Documentation and Analysis of a Geographic Information System Application for Combining Data Layers, Using Nonpoint-Source Pollution as an Example

In cooperation with the
Indiana Department of Environmental Management

Open-File Report 00–324

U.S. DEPARTMENT OF THE INTERIOR
U.S. GEOLOGICAL SURVEY

U.S. DEPARTMENT OF THE INTERIOR
U.S. GEOLOGICAL SURVEY

Documentation and Analysis of a Geographic Information System Application for Combining Data Layers, Using Nonpoint-Source Pollution as an Example

By James L. Kiesler, Jr.

In cooperation with the
Indiana Department of Environmental Management

Open-File Report 00–324

Indianapolis, Indiana
2002

U.S. DEPARTMENT OF THE INTERIOR
GALE A. NORTON, Secretary

U.S. GEOLOGICAL SURVEY
Charles G. Groat, Director

For additional information write to:
District Chief
U.S. Geological Survey
5957 Lakeside Boulevard
Indianapolis, IN 46278-1996

Copies of this report can be purchased from:
U.S. Geological Survey
Branch of Information Services
Box 25286
Federal Center
Denver, CO 80225-0286

CONTENTS

CONTENTS—Continued

CONTENTS—Continued

FIGURES

CONTENTS—Continued

FIGURES—Continued

CONTENTS—Continued

FIGURES—Continued

TABLES

Documentation and Analysis of a Geographic Information System Application for Combining Data Layers, Using Nonpoint-Source Pollution as an Example

By James L. Kiesler, Jr.

ABSTRACT

A geographical information system application has been developed that allows scientists to combine multiple data layers into a single data layer. This application provides an effective tool for identifying areas where the potential effect of the combination of data layers may be greater than any single data layer. Such a tool is useful in studying an activity that cannot be measured directly. Scientists wanting to identify areas where an activity may have the greatest effect can identify factors that directly or indirectly reflect the effect of the activity being studied. When combined, these factors would identify areas where the potential for the activity to have an effect are greatest.

The data layers used to develop the single data layer determine the activity addressed by the application—the application was developed to identify areas where the potential for nonpoint-source pollution to affect areas of Indiana is greater relative to other areas in Indiana. To evaluate the potential in other states or areas, data layers for those states or areas would be used. To address a different activity, even activities not related to water resources, data layers that directly or indirectly reflect the effects of the activity being studied would be used in the application.

The application was developed using Environmental System Research Institute's ArcView geographical information system and the ArcView extension, Spatial Analyst. To use the application, the user selects data layers related to the activity being studied, describes the variability of a data element between geographic areas on each data layer, assigns a relative importance factor to each data layer, and focuses the output by identifying a watershed of interest. The application then assigns a rank value to the features within each data layer on the basis of the spatial variation of the data layer. The data layers are converted to grids. The values of the grid cells are the product of the rank values and the importance factors. The cells for each data layer are combined to form a single grid. The combination is summarized to describe the spatial variation for the watershed of interest. The combined data layer does not show the actual potential effect of the selected factors but rather the relative difference in the potential effect among areas when all data layers are considered.

An analysis of the application indicates that the selected data layers to be combined should be at the greatest spatial resolution possible; however, all data layers do not have to be at the same spatial resolution. The spatial variation of the data layers should be adequately defined. The size of each grid cell should be small enough to maintain the spatial definition of smaller features within the data layers. The most accurate results are shown to occur when the values for the grid cells representing the individual data layers are summed and the mean of the summed grid-cell values is used to describe the watershed of interest.

INTRODUCTION

The analysis of spatial data has progressed significantly with the growth of geographic information systems (GIS). In many ways, however, the technique of analysis has remained virtually the same, especially for users who have difficulty with computers. Scientists overlay data layers (that is, maps of information), mentally combining the multiple data layers and analyzing the results. This technique works well when the scientist is analyzing small areas or when the analysis includes only a few data layers. This technique also allows the scientist to use scientific judgment when combining the data layers; however, the scientist can assimilate only so much information before something is likely to be missed. A GIS application was developed to automate the combining of data layers.

The use of this GIS application is limited to the information contained in the selected data layers. As an example, fishery managers can use the application to help identify potential areas for stocking. Potentially, data layers showing the location of lakes, camping facilities, motels, boat rentals, fish populations, most recent restocking, and the number of visitors to the area are selected. Combining these layers and examining the differences among areas could identify areas where the benefit of restocking is likely greater in comparison to other areas. An analysis of the change in these layers over time could identify areas where the potential need for restocking is increasing relative to other areas of the state. As another example, local planners can evaluate the potential effects of urban expansion by combining data layers related to land use, sewer and water lines, soils suited to the use of septic tanks, and transportation corridors.

The application developed by the U.S. Geological Survey (USGS), in cooperation with the Indiana Department of Environmental Management (IDEM), is described in this report. The application uses rank values and grids of existing data layers to develop a single data layer representing the combination of many data layers, thereby freeing scientists to focus on evaluating why one area differs from another. The report serves three purposes. First, it documents the use of the application. Second, it evaluates the results of the application and how various user selections can affect the output. Finally, the report describes the work performed by the IDEM's Prioritization of Water Bodies Subcommittee, Nonpoint Source Management Plan Task Force, as an example of the techniques used in the application.

DOCUMENTATION OF THE APPLICATION

This GIS application allows users to select any number of data layers, called "themes" in ArcView, describe the spatial variation of the data represented by each theme; and then create a new theme that represents the combination of the selected themes. The application is written in Environmental Systems Research Institute's (ESRI) ArcView[1] "Avenue" script language. An object-oriented language, "Avenue" allows the programmer to customize the graphical user interface to ArcView by modifying how the ArcView application window looks and responds to a user's input by removing selected functions or by adding functions. Although the application takes advantage of ArcView's functionality, this report does not attempt to discuss the full range of ArcView's functionality. Rather, this report presents the ArcView functions needed to execute the application and the functionality added by that application. Use of this application requires some understanding of ArcView.

The first step in developing the application was to create a database. This database consists of updated versions of the data layers developed by the IDEM's Prioritization of Water Bodies Subcommittee. The Subcommittee had created ArcInfo coverages for each data layer. The USGS converted the coverages to ArcView themes. A theme is stored in a shapefile format. The shapefile format consists of several data files with the following naming extensions:

[1]ArcView GIS GUI is the intellectual property of ESRI and is used herein with permission. ArcView is a registered trademark and the ArcView magnifying glass logo is a trademark of ESRI.

- .dbf—a dBase file containing the attribute information,
- .shp—the file containing the feature geometry,
- .shx—the file containing an index of the feature geometry,
- .sbn and .sbx—files containing a spatial index of feature geometry.

The information contained in a theme's attribute file determines whether a theme represents one or more data layers. Themes are displayed in the ArcView document type "View." A file that documents the source and resolution of the data, the map projection, the map units, and the distance units is included with each theme. The name of this documentation file follows the naming structure—theme name underscore "README.txt" (for example pop_README.txt).

Application Installation

The application requires access to ESRI's ArcView and the ArcView extension "Spatial Analyst." The project file "ws_scn.apr" should be copied to the user's work area. The directory "wss_themes" should be copied to disk space to which the user has access. Three system environment variables must be established: "EV_HOME" identifies the user's work area and "EV_DATA" identifies the directory containing the themes from "wss_themes." "EV_TEMP" is a temporary area where the results of intermediate calculations are temporarily stored. If the system variable "TEMP" or "TMP" has been initialized, "EV_TEMP" should be set to the same area. If neither system variable has been initialized, set "EV_TEMP" to the directory of choice.

Starting the Application

To initiate this application, the user must have access to ArcView and the ArcView extension, "Spatial Analyst." The application is stored as an ArcView project. To start the application, the user either issues the command "ws_scn.apr" or starts ArcView and then locates and executes the project "ws_scn.apr." Issuing the command "ws_scn.apr" will start ArcView and the application.

When ArcView is executed, the ArcView application window is displayed (Figure 1). This window will look just as it did when the project was last saved. The ArcView application window consists of a main viewing area where individual documents are displayed and the Project window. The Project window ("ws_scn.apr," Figure 1) shows the various document types (Views, Tables, Charts, Layouts, and Scripts) and the document names for the selected document type. Figure 1 has the View's document type selected and shows the names of three views, "Full View," "Opening View," and "Screening One." The appearance of the ArcView application window depends on the type of document that is active in the application window. A document is active when the title bar at the top of the window is blue. Figure 1 shows the ArcView application window when the Project window is active, and Figure 2 shows the window when a view is active. Figure 2 shows the title bar is blue, indicating the "Opening View" document is active. The title bar is the area at the top of the window containing "Opening View." The title bar for the Project window (ws_scn.apr) is gray, indicating the window is not active.

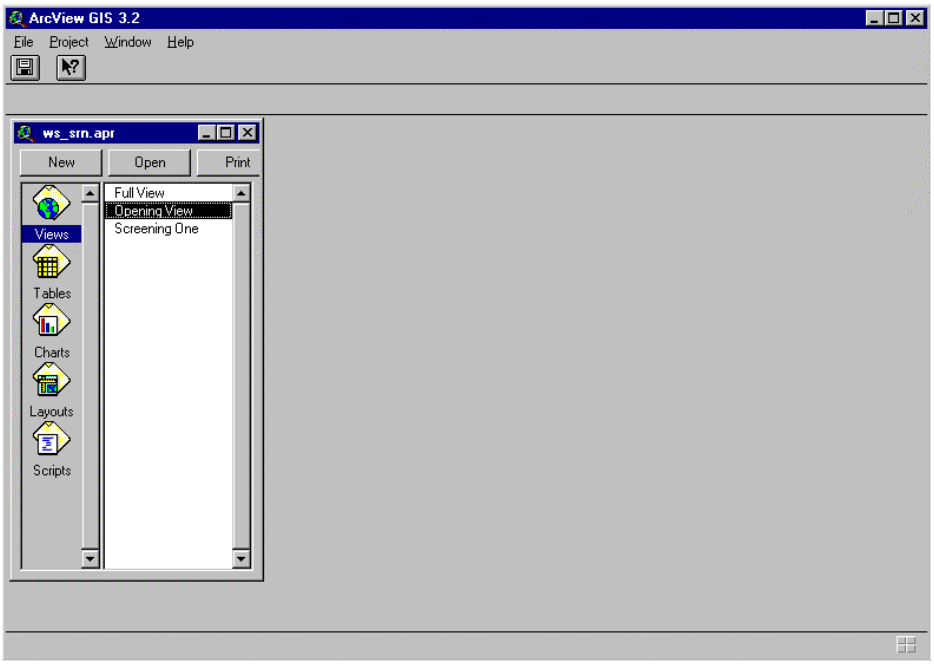


Figure 1. ArcView application window showing the ArcView Project window.

The ArcView application window shown in Figure 1 and Figure 2 contains the title bar, menu bar, button bar, and tool bar. The title bar is the blue bar containing the phrase "ArcView GIS 3.2." The content of the three other bars change with the type of document that is active. Figure 1 shows the menu, button, and tool bars when the Project window is active. Figure 2 shows the menu, button, and tool bars when a View window is active.

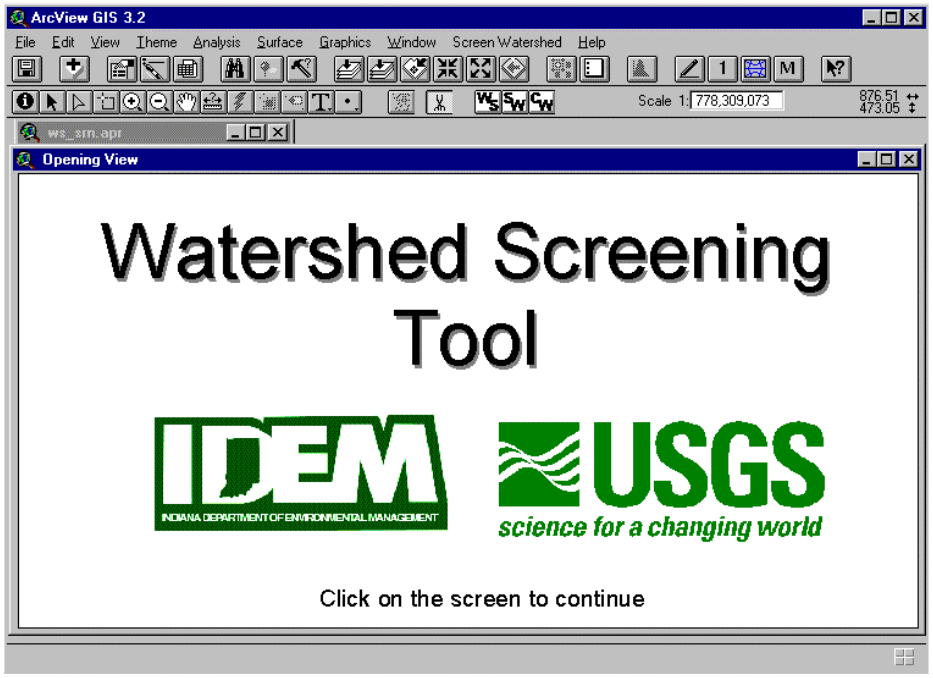


Figure 2. ArcView's application window showing the "Opening View" for the Watershed Screening Tool.

The menu bar is immediately below the title bar and almost always starts with the word "File." Each entry accesses a pull-down menu whose entries are used to access ArcView's functions. The "Screen Watershed" menu entry is used to access the functions to combine data layers.

The button bar, immediately below the menu bar, gives access to the more common functions accessed through the menu bar. The "pencil" button, fifth from the right in Figure 2, and the next three buttons access the functions used to screen a watershed. These functions are (1) "create a theme" (the pencil button), (2) "weight factors" (the one button), (3) "screen watersheds" (the blue spider button), and (4) "make a map" (the M button).

The tool bar is immediately below the button bar. The "WS," "SW," and "CW" tools were added as part of this application. The "WS" tool is used to select the watershed of interest. The "SW" button is used to split a watershed, and the "CW" button is used to cut a piece from an existing watershed. The remaining menus, buttons, and tools are standard for ArcView with the "Spatial Analyst" extension.

When the application is started, the ArcView application window is displayed (Figure 2). This window contains the View named "Opening View." By clicking within this window, "Watershed Screening Tool" starts and looks as it did when this application was last closed. The user can open a view or create a new view by using ArcView's "New" or "Open" buttons on the Project window ("ws_scn.apr" in Figure 3).

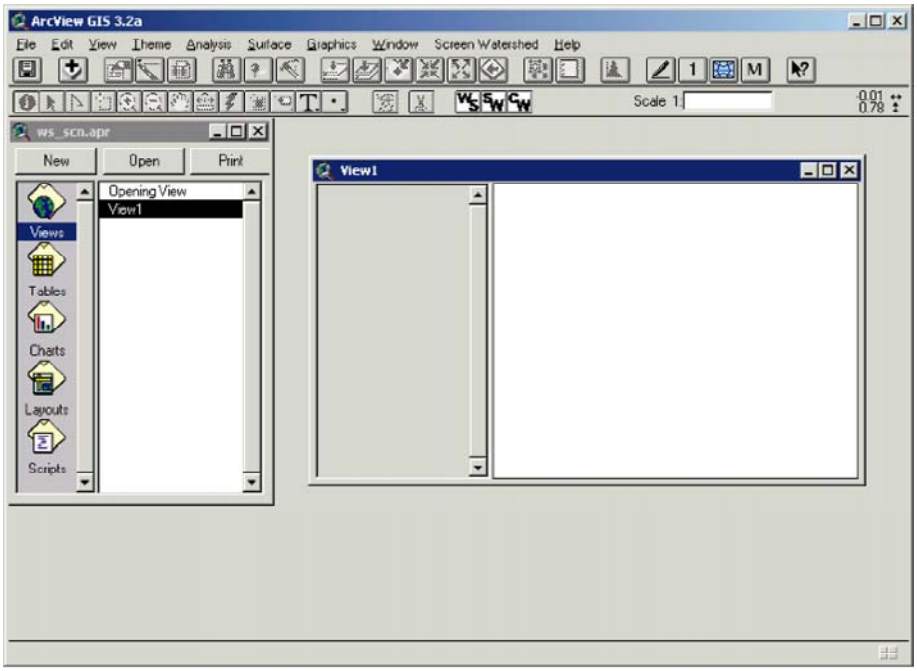


Figure 3. ArcView's application window showing an empty view named "View1."

