels that are displayed in the legend on the left side of the map can be searched for in the **.gvp* file to determine missing files.



Appendix_ConfigCDSSDataSet_Spatial3

StateMod Spatial Data Configuration – StateMod GUI Map with Missing Data Layers

Note that the map interface, when configured by specifying a **.gvp* file, takes over the bulk of the main StateMod GUI window. This is because the map interface is better suited for the main window whereas the network editor is better suited for a separate window.

In order to enable the data layers, edit the **.gvp* file (*wm2009.gvp* in this case), update the names of spatial data files as needed, and copy the needed files from the main *CDSS\GIS* folder to the data set *GIS* folder. Below is an excerpt of the **.gvp* file that is distributed as an example with the StateMod GUI, with highlights to illustrate changes that will need to be made for a specific data set.

```
# GeoView project file for Colorado basin.

# Main GeoView properties.

[GeoView]

# <by default the location is relative to the response file so no need to define the home>
# Main home for data
# If a directory is not specified, the directory will be determined when the
# GeoView project file is selected.
#GeoDataHome = "C:\cdss\data\cm2009\gis"
# ArcView/ArcExplorer Default...
#SelectColor = Yellow
# Arc 8...
#SelectColor = Cyan
# All-purpose (magenta/pink)
SelectColor = "255,120,255"
MaximumExtent = "140000,4313800 450000,4485000" <comment this out until some data are added,
                                                 and then use the mouse cursor in the GUI to
                                                   determine lower-left and upper-right; if
                                                     inaccurate, some features may be clipped>
# Projection for data unless otherwise indicated with data files
Projection = "UTM,13" <for CDSS use this, and use "Geographic" below for specific layers>

# Now list the layer views.  A layer view consists of specifying a data layer
# (e.g., shapefile) and view information (e.g., symbol).  This is equivalent to
# the ESRI "theme" concept.  The layers specified first are drawn on the bottom.
# Start with number 1 and increase the layer number sequentially as layers are
# added on top.
```

```
#
# Important properties are:
#
# GeoLayer - file name for layer data
# Name - Name to show in legend
# SkipLayerView - if true, don't display (but keep in file to maintain layer
#                 numbering)
# AppLayerType - this is "hardcoded" in software to allow recognition in
#                 specific displays.  The value is "softcoded" here to
#                 allow flexibility.
# AppJoinField - attribute field(s) in the GIS data that are used to link to
#                 the StateMod data.  The number of fields must match what
#                 is needed for the AppLayerType (e.g., diversions
#                 require a wd,id).

[GeoLayerView 1] <the layer numbers currently MUST be sequential from 1>
GeoLayer = ../GIS/div5_districts.shp
Name = "Water Districts"
# RGB 153 204 50 - green-yellow
#Color = "0x99CC32"
# tan
Color = "255,240,190"
OutlineColor = black
ReferenceLayer = true
AppLayerType = "BaseLayer" <this indicates how the StateMod GUI should treat the layer>

#<although displaying irrigated lands would be nice, additional software enhancements are needed>
# The following layer is currently skipped.  It can be turned on.  Additional
# classification/symbolization is being developed.
[GeoLayerView 2]
SkipLayerView = true <consequently, skip the layer in the map display>
GeoLayer = ../GIS/div5_irrig_2005.shp
#Name = "Irrigated Parcels"
Name = "Crops"
AppLayerType = "BaseLayer"
# Draw as single color...
#SymbolClassification = "Single"
# Green
#Color = "0,255,0"
# black
OutlineColor = "0,0,0"
# ---- OR draw as unique values -----------
#SymbolClassification = "UniqueValues"
#SymbolClassField = "wd"
#Color = "red;blue;green;yellow;pink;cyan;orange;magenta"
# ---- OR draw as class breaks -----------
SymbolClassification = "ClassBreaks"
#ColorTable  = "Custom;3;red;green;blue"
ColorTable = "YellowToRed,10"
#ColorTable = "BlueToCyan,10"
#SymbolClassField = "wd"
#SymbolClassBreaks = "20,22,24,26,30"
# Try acres for testing
SymbolClassField = "ACREAGE"
SymbolClassBreaks = "1,10,20,40,100,150,200,300,400,500"

[GeoLayerView 3]
GeoLayer = ../GIS/div5_lakes.shp
#GeoLayer = div5_lakes.shp
Name = "Lakes"
# - blue
Color = "165,250,254"
OutlineColor = "0,130,254"
AppLayerType = "BaseLayer"
```