The ArcView application window, shown in Figure 3, shows the Project window "ws_scn.apr" and the empty view window "View1." "View1" is the result of clicking on the "New" button in the Project window. ArcView automatically names new Views as "View #," where # is a sequential number, beginning with "1." The user can open existing views by pointing and double clicking on the View name listed in the Project window or by pointing and clicking on the View name and then pointing and clicking on the "Open" button.

The “View Properties” menu, accessed through the “View” pull-down menu (Figure 4), is used to change the View name and other View properties. Other View properties that should be set for the “Watershed Screening Tool” to function properly are the map and distance units associated with the themes contained in the view (Figure 5). Themes in the data base accompanying the application have map units of meters and distance units of miles. The “View Properties” screen contains a “Projection. . .” button; although this functionality allows a projection from degrees of latitude and longitude (geographic map projection) to another map projection system, the function does not allow any reprojection from a nongeographic map projection to another projection.

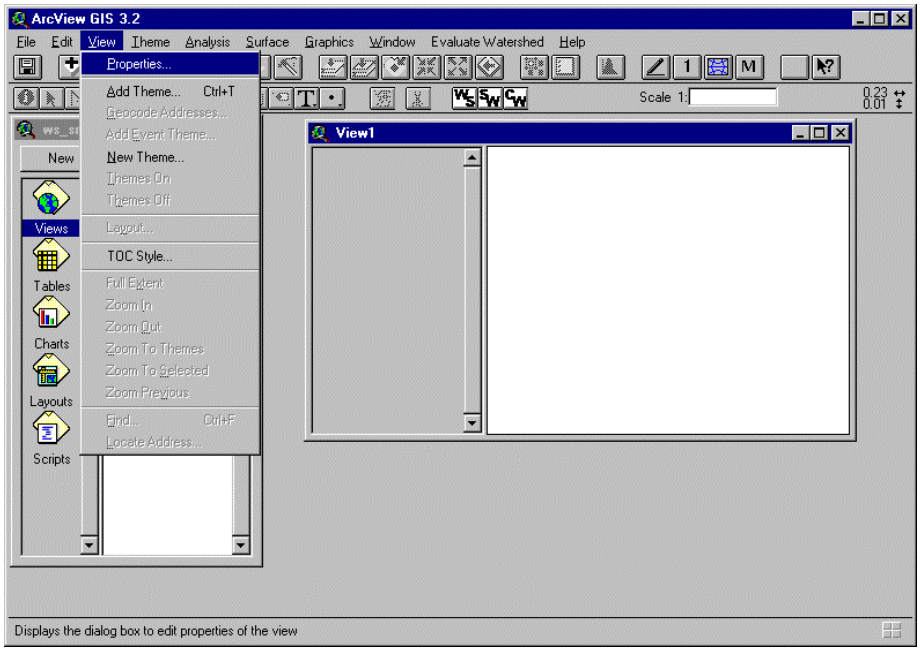


Figure 4. ArcView’s “View” pull-down menu.

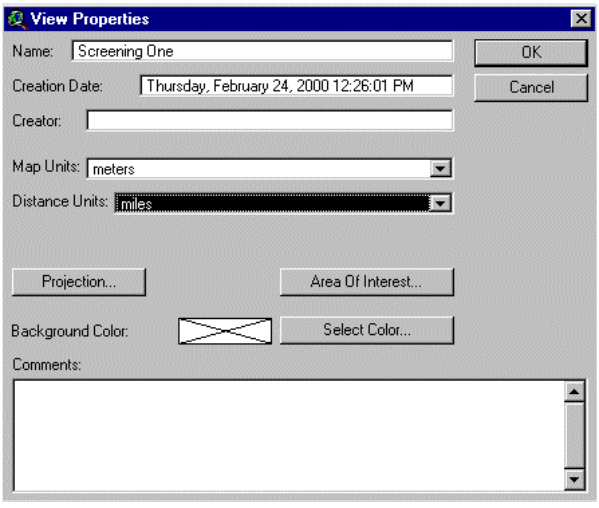


Figure 5. ArcView’s “View Properties” window.

Selection of Data Layers

One of the most critical steps in using this application is the selection of data layers to include in the analysis. The objective of the watershed screening will dictate which data layers are selected. ArcView's "Add Theme" function is used to select the data layers. The "Add Theme" function is accessed using the plus/diamond button (Figure 6) located on the ArcView button bar or the "View" pull-down menu (Figure 4). The "Add Theme" function opens a window (Figure 7) showing the available themes. The first time the "Add Theme" function is executed, "Add Theme" will display the themes in the database accompanying the application. These themes will be in the directory that has been specified with the system environment variable "EV_DATA." Themes in other directories can be accessed using the functions available in the "Add Theme" window. Subsequent executions of the "Add Theme" function will result in the themes contained in the last directory accessed to be displayed.

Pointing and clicking on the theme name selects the theme. Multiple themes can be selected by pointing and clicking on the first theme and, then, holding down the shift key, and pointing and clicking on additional themes. Selected themes will be highlighted. The selected themes are added to the view's table of contents by pointing and clicking the "OK" button. The view's table of contents, which lists the themes contained in the View, is on the left-hand side of the View window ("Screening One," Figure 8).

Figure 6. ArcView's "Add Theme" button located on the ArcView button bar.

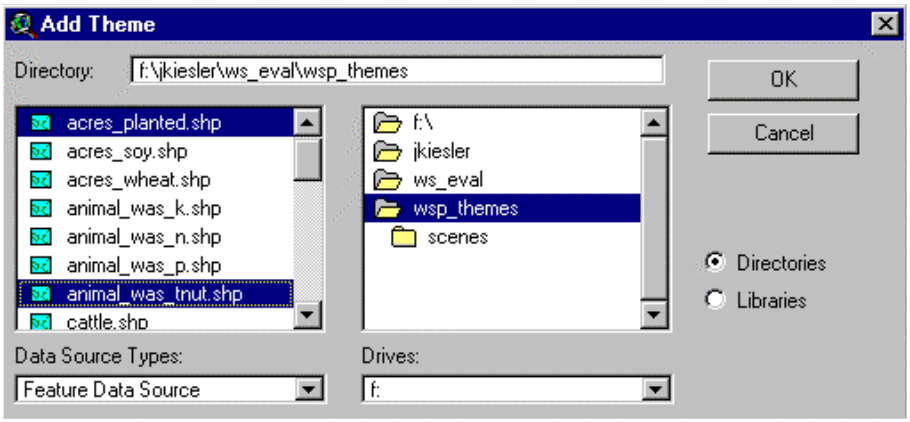


Figure 7. ArcView's "Add Theme" window.

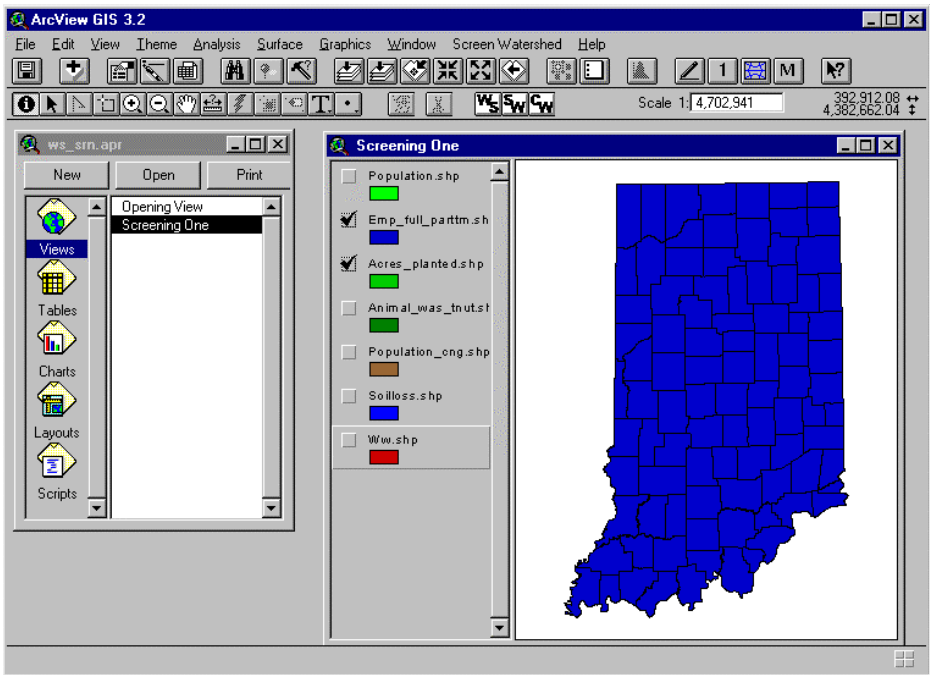


Figure 8. ArcView's application window showing seven themes listed in the Views' "Screening One" table of contents.

ArcView adds themes to the table of contents but does not make the theme visible in the View window. A theme is made visible by pointing and clicking on the box (toggle button) to the left of the theme name in the view's table of contents (Figure 8). The toggle button then will have a "check mark" to indicate the theme is displayed. Themes are added to the top of the view's table of contents. Thus, the theme at the top of the list in the "Add Themes" window (Figure 7) will be at the bottom of the view's table of contents. Themes at the top of the list overlay themes lower in the list. Figure 8 shows two themes are displayed, but only one is visible to the user. Clicking on the theme and dragging it to the desired location in the view's table of contents can change the drawing order of themes. Themes are added to the view by default using a legend type of "Single Symbol," which assigns one color to all features of a theme.

The user must describe the spatial variation of each theme. The spatial variation is described by converting the theme legend from "Single Symbol" to "Graduated Color," using ArcView's "Legend Editor." The "Legend Editor" (Figure 9) is accessed by double clicking on the theme name in the view's table of contents.

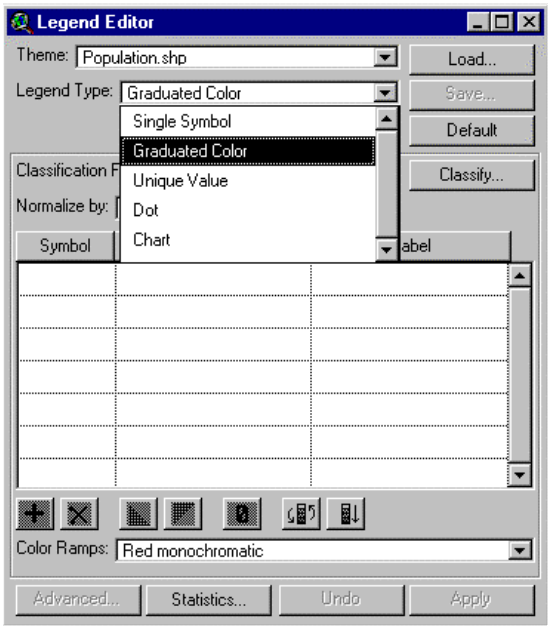


Figure 9. ArcView's "Legend Editor" showing the "Legend Type" drop-down menu.

The "Legend Type" is selected from the "Legend Type" drop-down menu (Figure 9). The "Graduated Color" legend is recommend for this application. The "Graduated Color" option will use the data element specified in the "Classification Field" (Figure 10) and will automatically display the theme using five graduated colors. A data element from the theme's attribute table can be selected to normalize the data ("Normalize by," Figure 11). Normalization is the process of making data values equivalent between features in the theme. To adjust for the size of a county, the county population can be divided by the area of the county. Thus, when a county with greater area is compared to a county with less area, the area of the county is irrelevant. Clicking on the "Classify..." button (Figure 11) allows the partitioning of the selected data element to be further modified. The "Classification" window (Figure 12) offers several methods for defining the class values: equal area, equal interval, natural breaks, quartile, and standard deviation ("Type"). The number of classes the data are partitioned into ("Number of classes," Figure 12) and the number of decimal places to which the class values are rounded ("Round values at," Figure 12) also can be changed. The class values can be entered manually in the "Legend Editor" (Figure 13).

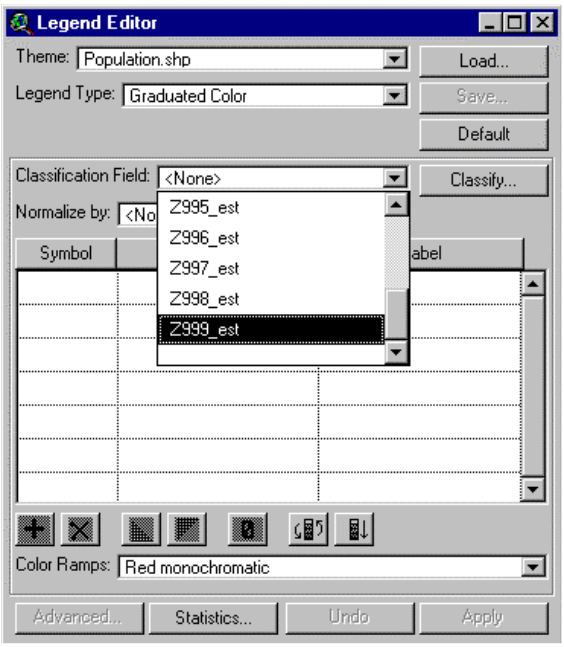


Figure 10. ArcView's "Legend Editor" "Classification Field" drop-down menu for the theme "Population.shp."

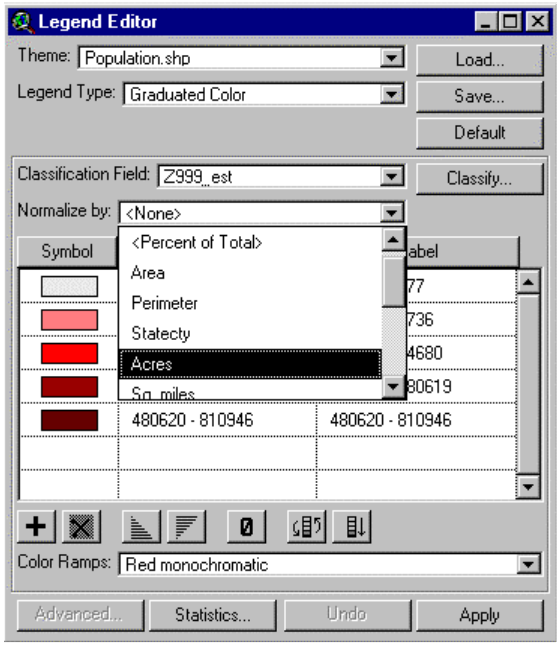


Figure 11. ArcView's "Legend Editor" "Normalize by" drop-down menu for the theme "Population.shp."

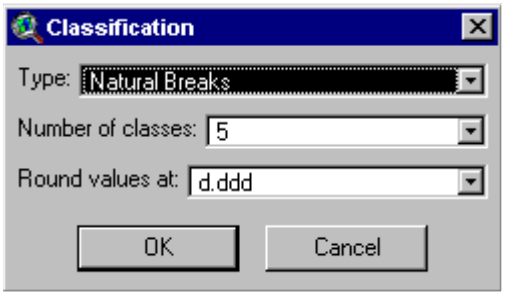


Figure 12. ArcView's "Legend Editor" "Classification" window.

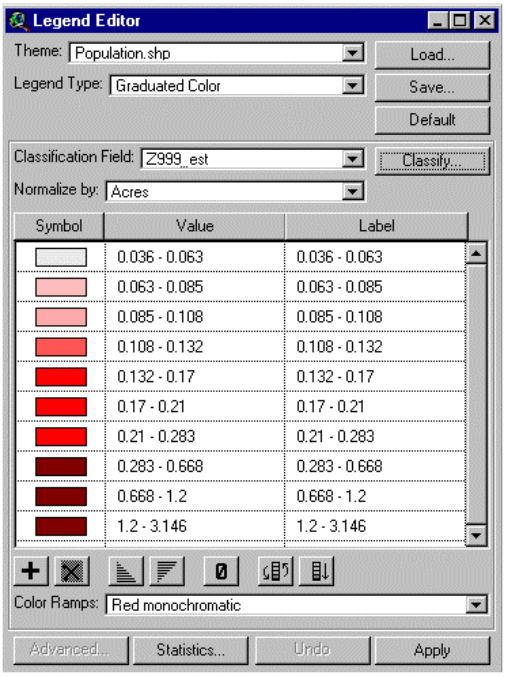


Figure 13. ArcView's "Legend Editor" for the theme "Population.shp" with 10 classes.

Figure 13 shows the "Legend Editor" for the theme "Population.shp" with 10 classes. The population in a county was normalized using the acres in the county. The number of classes and the number of decimal points should be varied to ensure the spatial variation of the data represented by the legend is accurate. Examination of Figure 13 shows the color variation between classes is not distinctive. By double clicking on the "Symbol" entry for a specific class, "Fill" and "Color Palettes" (Figure 14) can be used to modify a pattern to some classes to make those classes more distinctive. Different color ramps also can be specified by using the drop-down "Color Ramp" menu. Figure 15 shows the theme's legend after the patterns have been added, and Figure 16 shows the ArcView application window after the legend entry has been modified. This process should be followed for all themes that will be included in the watershed screening.

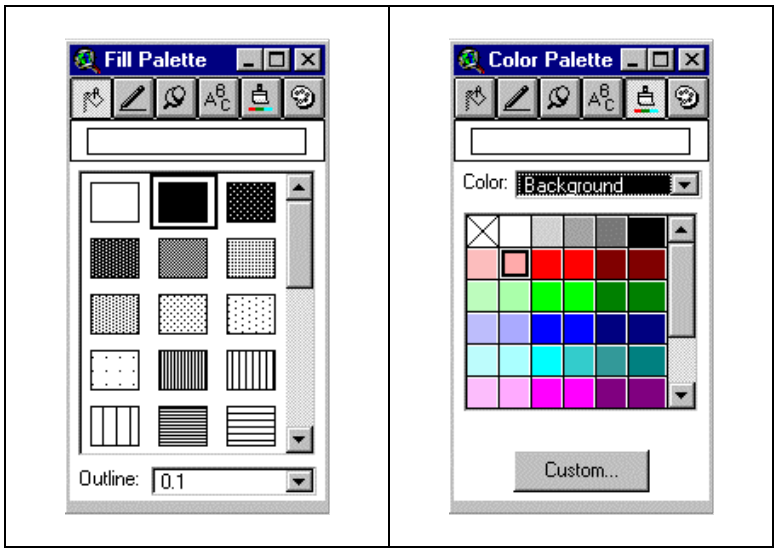


Figure 14. "Fill" and "Color Palettes" used to modify a theme's legend.

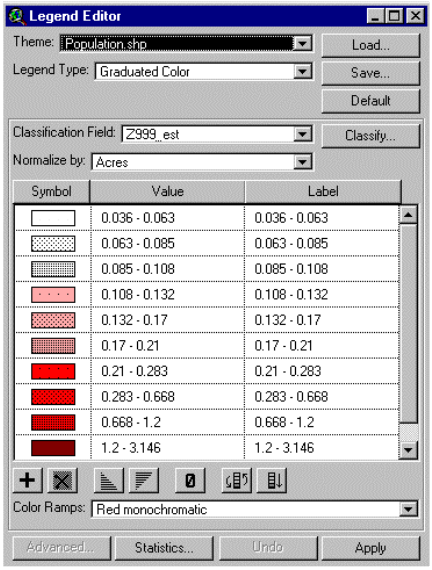


Figure 15. ArcView's "Legend Editor" for the theme "Population.shp" with 10 classes and patterns to help differentiate classes.

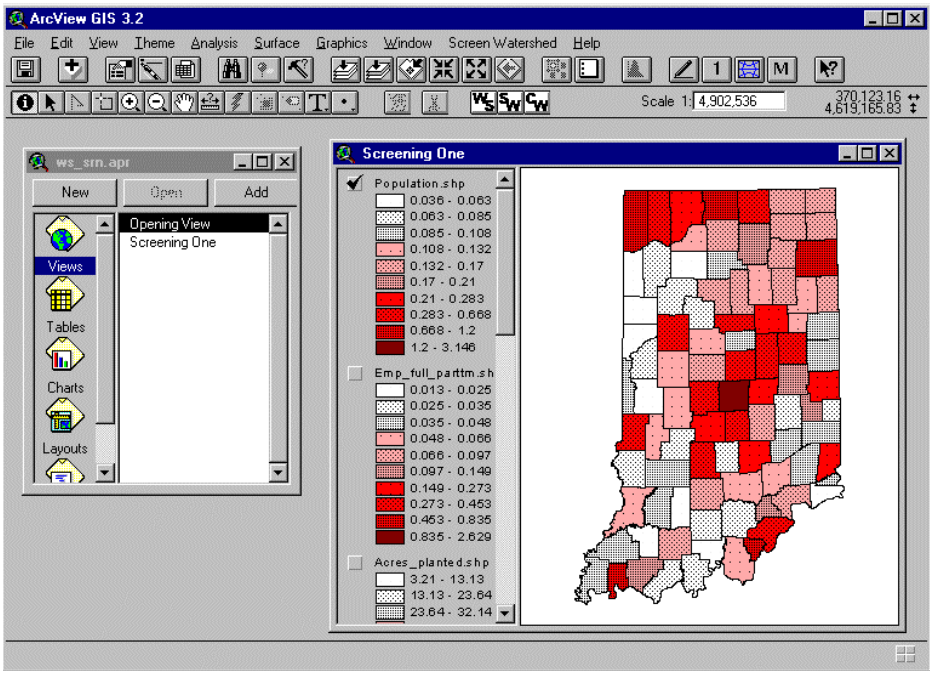


Figure 16. ArcView's application window showing a view's table of contents with "Graduated Color" legends and a "Graduated Color" data layer.

Figure 16 shows the results of applying the "Graduated Color" legend type and the resulting representation of the data layer. ArcView lists the legend entries with the smallest number first. The legend entries must be reversed if the effect of the theme is greater when the magnitude of the number is smaller (dissolved oxygen, as an example). The sort ascending/descending buttons (triangles) at the bottom of the "Legend Editor" (Figure 15) are used to reverse the order. This reversal is important because the application assigns rank values on the basis of the legend entries, with a rank value of "1" assigned to the first entry. The "Legend Tool" can be used to revise the description of the spatial variation at any time during the screening.

This technique assumes that the themes used in the analysis contain only polygons. Themes that represent data as lines or points must be modified to polygons. ArcView and "Spatial Analyst" offer methods for converting point and line data to polygons. The assignment of buffers along lines or neighborhoods around points (just two of the methods available for converting lines and points to polygons) is best left to those who know the resolution of the data layer. This application, therefore, will not address the use of line and point data.

Defining the Watershed of Interest

A watershed of primary interest must be identified. The watershed of interest has two functions. First, the watershed of interest is used as the focal point for displaying the results of the watershed screening. Second, when the watershed of interest is an 11-digit hydrologic unit or smaller, the watershed of interest is used to restrict the information processed to the 8-digit hydrologic unit containing the watershed of interest. The "Watershed of Interest" function is accessed through the "WS" button (Figure 17) on the "tool bar." The "tool bar" is the last row of icons at the top of the ArcView application window (Figure 16).

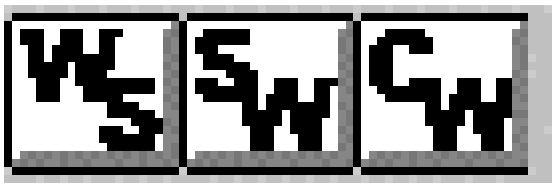


Figure 17. Watershed of Interest, Split Watershed, and Cut Watershed tool buttons.

Select Watershed of Interest

Clicking on the "WS" tool causes the "Save Watershed of Interest" window to open (Figure 18). This window is used to create a new watershed of interest theme or to select an existing watershed of interest theme. When the "WS" tool is activated, the application uses the themes contained in the active view. Themes whose attribute tables contain a data field with "WOI" in the name are listed in the "Save Watershed of Interest" window. A default name for a new watershed of interest also is shown as "File Name." This name consists of the character string "WOI" followed by a number and ".shp." The number is automatically incremented as new watershed of interest themes are created with the default name. To use the default name, click on the "OK" button. To use an existing watershed of interest theme, select the theme and click the "OK" button. To assign a different name, enter that name in place of the default name and click the "OK" button.

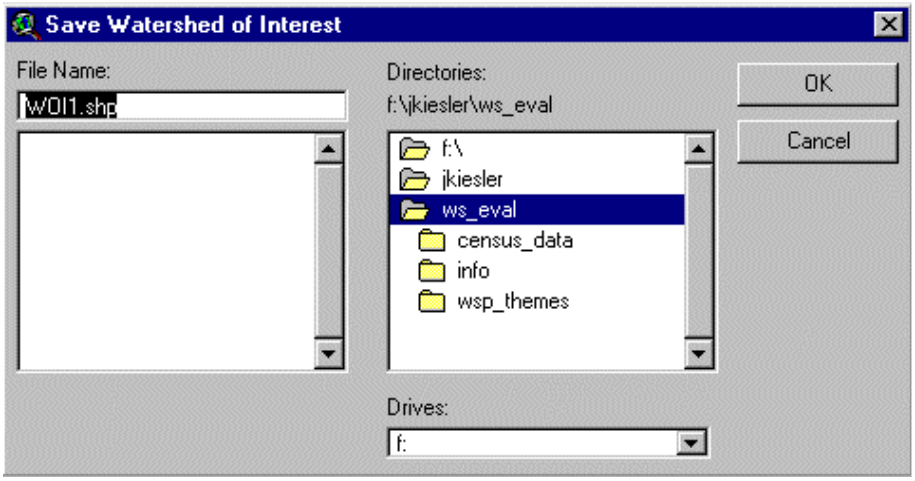


Figure 18. Watershed Screen Tools' "Save Watershed of Interest" window.

After clicking the "OK" button in the "Save Watershed of Interest" window, the 8-digit hydrologic units for the State of Indiana are displayed (Figure 19). Pointing and clicking within the bounds of the 8-digit hydrologic unit that is the primary watershed of interest or that contains a smaller watershed that is the primary watershed of interest will result in the "Is this your watershed of interest?" window being opened (Figure 20). This window shows the hydrologic unit code and name of the selected basin. The window also describes three options. "Yes," the selected watershed is the watershed of interest and the selection process stops. The user is free to execute any other function or select the "WS" tool again. "Cancel," the application waits for another 8-digit hydrologic unit to be selected. "No," the 8-digit hydrologic unit contains a smaller watershed that is the watershed of interest. When "No" is clicked, the 11-digit hydrologic units contained in the 8-digit hydrologic unit are displayed (Figure 21).

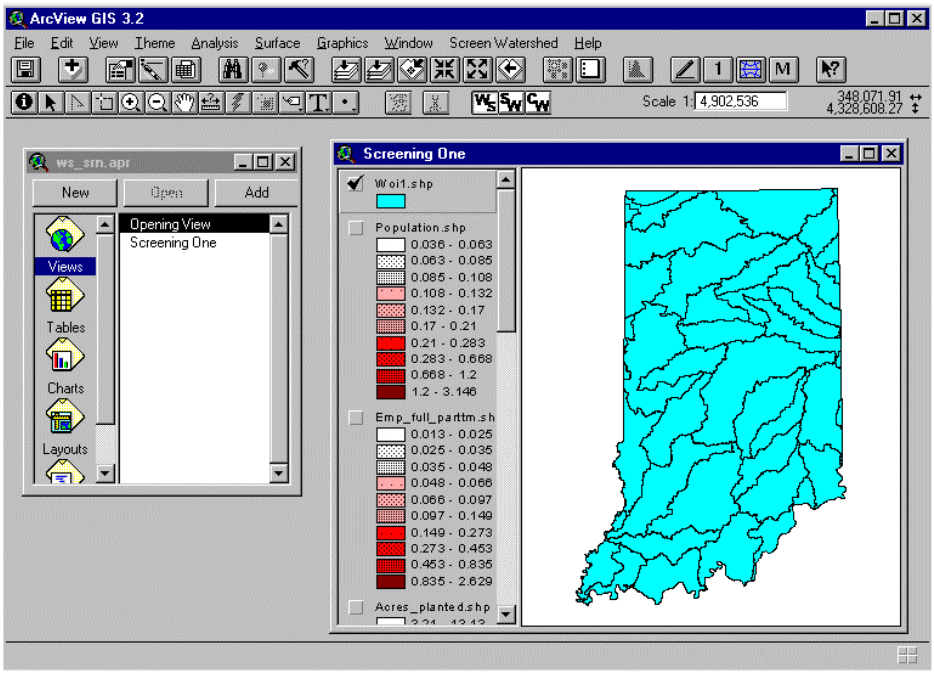


Figure 19. ArcView's application window showing the 8-digit hydrologic units from which the user can select a watershed of interest.

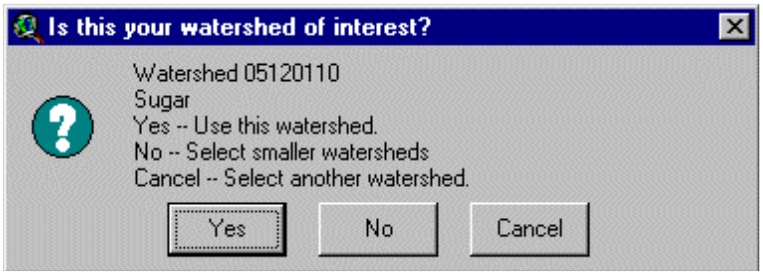


Figure 20. "Is this your watershed of interest?" window showing the selected watershed and the available options.

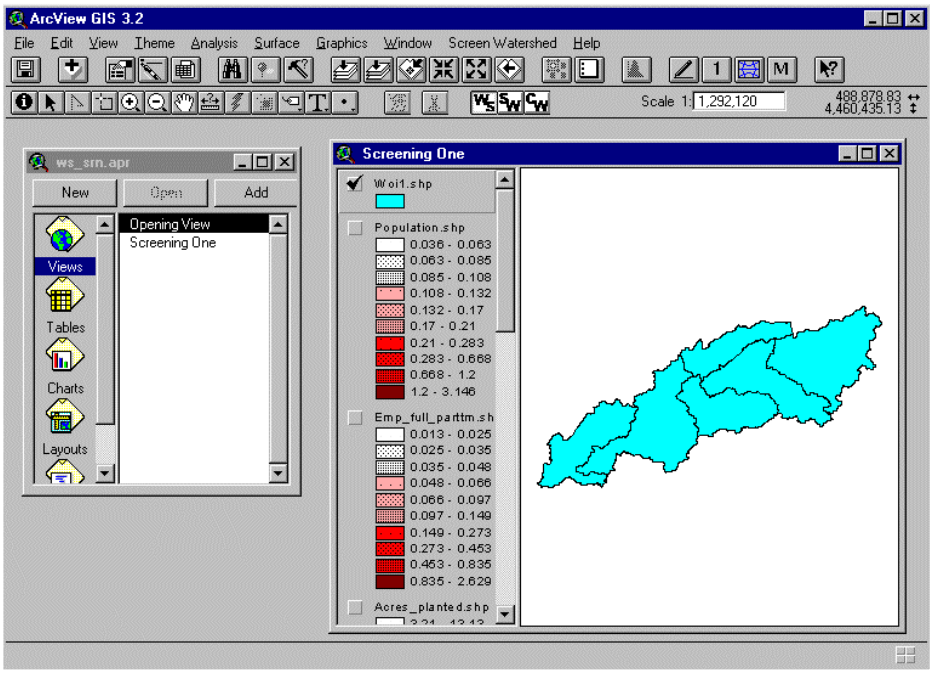


Figure 21. ArcView's application window showing the 11-digit hydrologic units within the selected 8-digit hydrologic unit.