Note that there are global map properties on the top of the file. Because CDSS spatial data are distributed in UTM Zone 13 coordinate system, the **.gvp* file specifies this as the projection and all layers are assumed to be in this projection unless otherwise noted. The StateMod GUI does support geographic (decimal degrees) coordinates to allow layers to be easily added by users (see discussion below related to missing spatial data).

To make the map functional, look for files in the CDSS main *GIS* folder that are similar to those used in the initial **.gvp* file, copy the spatial data files to the data set *GIS* folder, and edit the **.gvp* file accordingly. Note that ESRI "shapefiles" consist of more than one file per layer; therefore, copy the

group of similarly named files. Layers are listed in the **.gvp* with base (background) layers first and topmost layers last. If a GIS layer cannot be found, remove it from the **.gvp* file (renumbering the layers as needed), or set the `SkipLayerView=True` property. After making one or more changes, reload the response file in the StateMod GUI to display changes to the map configuration. After a few iterations, the map should be close to complete and StateMod GUI map features will be enabled accordingly. See below for consideration of specific issues.

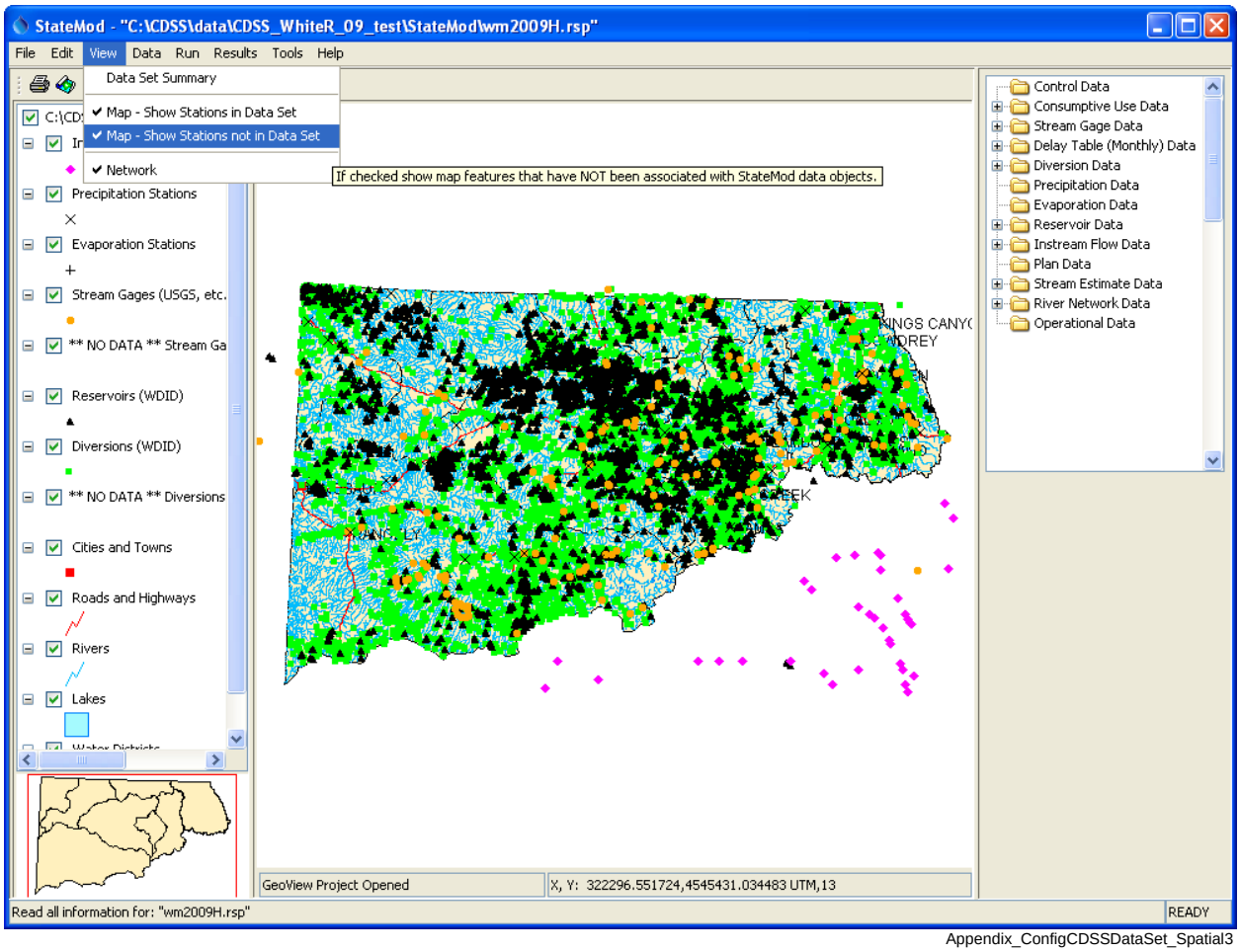
A layer for a single water division may not be sufficient for a data set. For example, a data set may contain streamflow gages from multiple divisions. In this case, data should be copied from multiple *CDSS\GIS* division folders and should be configured in the **.gvp*. Each layer will be displayed separately in the legend and the StateMod GUI will use all available data.