The options available when the 11-digit hydrologic units are displayed are the same as when the 8-digit hydrologic units were displayed. Clicking on the 11-digit hydrologic unit that is or that contains the watershed of interest causes the "Is this your watershed of interest?" window to be displayed (Figure 20). Clicking the "No" button, however, will result in the 14-digit hydrologic units contained in that 11-digit hydrologic unit to be displayed (Figure 22). Pointing and clicking on the 14-digit hydrologic unit that is the watershed of interest or that contains the watershed of interest again results in the "Is this your watershed of interest?" window being displayed (Figure 20). When a 14-digit hydrologic unit is selected and "No" is clicked, the 1:100,000 stream theme is made visible (Figure 23). After the watershed of interest has been identified, any of the application's or ArcView's functions can be executed.

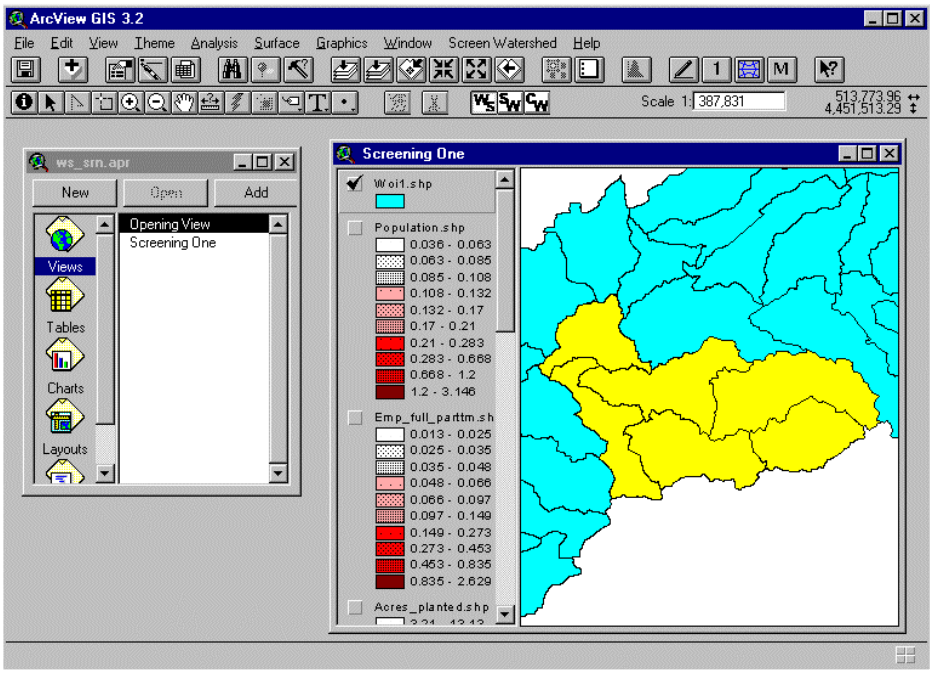


Figure 22. ArcView's application window showing the 14-digit hydrologic units within the selected 11-digit hydrologic unit.

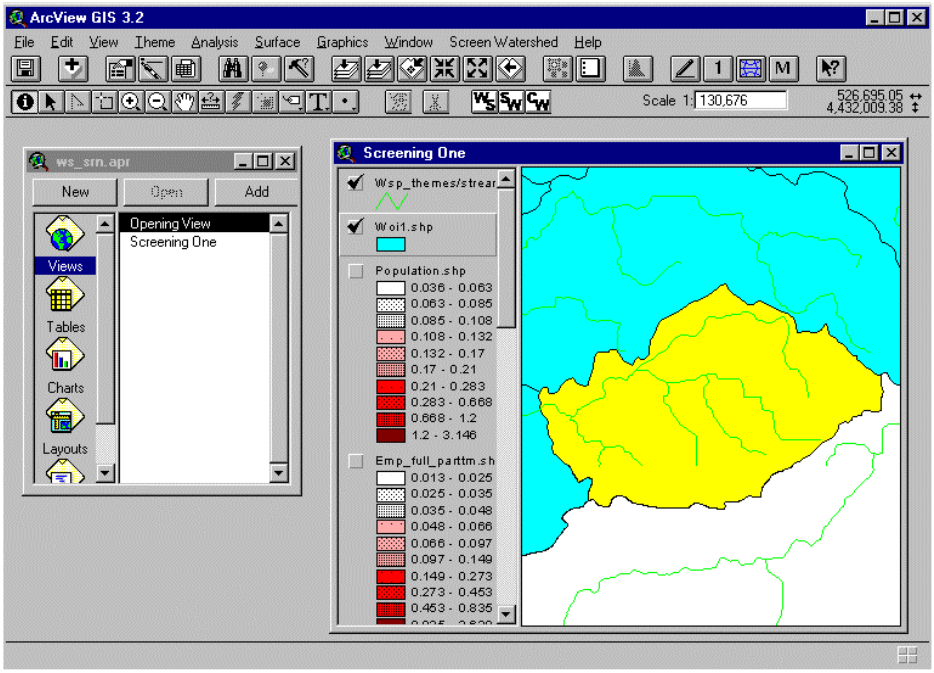


Figure 23. ArcView's application window showing a 14-digit hydrologic unit and 1:100,000 stream theme.

Split or Cut a Watershed

The split or cut watershed function, tools "SW" and "CW" from the tool bar (Figure 17), can be applied to the watershed of interest at any level (8-digit, 11-digit, 14-digit hydrologic units) or to a previously split or cut watershed. When either tool is activated, a list of watershed-of-interest themes is shown to verify the proper theme is in the view (Figure 24). When "Yes" is selected, the themes are displayed in the "Select a watershed of interest" window (Figure 25). The theme that contains the watershed to be split or cut is selected using this window.

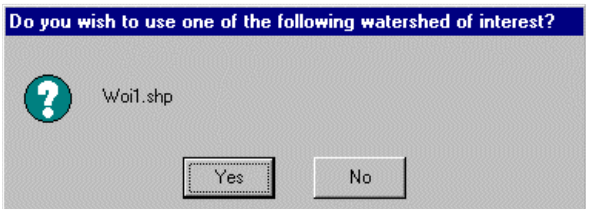


Figure 24. Window asking if one of the existing watersheds of interest should be used.

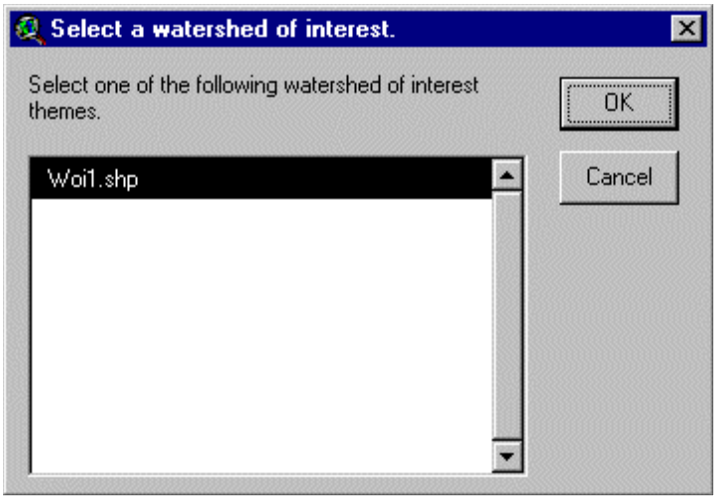


Figure 25. Watershed of Interest's "Select a watershed of interest" window.

The "SW" tool (Figure 17) is used to split a watershed into two sections. A watershed or part of a watershed is split into two areas along a line (Figure 26) that is defined by the user. The split is accomplished by specifying the dividing line using the mouse. The first and last points should be outside the watershed being split. Double clicking on the last point ends the line definition. The "WS" tool (Figure 17) must be used to identify the portion of the split watershed that will be the watershed of interest.

An interior part of a watershed can be identified as a separate watershed by cutting the desired section from the watershed using the "CW" tool (Figure 17). Pointing and clicking along a line that defines the interior part identifies the part of the watershed to be cut out of the watershed. Double clicking on the last point will close the polygon. Figure 27 shows a watershed that has been cut. The "WS" tool (Figure 17) must be used to identify the cut watershed that will be the watershed of interest.

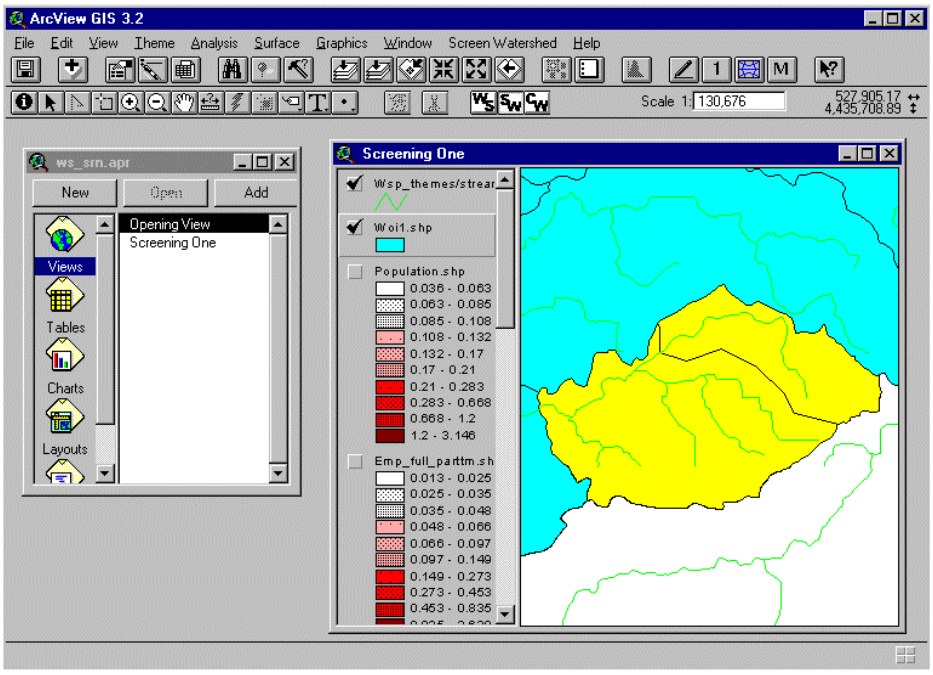


Figure 26. ArcView's application window showing a 14-digit hydrologic unit that has been split into two watersheds.

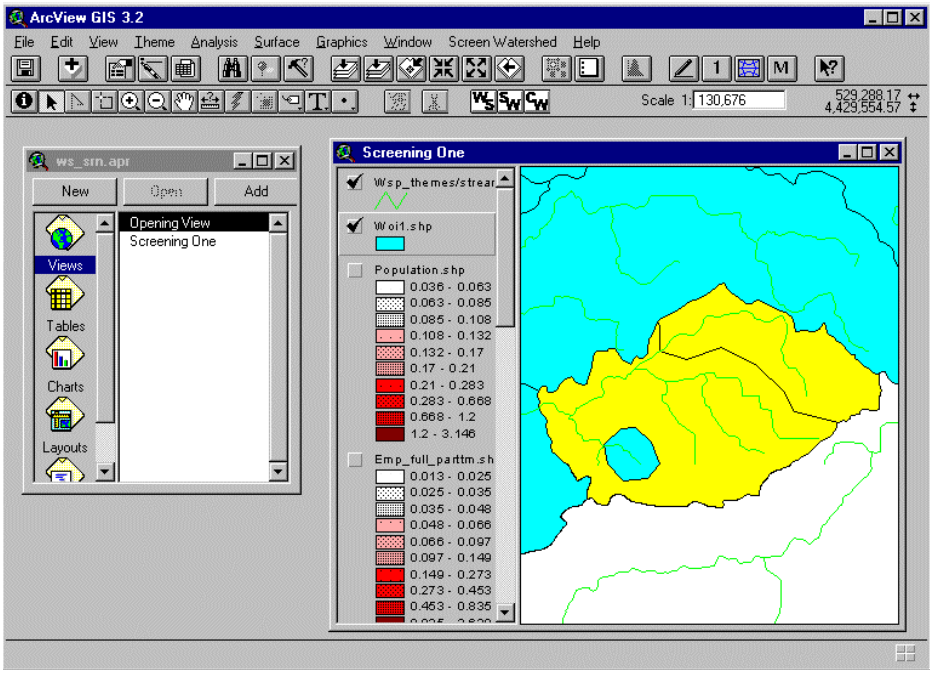


Figure 27. ArcView's application window showing a 14-digit hydrologic unit that has been split and cut into three watersheds.

Weight Factors

The relative importance of one theme in relation to others can be defined by using the "Weight Factor" function. The "Weight Factor" function is accessed using the "1" button (Figure 28) on ArcView's application window button bar. When the "Weight Factor" function is activated, a list of "active" themes is displayed in the "Select themes to include in the scenario" window (Figure 29). The "active" themes will be selected from the "active" view. A View is active when the title bar is blue (the blue bar with "Screening one" in Figure 27). A theme is active when the legend entry in the view's table of contents is raised ("Woi1.shp" in Figure 27). Pointing and clicking on a theme's legend entry in the view's table of contents makes the theme active or will deactivate any active themes in the view. A double click will activate the "Legend Editor." To activate multiple themes, point and click on the first theme and then, while holding down the shift key, point and click on the additional themes.

Each group of themes and weight factors used as input to a watershed screening is considered a "scenario." The "Weight Factor" function creates a scenario file that maintains the list of themes and weight factors and another file that maintains a list of scenario files and descriptions. The "Weight Factor" function allows a record of themes and weight factors to be maintained by the application and the reuse of a scenario.

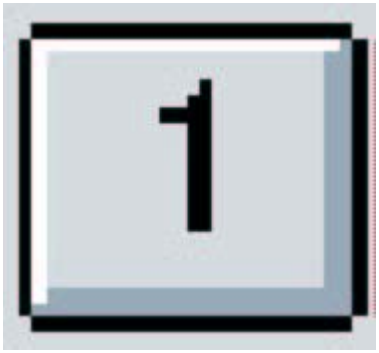


Figure 28. The "Assign Weight Factor" button.

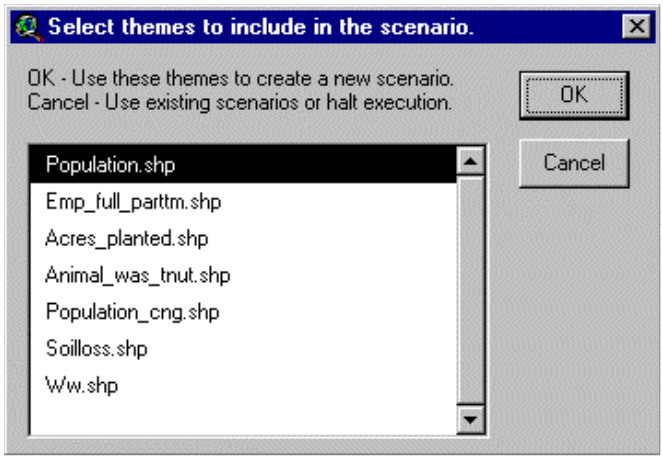


Figure 29. The "Weight Factor" function's "Select themes to include in the scenario" window listing active themes.

Creating New Scenarios

The "Weight Factor" function progresses along two paths. Either a new scenario is created using the active themes in a view or an existing scenario file is used or modified. To create a new scenario, the "OK" button is activated in the "Select themes to include in the scenario." window (Figure 29). Pointing and clicking on the "OK" button tells the function that all themes listed in the window are the themes to include in this scenario and activates the "Select the file to contain the scenario file." window (Figure 30). To stop the "Weight Factor" function or to create a new file for maintaining scenario files, point and click on the "Cancel" button in the "Select the file to contain the scenario file." window. This cancellation results in activating the "New File?" window (Figure 31). Pointing and clicking on "Yes" will cause a new file to be created and "No" will cause the "Weight Factor" function to end.

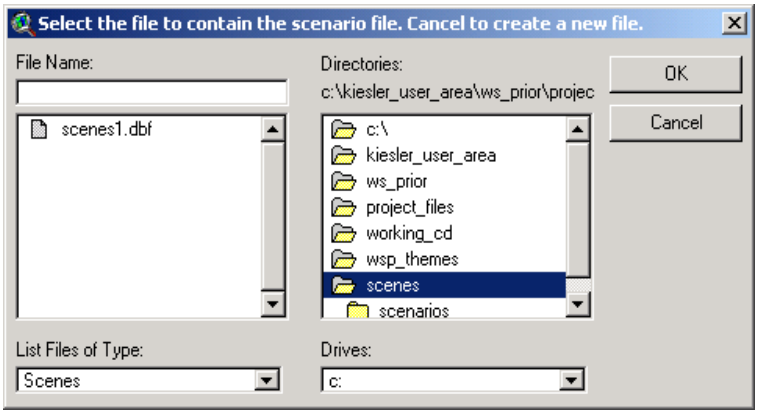


Figure 30. The "Weight Factor" function's "Select the file to contain the scenario file."

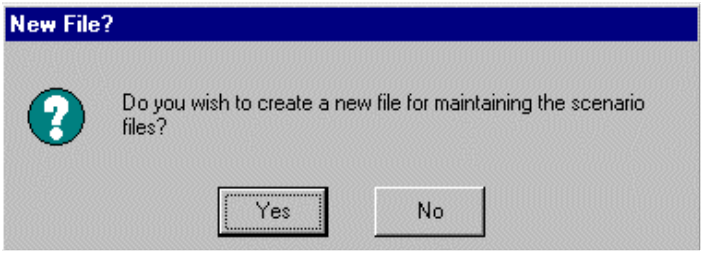


Figure 31. The "Weight Factor" function's "New File?" window.

Whether an existing file is selected or a new file for maintaining the scenario files is created, the active themes and text entry used for entering the weight factors are shown in the "Weighting Factors" window (Figure 32). Pointing and clicking on "Cancel" will stop the "Weight Factor" function. Pointing and clicking on "OK" results in windows asking for the scenario file name (Figure 33) and description (Figure 34). The themes and weight factors are saved in the scenario file. The file name should not include blanks, and the file name and description are limited to 150 characters each.

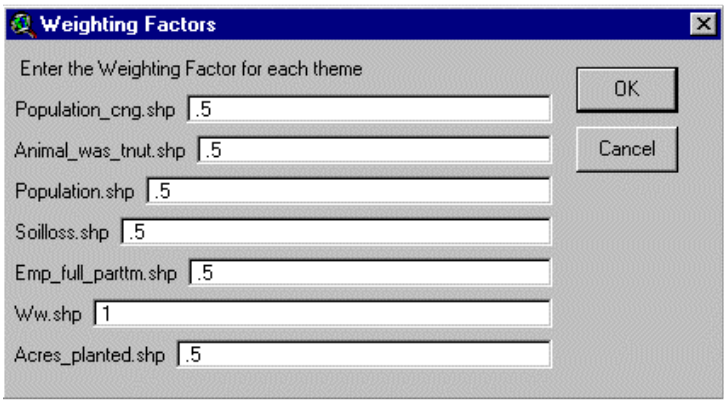


Figure 32. The "Weight Factor" function's "Weighting Factors" window.

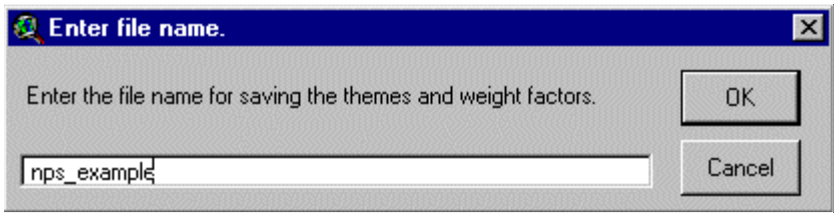


Figure 33. The "Weight Factor" function's "Enter file name" window for naming the scenario file.

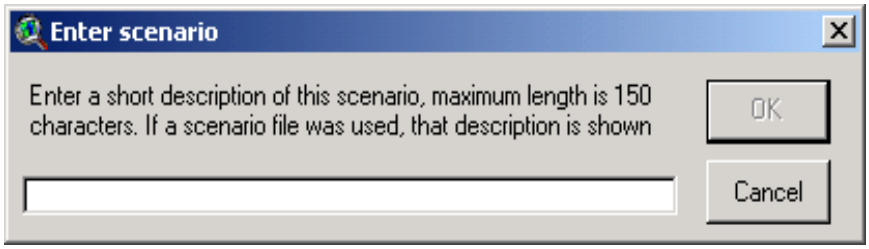


Figure 34. The "Weight Factor" function's "Enter scenario" window for entering the description of the scenario file.

Using Existing Scenarios

To use or modify an existing scenario, point and click on the "Cancel" button in the "Select themes to include in the scenario" window (Figure 29). "Cancel" will open the "Select the file to contain the scenario file and description" window (Figure 30). The behavior of this window is similar to that when creating a new scenario file except that after selecting a file and pointing and clicking on the "OK" button, the "Weight Factor" function opens the "Select Scenario" window (Figure 35), and the scenario files contained in this file are displayed. Pointing and clicking on "Cancel" (Figure 35) will end the "Weight Factor" function. After a scenario file is selected, the themes and their weight factors are displayed in the "Weighting Factors" window (Figure 32). The behavior of this window is similar to its use in "Creating new scenarios." The weight factors can be saved in the same file or saved to a new file using the "Enter file name" window (Figure 36). Pointing and clicking on "Cancel" causes the "Weight Factor" function to save the themes and weight factors in the same file. A revised file name and the "OK" button result in saving the themes and weight factors in a new file.

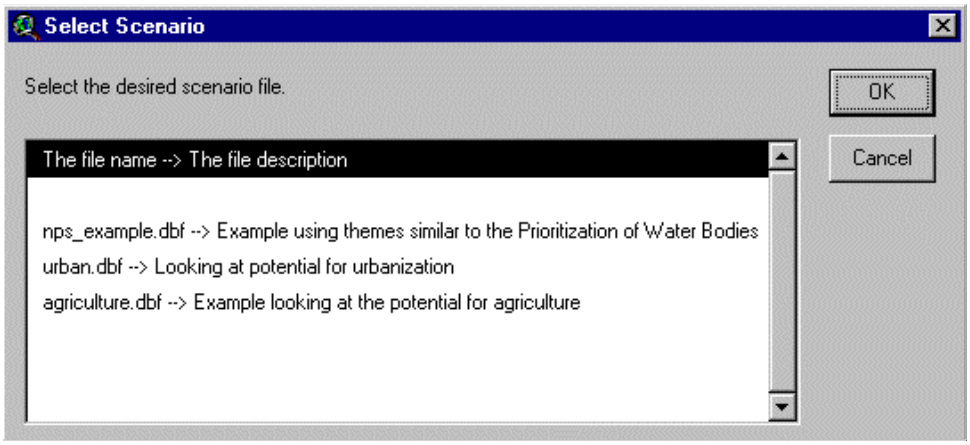


Figure 35. The "Weight Factor" function's "Select Scenario" window.

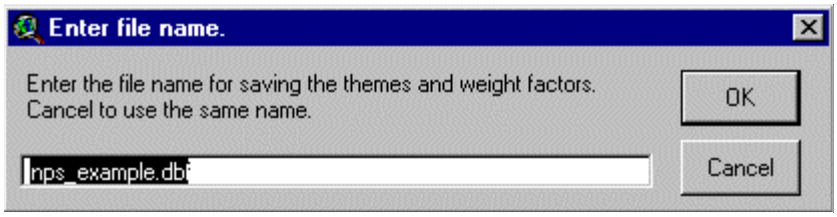


Figure 36. The "Weight Factor" function's "Enter file name" window used when weight factors have been modified.

Screen Watershed

The combining of the themes occurs in the "Screen Watershed" function accessed through the "Screen Watershed" pull-down menu or the "blue spider web" button (Figure 37) on the button bar. When the function is activated, the "Select the file to contain the scenario file and description" window is opened (Figure 30). After a file has been selected, the "Select Scenario" window (Figure 35) is opened and the desired scenario is selected. The themes and weight factors for that scenario are displayed in the "Weighting Factors" window (Figure 32). The weight factors can be modified at this time, but no record of these changes is maintained. To maintain a record of the themes and weight factors, the "Weight Factor" function should be used to change the weight factors. "Screen Watershed" verifies that the themes contained in the scenario file exist in the View. The themes do not have to be active.

Figure 37. The "Evaluate Watersheds" button.

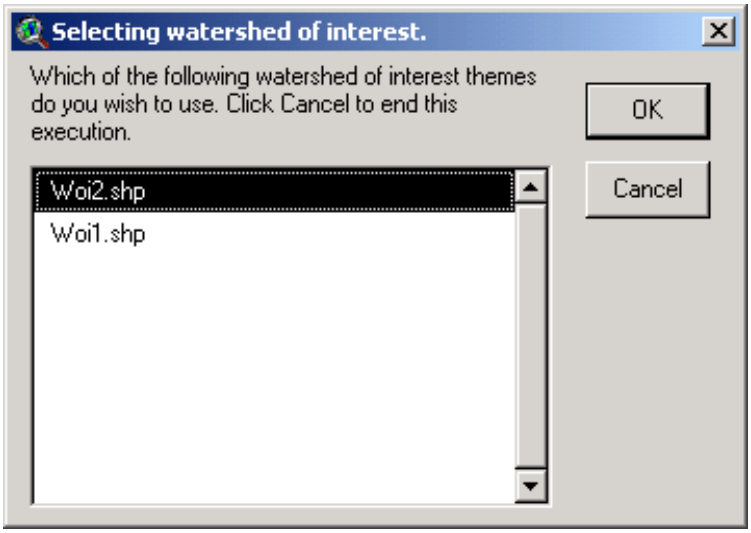


Figure 38. The "Evaluate Watershed" function's "Selecting watershed of interest" window used to select the watershed of interest being screened.

After the scenario file has been selected and the weight factors accepted, the watershed being screened is selected using the "Selecting watershed of interest." window (Figure 38). The contents of Figure 38 are any themes whose attribute table contains a data field whose name contains "WOI." The watershed of interest can be selected from the themes shown in Figure 38.

Assign Rank Values

Next, "Screen Watershed" assigns a rank value to the features (counties, watersheds) that make up each theme contained in the scenario file. Ranks are assigned on the basis of the number of classes in a theme's legend. The maximum rank value is the maximum number of classes describing the spatial variation of any theme in the scenario. In Figure 16, the maximum rank value is "10." Features with values in the class at the top of the legend entry, Class Index 0, are assigned a rank value of "1"; features in the next class, Class Index 1, a rank value of "2"; and so on. The increment between rank values will change for themes with fewer classes. Rank values are assigned using the formula

1 + (Class Index * ((Maximum number of classes – 1) / number of classes in theme)).

After rank values have been assigned, each theme is converted to a grid. The size of the grid cell is defined using the "Enter cell size" window (Figure 39). The grid-cell-size units are in the map units defined in the "View Properties" window (Figure 5), which are in meters. The default is 500 map units. The cell size

selected should be small enough to maintain the spatial definition of smaller features within the themes used in this scenario. The product of the rank value and the weight factor defines the value of each grid cell.

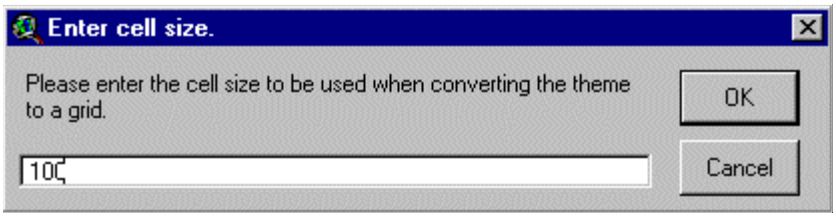


Figure 39. The "Evaluate Watershed" function's "Enter cell size" window.

When the watershed of interest is smaller than an 8-digit hydrologic unit, the "Screen Watershed" does not convert the complete theme to a grid. In this case, "Screen Watershed" converts only those features that are part of the 8-digit hydrologic unit that contains the watershed of interest. Figure 40 shows such a case. The counties that the 8-digit hydrologic unit overlay contain data values. The remainder of the grid contains a "no data" indicator although data exist for those areas.

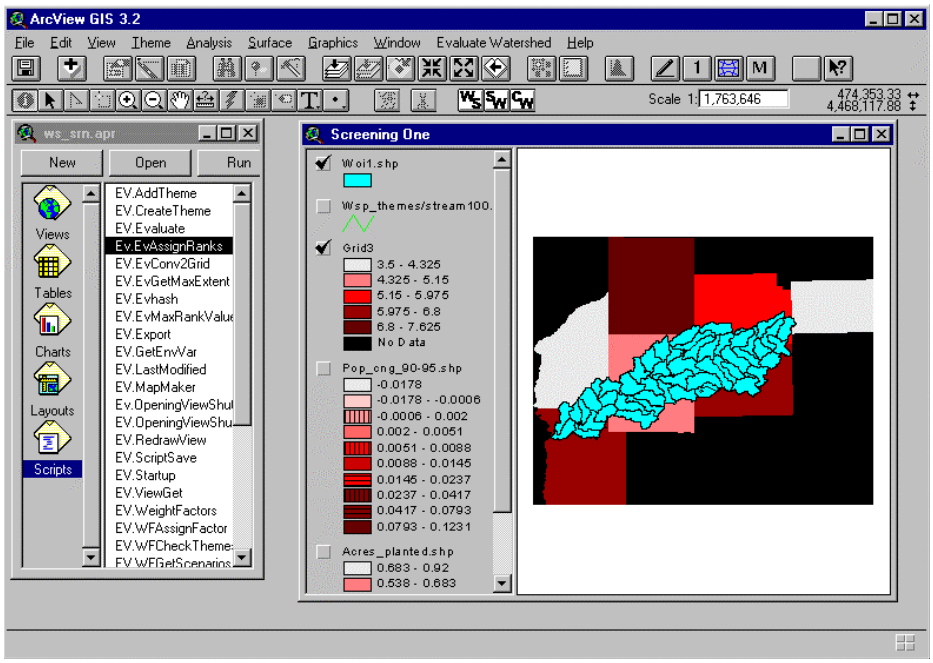


Figure 40. ArcView's application window showing the limits of gridded data.

Combining Themes

After each theme has been converted to a grid, a method for combining the grids is selected using the "Select statistic used to combine grids." window (Figure 41). Using any entry shown in Figure 41 is justified, but the "Sum" entry is suggested for use in the "Screen Watershed" function.

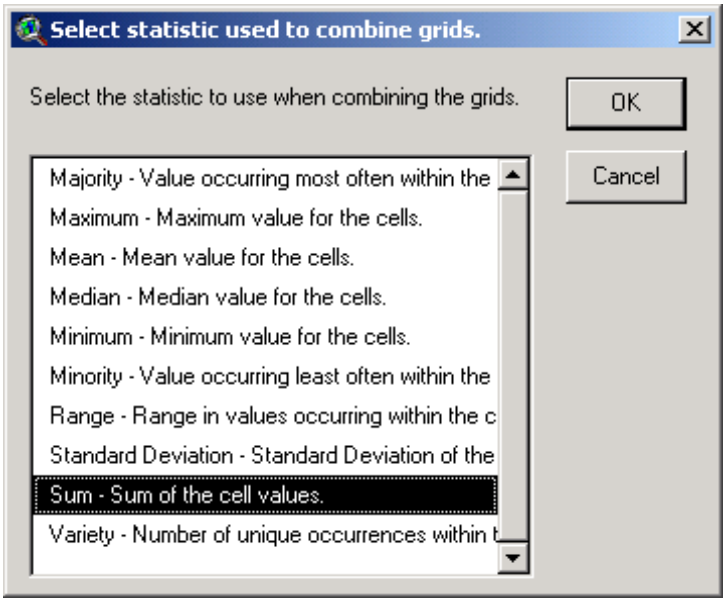


Figure 41. The "Evaluate Watershed" function's "Select statistic used to combine grids." window allows for the selection of the entry to be used when combining grids.