The sample **.gvp* distributed with StateMod uses symbols and colors for each layer. However, cartography is somewhat of an art and there are many standards. Symbols and colors can be changed; however, it is recommended that a "look and feel" be chosen that is similar to other CDSS and state map products. Should we attempt to standardize symbols and colors? Who should define the standard? How should it be published and enforced?

8. Missing Spatial Data

When loading a data set corresponding to a response file, the GUI uses the `AppLayerType` property in the spatial data map file (**.gvp*), to match a layer with a model node type, and the `AppJoinField` to match the identifier in a layer with data set node identifiers. Missing layers, and unmatched identifiers within layers, will result in StateMod data set nodes NOT having a spatial coordinate. Spatial data internally are handled by keeping track of matched features in layers using the StateMod identifiers. The spatial data and StateMod data therefore are loosely coupled. A limitation of this approach is that the StateMod GUI does not force spatial data to be entered when adding nodes to the data set (because the locations are stored in the original data layers). Although this may change in the future, users are required to prepare spatial data outside of the GUI (see the previous section).

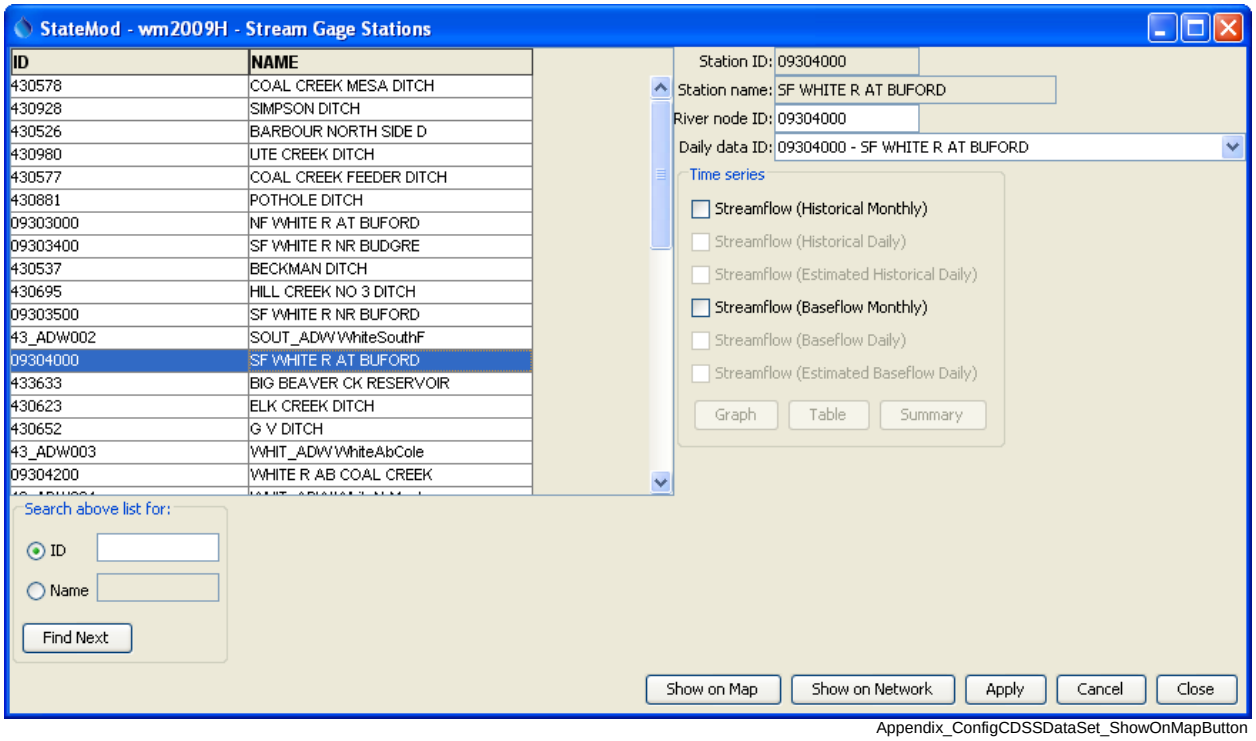
Missing spatial data are a common issue for various reasons (e.g., structures have not been GPS'ed, model nodes do not correspond to physical structures). Although StateMod GUI features will be limited when spatial data are missing, primary features of the GUI will function normally. The following figure illustrates a tool that lets the user visualize features in a layer that have been matched with the StateMod data set, and/or features that have not been matched. The default is that the GUI only shows features that are matched with data set stations.



Appendix_ConfigCDSSDataSet_Spatial3

StateMod GUI Illustrating Matching of Spatial Data to Model Data

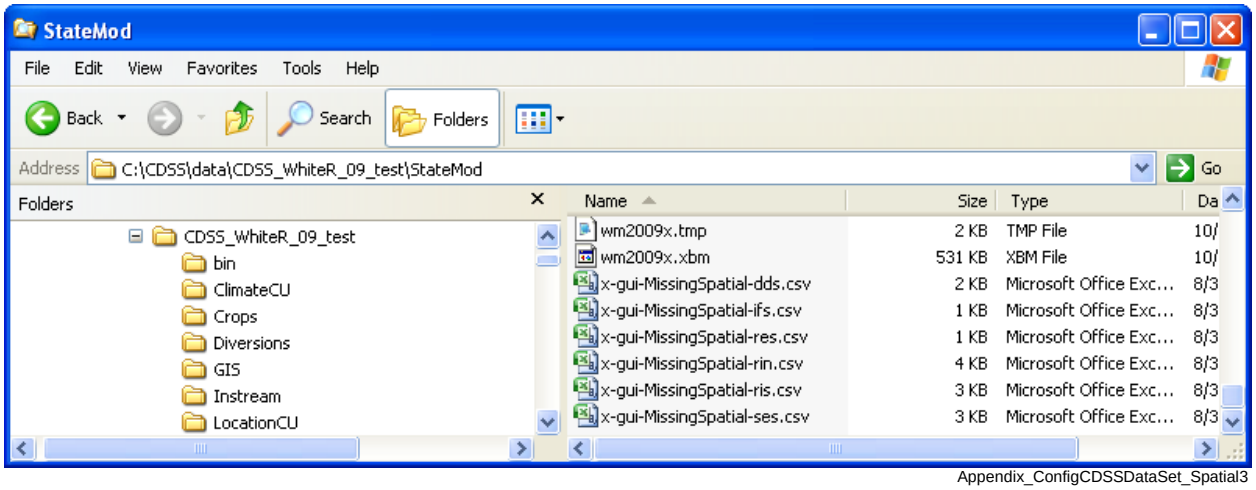
Additionally, if a location is viewed (e.g., using the ***Data…Stream Gage*** menu), the ***Show on Map*** button will be enabled only if the StateMod station has been successfully matched to a feature in a map layer:



Appendix_ConfigCDSSDataSet_ShowOnMapButton

StateMod GUI Map Integration – See "Show on Map" Button at Bottom

In order to help resolve missing spatial data, the GUI automatically creates *x-gui-MissingSpatial-*.csv* files in the data set folder when a data set is loaded, similar to the following:



Appendix_ConfigCDSSDataSet_Spatial3

StateMod GUI Illustrating Matching of Spatial Data to Model Data

A small file size indicates fewer missing coordinates. Each of these files contains comments, a header line to define the names of data columns, and data records in comma-separated value format. For example, the following excerpt is for a diversion stations file:

```
# File generated by...
# program:      StateModGUI 07.03.01 (2009-06-19)
# user:         sam
# date:         Wed Aug 31 10:59:58 MDT 2011
# host:         AMAZON
# directory:    C:\CDSS\data\CDSS_WhiteR_09_test\StateMod
```

```
# command line: StateModGUI -home
#            C:\Develop\StateModGUI_SourceBuild/StateModGUI/test/operational/CDSS
#
# dds has 21 out of 127 locations with missing spatial data.
#
# Specify X and Y in projected coordinates to match other layer data.
#
"ID","Name","Long","Lat","X","Y","Note"
436045,MEEKER WELLS,,,,,"GUI Detected no coordinates"
43_ADW001,NORT_ADW WhiteNorthF,,,,,"GUI Detected no coordinates"
43_ADW002,SOUT_ADW WhiteSouthF,,,,,"GUI Detected no coordinates"
43_ADW003,WHIT_ADW WhiteAbCole,,,,,"GUI Detected no coordinates"
43_ADW004,WHIT_ADW WhiteNrMeek,,,,,"GUI Detected no coordinates"
43_ADW005,WHIT_ADW WhiteNBLMee,,,,,"GUI Detected no coordinates"
43_ADW006,WHIT_ADW WhiteAbPice,,,,,"GUI Detected no coordinates"
43_ADW007,PICE_ADW Upper,,,,,"GUI Detected no coordinates"
43_ADW008,PICE_ADW PicCrBlRioB,,,,,"GUI Detected no coordinates"
```

These files can be used to troubleshoot why locations are not being matched. In the example above, the identifier 436045 corresponds to a structure from HydroBase that has no coordinates. The other identifiers are model-specific locations that do not match HydroBase identifiers.

The coordinates for each of the indicated points should be determined using any available mapping software, and the missing values in the file can be filled in with a text editor. It is recommended that longitude and latitude be specified in any case using the WGS84 projection (commonly used in many online mapping tools). The `X` and `Y` coordinates should be specified as UTM zone 13 for CDSS data or as duplicates of the `Long` and `Lat` values.

Next, save the updated **.csv* file in the data set GIS folder, for example with the name *dds.csv* or *StateMod-dds.csv*. Perhaps we need to provide some guidelines for this in data set guidelines.

Finally, add a layer reference to this file in the **.gvp* file similar to the following:

```
[GeoLayerView 11]
GeoLayer = ../GIS/dds.csv
Projection = "Geographic"
Name = "Diversions (Aggregate/System)"
# Green
Color = "0,255,0"
SymbolStyle = "Square-Filled"
SymbolSize = 5
AppLayerType = "Diversion"
AppJoinField = "ID"
#LabelField = "ID"
```

The mapping software will automatically use the `X` and `Y` column values from the **.csv* file for the coordinates and will use the `Projection` property that is specified for the layer (or the **.gvp* global projection if not specified).

The end result is that the map shown in the StateMod GUI will have layers in the legend that match the official data layer release for CDSS and a similarly-symbolized layer with user-supplied data. Ideally, missing coordinates for structures in HydroBase can be corrected, in which case the list of locations in the **.csv* file will eventually only contain model-specific identifiers. In the future the StateMod GUI may be enhanced so that spatial data coordinates can be specified in the GUI and automatically are saved to the GIS layer files. However, the guidelines described above involving simple **.csv* files are relatively easy to follow in the short term.