The combined grid is summarized to populate the watershed of interest theme. The entry used to summarize the combined grid is selected using the "Determining watershed screening factor." window (Figure 42). The watershed of interest selected in the "Selecting watershed of interest" window (Figure 38) is used to define the aerial extent of each summarization. As with the "Select statistic used to combine grids." window, using any entry in Figure 42 is justified. The "Mean" entry, however, is best used when the cell size is small when compared to the size of the features being summarized. The "Sum" entry is best used when a few cells have a disproportionate affect on the mean. When the "Sum" entry is used, the "Legend Editor" should be used to normalize the sum using the area of the features in the combined theme. When the cell size is small, the number of cells being used is greater, making the mean more representative of the cells across the feature. When the cell size is large, however, the mean could be affected by parts of the input themes that are outside the output feature. In this case, the "Sum" entry, which has been normalized by the size of the output feature, should be considered. Figure 43 shows the results of the "Screen Watershed" function.

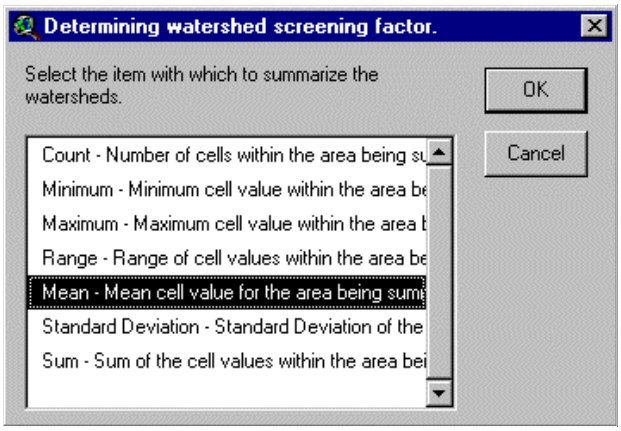


Figure 42. The "Evaluate Watershed" function's "Determining watershed screening factor." window for selecting the entry used to summarize the combined grid.

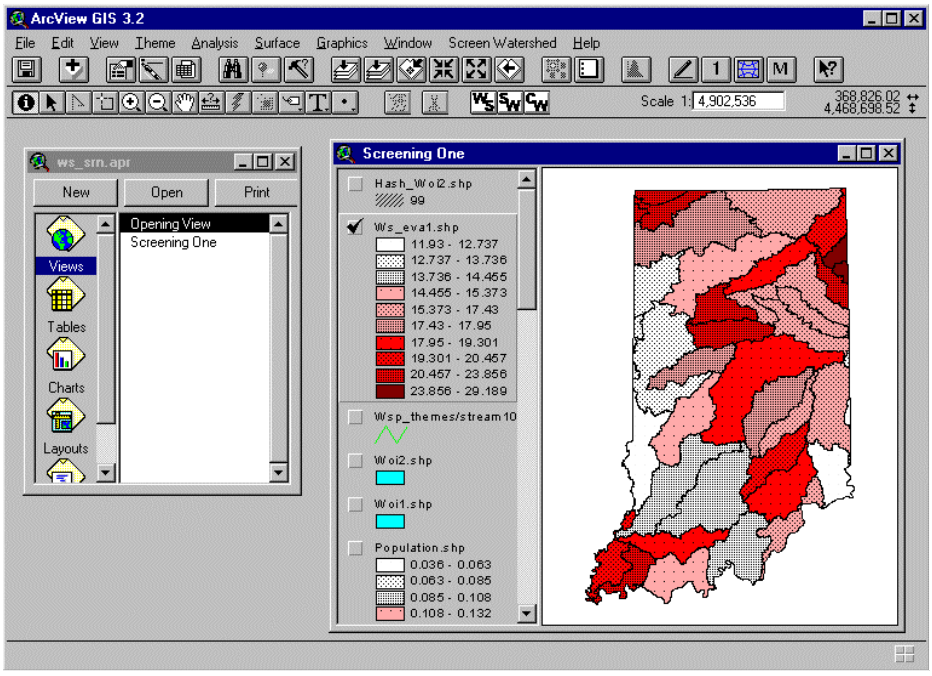


Figure 43. ArcView's application window showing an entire screened watershed.

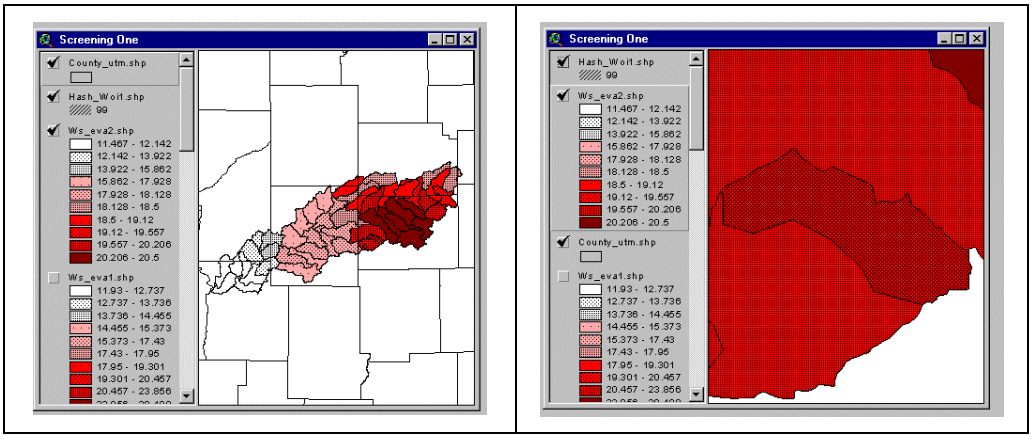


Figure 44. ArcView's application window showing a screened watershed, with the watershed of interest as the focal point.

Create Themes

New themes based on county or 8-, 11-, or 14-digit hydrologic units can be created using the "Create Themes" function. The "Create Themes" function is accessed using the pencil button (Figure 1) in the ArcView application window or through the "Screen Watershed" pull-down menu (Figure 43). The "Create Themes" window (Figure 46) opens when the "Create Themes" function is started. This window is used to identify which base theme is to be used when creating the new theme. If the county base is selected, the type of identifier used to uniquely identify each county is selected using the "Creating a theme" window (Figure 47). The definitions of the data fields used to create a new theme are shown in Table 1.

Table 1. Field definitions used when creating new themes

[FIPS; Federal Information Processing Standard]

Base theme	Identifier field description	Identifier field name	Identifier field definition
County	FIPS State and County Code	Stcnty	Character 5
County	FIPS County Code	FIPS_Cnty	Number 3.0
County	State Specific County Code	IN_Cnty	Number 2.0
8-digit hydrologic units	8-digit hydrologic-unit code	Huc_8	Character 8
11-digit hydrologic units	11-digit hydrologic-unit code	Huc_11	Character 11
14-digit hydrologic units	14-digit hydrologic-unit code	Huc_14	Character 14

Figure 45. The “Evaluate Watershed” function’s “Create Theme” button.

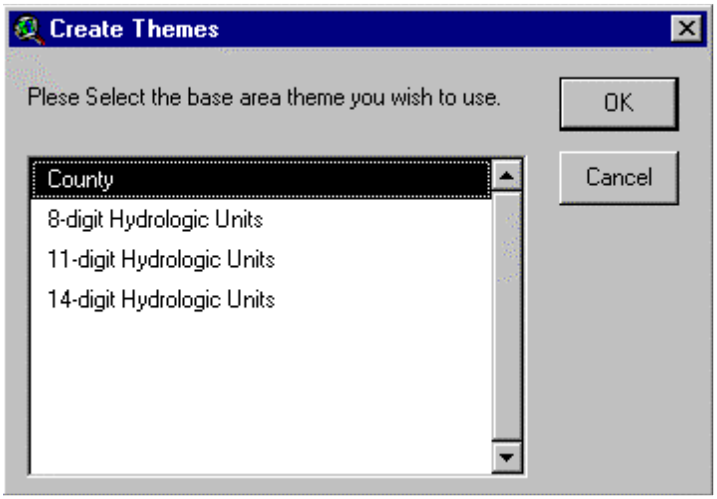


Figure 46. The “Create Themes” window for selecting the base theme to which a data file will be joined.

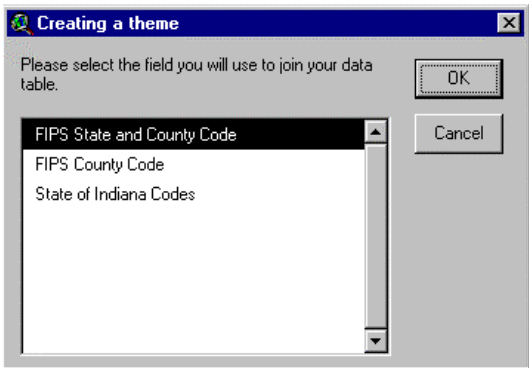


Figure 47. "Creating a theme" window for selecting the type of county identifier to be used when joining a data table to the county coverage.

The data table used to create the new theme must contain a minimum of two columns: an identifier column and a data column. The identifier column must contain the appropriate identifiers in a format suitable for the selected base theme. The column name does not have to be the same as the field name given in Table 1, but the definition must be the same. The table format can be dBase, INFO, or text delimited. The data file is selected using the "Add Table" window (Figure 48). The data column that contains the identifier data is selected using the window shown in Figure 49.

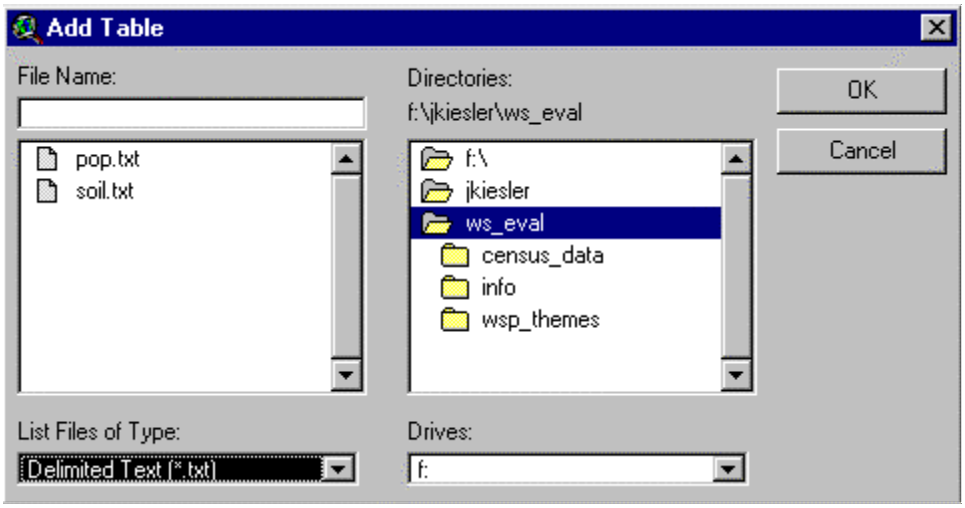


Figure 48. The "Create Theme" function's "Add Table" window used to select the data file that will be joined to a base theme.

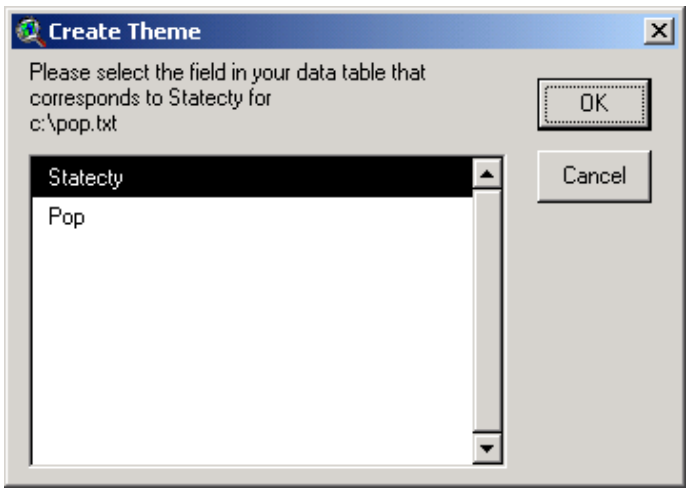


Figure 49. The "Create Theme" function's window for identifying the field in the data table that contains the identifier to be used to join the data table to the base theme.

The "Create Theme" function creates a default name for the new theme. The default name follows the structure "wse_jo#.shp" where # is a number, beginning with "1", assigned by the application to uniquely identify the theme. A different name can be entered using the window shown in Figure 50. This window opens once the joining of the data file and the base theme's attribute table is complete. An example of a newly created theme is shown in Figure 51.

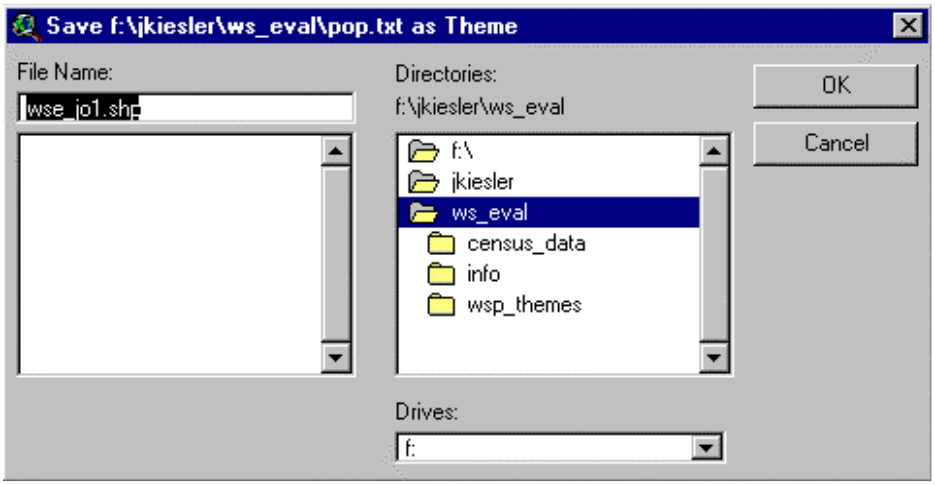


Figure 50. The "Create Theme" function's window for naming the newly created theme.

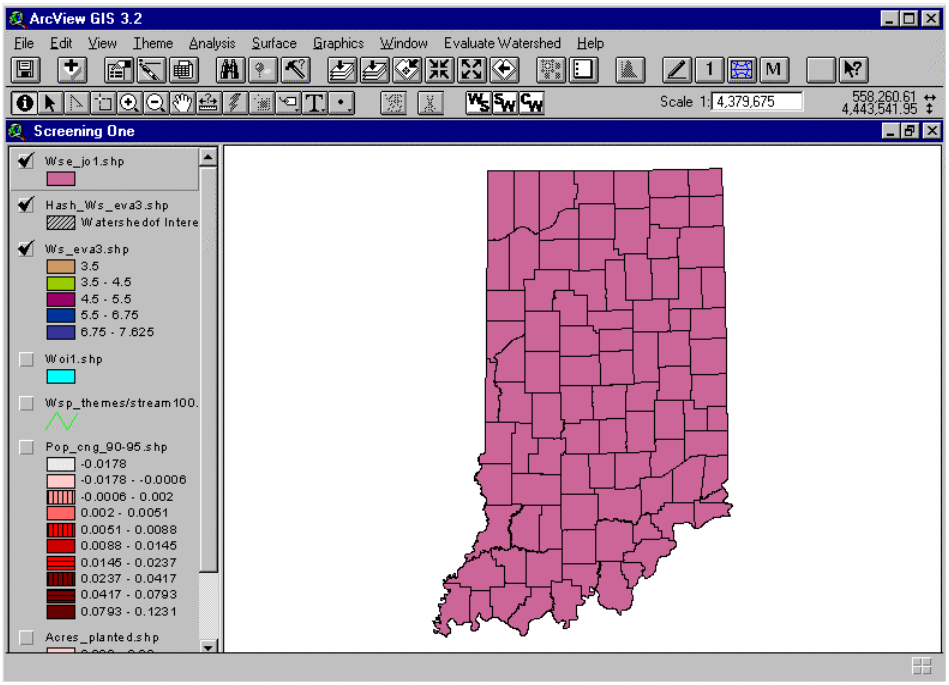


Figure 51. ArcView's application window showing a newly created theme.

Print Maps

This function uses ArcView's "Layout" option to create a hard copy of themes visible in the active view. The information to be shown on the map is made visible in the active view, and the "M" button (Figure 52) on ArcView's application button bar is activated. What is visible in the view will be shown on the map. The "Map" function opens a window that asks for a two-line title that will be printed on the map (Figure 53). The "Map" function generates a map that is focused on the watershed of interest but also contains a smaller reference map (Figure 54). When the map contains all the desired information, the print option from the "File" pull-down menu (Figure 55) is activated to print the hard copy.

Figure 52. The "Print Map" button.

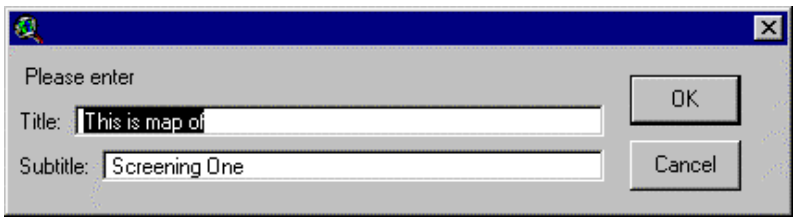


Figure 53. Screen allowing the user to enter a title for the map.

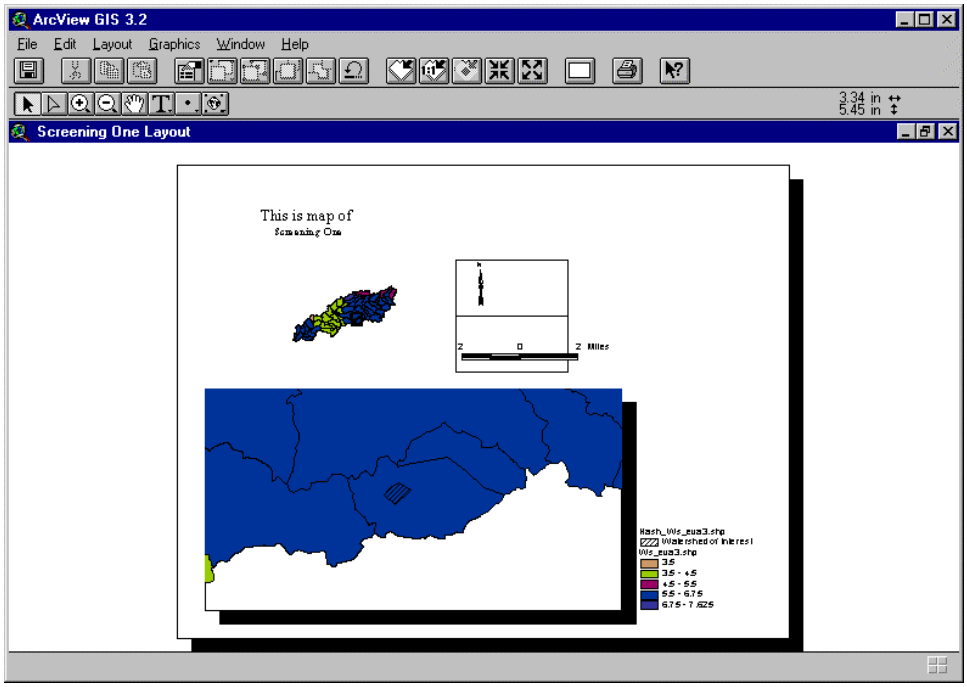


Figure 54. Window showing a map ready for printing.

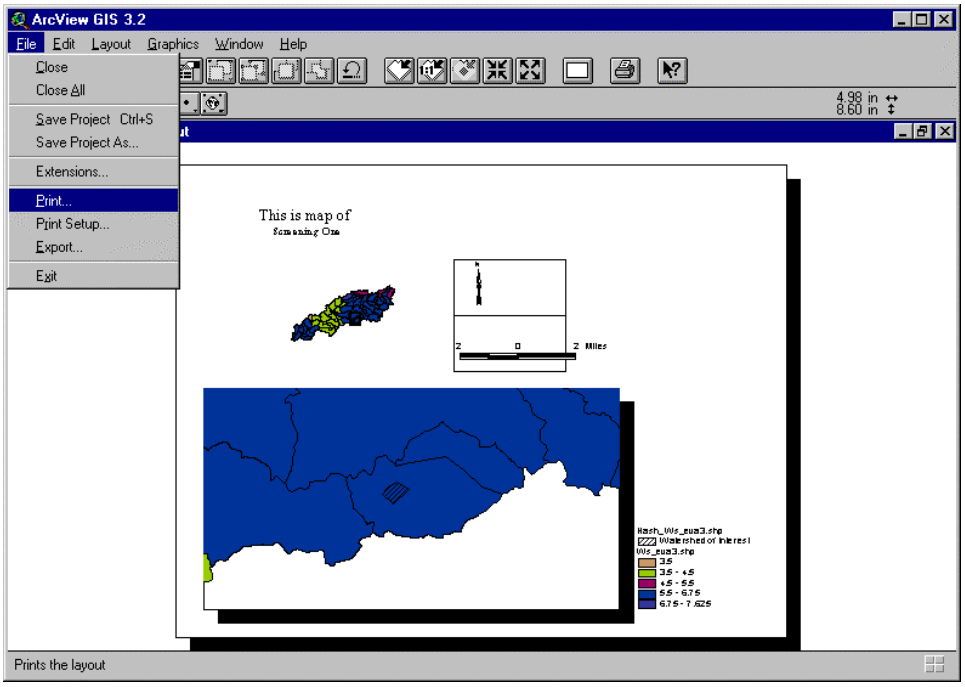


Figure 55. The "Print" option in ArcView's application "File" pull-down menu.

SENSITIVITY ANALYSIS

This section of the report examines the ability of the application to accurately duplicate the spatial variability of a theme, the effect spatial resolution of a theme could have on the combined theme, and several of the application's input options and how the selections made could affect the combined theme. The input options examined are the number and magnitude of classes used to define the spatial variability of the input themes, the grid-cell size used to convert themes to grids, and the statistic used to convert a grid to a watershed.

Because no data sets were readily available for 14-, 11-, and 8-digit hydrologic units and counties, a test data set was developed. The test data set was developed by assigning a random number to each 14-digit hydrologic unit in Indiana. The value assigned to an 11-digit hydrologic unit was the median of the random numbers for each 14-digit hydrologic unit within the 11-digit hydrologic unit. The median of the random numbers assigned to the 14-digit hydrologic units within an 8-digit hydrologic unit was assigned to that 8-digit hydrologic unit. The median of the random numbers assigned to any 14-digit hydrologic unit that lies partially or completely within a county was assigned to the county. Each data set was used to create the test themes used to evaluate the application.

Application Accuracy

Figure 56 demonstrates the application can duplicate the spatial variability of a test theme. Figure 56 shows the 8-digit-hydrologic-unit test theme divided into 15 classes overlaying the application's output with 15 classes. The output was generated using the 8-digit-hydrologic-unit test theme divided into 15 classes, a weight factor of "1" was assigned to the test theme, the sum was selected to combine grids (this is irrelevant because only one theme was included in the analysis), and the mean of the grid cell values was used to summarize the grid and populate the 8-digit-hydrologic-unit output theme. The class definitions were entered by hand to represent the full range of rank values. The overlay indicates the spatial distribution of the theme produced by the application is identical to the original data set.

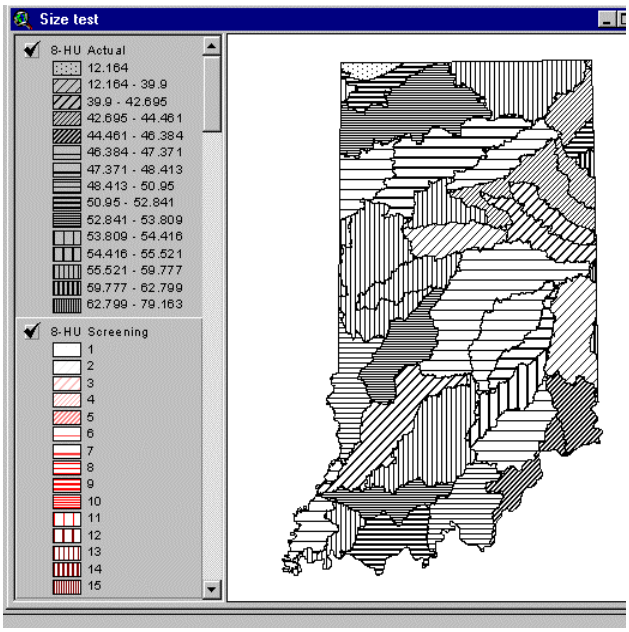


Figure 56. The accuracy of the application to duplicate the spatial variation of a test theme.

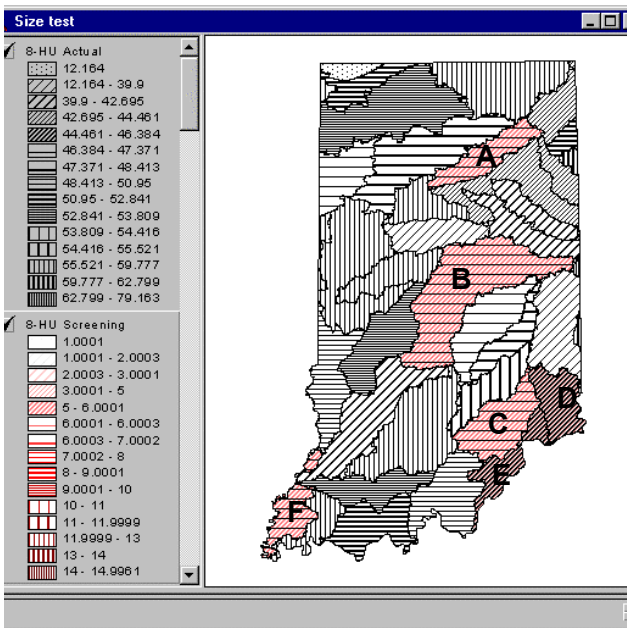


Figure 57. The effect of the class definitions on the combined theme.

Figure 57 shows the effect that class definitions can have on the combined theme. Figure 57 shows the same output as Figure 56, except ArcView generated the class definitions to four decimal places using the "natural break" function. In Figure 57, six of the 8-digit hydrologic units did not match the class assignments of the original test theme. Examination of Figure 57 and Table 2 shows the mean grid values for these six 8-digit hydrologic units that differ are the same as the rank values assigned to the hydrologic units in the test theme. This indicates the application is creating an accurate representation of the original data set's spatial variability. Examination of the class values indicates that the automated division using natural breaks with 15 classes to four decimal places and the method ArcView uses to assign features to classes caused the classes of these six 8-digit hydrologic units to be shifted.

Table 2. The rank and class number for the 8-digit hydrologic unit and the application-generated theme that differ

Hydrologic unit	Original data set			Generated theme	
	Value	Rank	Class	Mean	Class
A	47.23	6	6	6	5
B	47.371	6	6	6	5
C	47.150	6	6	6	5
D	45.570	5	5	5	4
E	46.384	5	5	5	4
F	47.235	6	6	6	5

Spatial Resolution

One key to understanding the combined data layer is how the size of the features that make up the input theme compares to the size of the output features (that is, the features in the watershed of interest theme). When the input features are larger than the output features, the smaller output features will have the same values within the input feature; the only variations will occur where the output features intersect the boundaries of the input features. When the output features are larger than the input features, is the value of an output feature affected by one input feature more than the others?

In a study looking at the economic value of urban forest in Louisiana, Mitchell (1996) compares parish-wide surveys with site-specific surveys. He found that the site-specific surveys resulted in a 510-percent increase in the value assigned to the urban forest within his area of study. Generally speaking, the finer the resolution of the input data, the smaller the output watershed can be and the more likely the results will be accurate.

Figure 58 shows how the spatial resolution of an input-data set could affect the results of this application. The 11-digit hydrologic units shown in the figure were generated using the county test theme. The rank values assigned to each county are shown in the figure. The maximum statistic was used to convert the grid-cell values to a watershed. Examination of Figure 58 shows how one county can affect a watershed that spans several counties. The most southern dark-colored watershed lies within three counties with rank values of 13, 13, and 14. Most of the watershed spans two counties whose ranks are 13, but the county with the 14 controls the results.

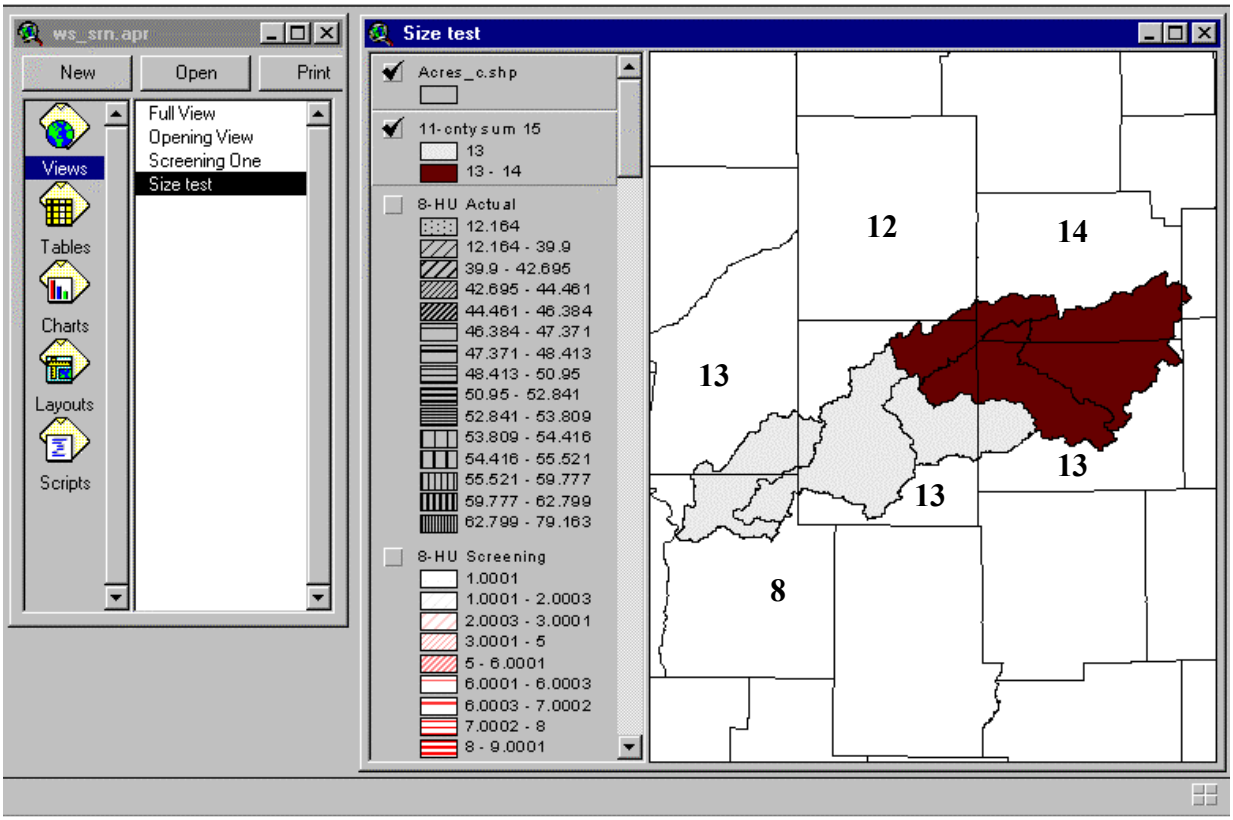


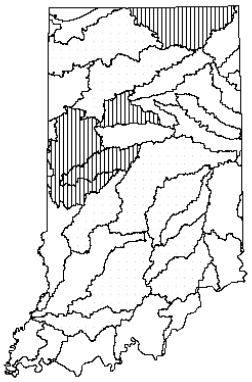
Figure 58. The potential effect of large features being used to screen a small watershed.

Figure 59 also demonstrates the potential effect of using data of lesser spatial resolution. The left-most figures show the 8-digit-hydrologic-unit theme generated using the 14-digit-hydrologic-unit test theme. The upper theme shows those watersheds in the upper 20 percent of the classes, and the lower theme shows those watersheds in the upper 33 percent of the classes. The results using the 14-digit-hydrologic-unit test theme are

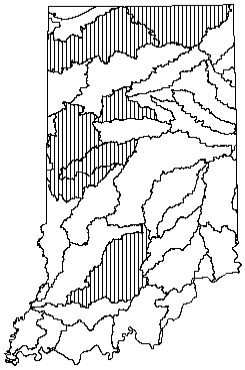
assumed to be correct.

The middle themes (Figure 59) show the 8-digit hydrologic units generated using the 11-digit-hydrologic-unit test theme. Analysis of these themes shows that two additional watersheds are placed in the upper 20 percent of the classes. The lower middle theme shows that the results using the 11-digit hydrologic units has two fewer hydrologic units in the upper 33 percent but, in general, the identified hydrologic units are similar to the 14-digit hydrologic units.

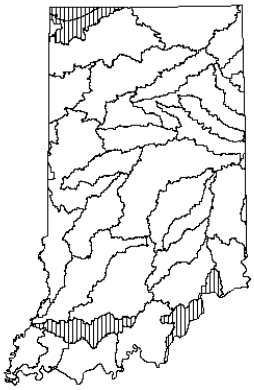
The themes generated by the application using the county data (right-most themes in Figure 59) show that the watersheds identified in the upper 20 percent of the classes differ significantly from those watersheds identified when the 14-digit-hydrologic-unit test theme was used. The watersheds identified in the upper 33 percent of the classes, however, are similar to those watersheds identified when the 14-digit hydrologic units are used but are not as similar as when the 11-digit hydrologic units were used.



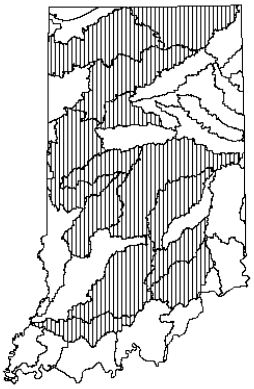
8-digit-hydrologic-unit rankings, using 14-digit hydrologic unit



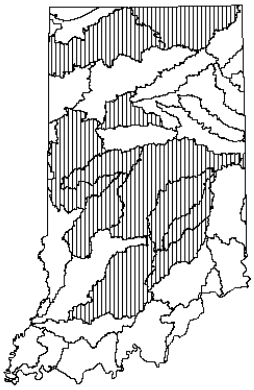
8-digit-hydrologic-unit rankings, using 11-digit hydrologic units whose values were the median of the 14-digit hydrologic units within each 11-digit hydrologic unit



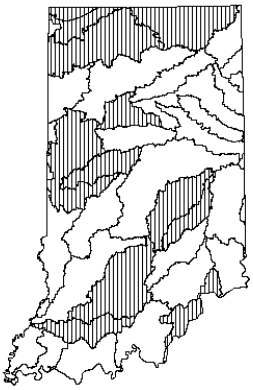
8-digit-hydrologic-unit rankings, using counties whose values were the median of the 14-digit hydrologic units completely or partially in the county



8-digit-hydrologic-unit rankings, using 14-digit hydrologic unit



8-digit-hydrologic-unit rankings, using 11-digit hydrologic units whose values were the median of the 14-digit hydrologic units within each 11-digit hydrologic unit



8-digit-hydrologic-unit rankings, using counties whose values were the median of the 14-digit hydrologic units completely or partially in the county

Figure 59. The results of using 14- and 11-digit hydrologic data and county data to screen 8-digit hydrologic data.

Number of Classes—Spatial Variability

The number of classes into which the theme is divided describes the spatial variability of a theme. Too many classes cause the application to take a longer completion time. Too few classes cause the spatial variability to be defined inadequately. Because rank values are assigned to a feature on the basis of the class to which that feature is assigned, inaccuracy in the spatial variability will cause inaccuracy to the combined theme. Figure 60 shows the results of screening 11-digit hydrologic units using the 14-digit-hydrologic-unit and county test themes.

The 14-digit-hydrologic-unit test theme was divided into 64, 32, 16, 10, and 5 classes and used to generate the 11-digit hydrologic units seen in the left-hand column of Figure 60. Examination of the left-hand column shows the results differ when the input data were divided into 5 and 10 classes but are the same for 16, 32, and 64 classes. The 14-digit-hydrologic-unit test theme has 2,426 features. To obtain reliable results, between 10 and 16 classes are needed to describe the spatial variability of the 14-digit-hydrologic-unit test.

The county test theme was divided into 5, 10, 15, 20, and 25 classes and used to generate the 11-digit hydrologic units seen in the right-hand column of Figure 60. Examination of the right-hand column shows the results are the same for 10, 15, 32, and 64 classes. The county test theme has 92 features. To obtain reliable results, between 5 and 10 classes are needed to describe the spatial variability of the county test.

Defining the spatial variation of an input theme does not have a rule of thumb. Some understanding of the spatial variability of the input themes must be known and care must be taken to verify that the spatial variability shown in the view is representative of the data.

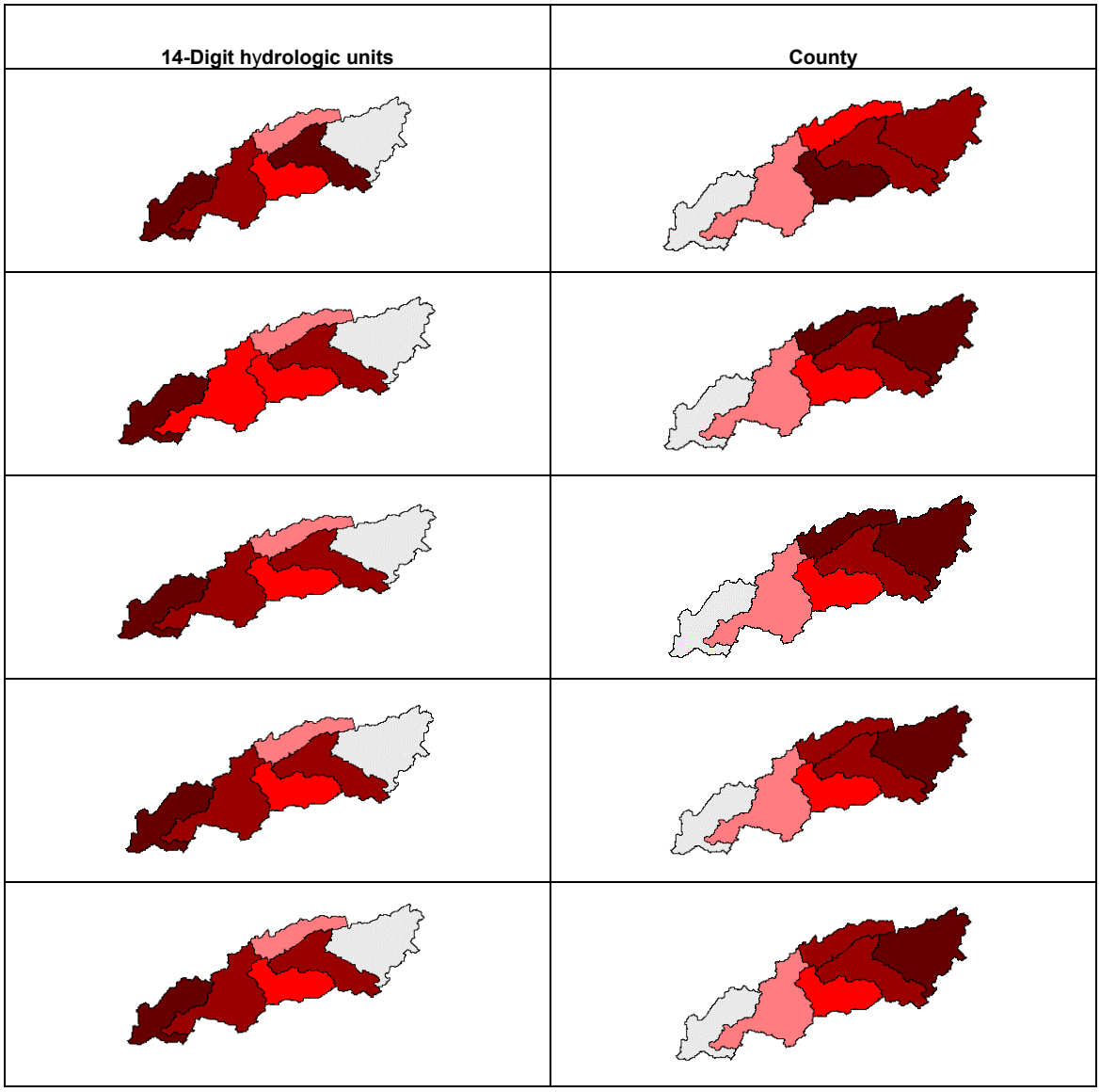


Figure 60. Results of generating 11-digit hydrologic units by using 14-digit-hydrologic-unit and county test themes that use varying numbers of classes.

Classification Type

The type of classification selected to define the spatial variation of a theme will also affect which classification class an individual feature in a theme will be placed. Figure 61 shows how the members of a class change when the classification type is changed. The "Standard Deviation" entry shows features with a negative standard deviation in blue. The source data must be examined to determine which classification type best describes the spatial variation of the data set. The "Natural Breaks" classification type was used throughout this report and example.

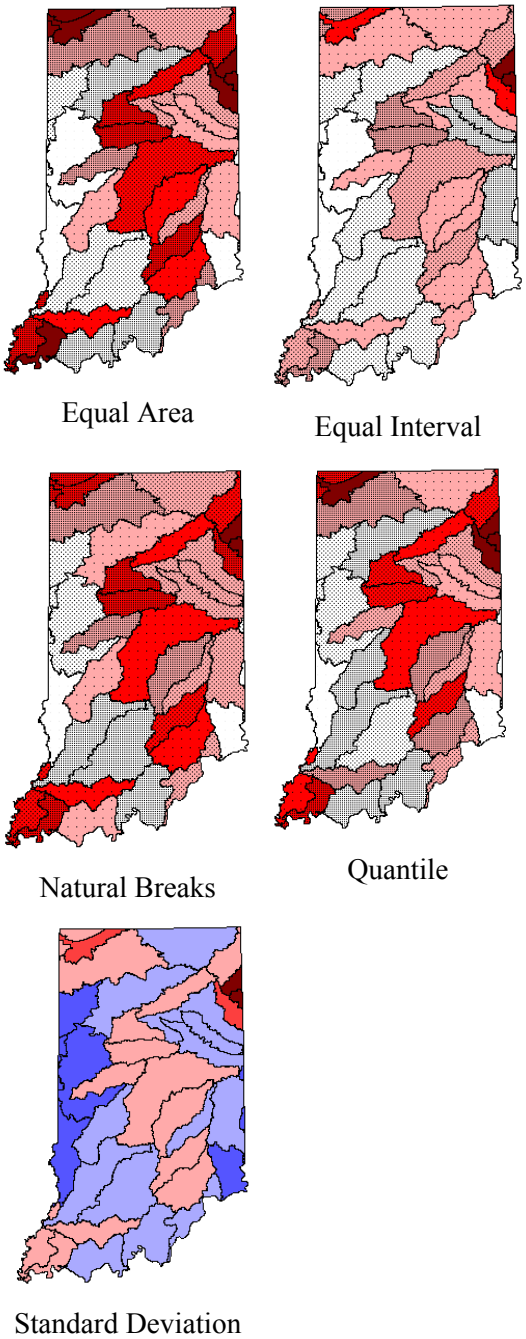
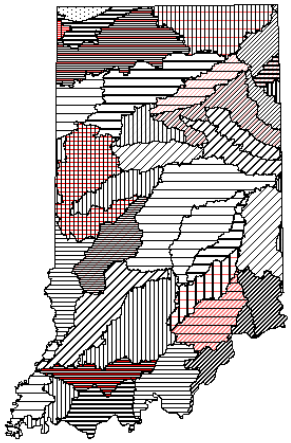


Figure 61. Results showing different spatial variations caused by different classification types.

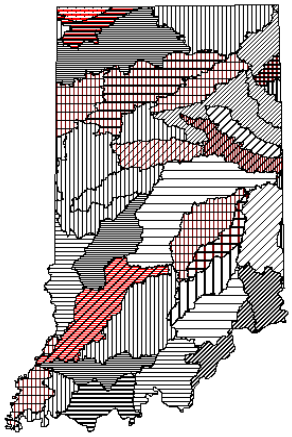
Grids

Grid-cell size can affect the results—too large a cell size and the spatial resolution of the input themes will be lost. If the cell size is too small, excess computer resources are required to generate the results. The grid-cell size selected should be adequate to maintain the spatial resolution of important, smaller features within the input themes. Tests indicate 50-meter grid size is adequate to maintain the spatial resolution of the 14-digit hydrologic units. The statistic used to combine the grids also will affect the results. This statistic tells the application how to combine the products of the rank values and weight factors that are the individual grid-cell values. Although other statistics are available, using the sum statistic to combine grids is strongly recommended.

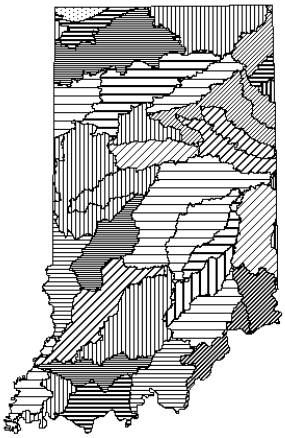
The statistical method used to summarize the individual grid cells within a feature is another input that can affect the theme generated by the application. Figure 62 shows the results for the output theme present in Figure 56 for six statistics that can be used to summarize grid cells and the sum statistic normalized by the acres in the hydrologic unit. The results of the Min, Max, Mean, Sum, and Sum normalized by the acres in the hydrologic unit are shown overlain by the 8-digit-hydrologic-unit test theme. Each option gives a different result. When the number of grid cells is large (small cell size), the mean gives the best representation of the original data. At other times, however, the user should consider using the sum of the grid points normalized by the area of the features. An understanding of the data sets and their spatial resolution and variability is important when selecting the statistic to summarize the combined grid or to analyze the application results.



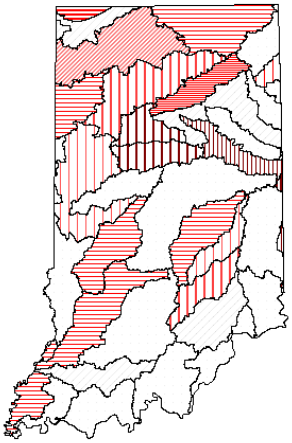
Min



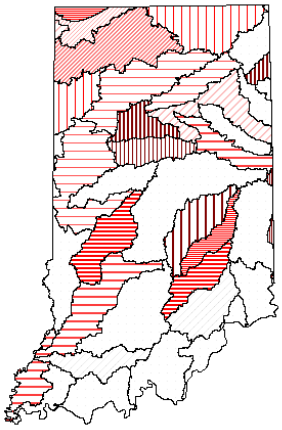
Max



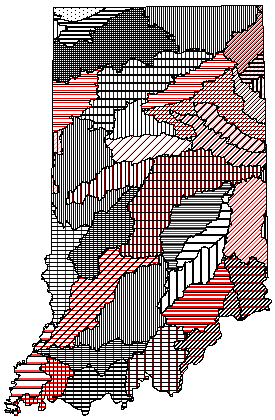
Mean



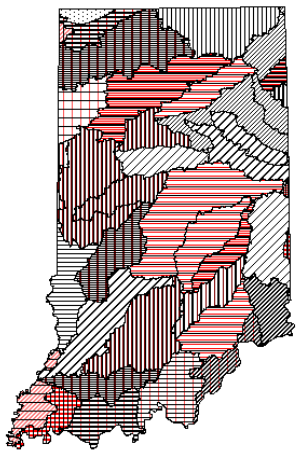
Range



Standard Deviation



Sum



Sum normalized by Acres

Figure 62. The results of using various statistics to summarize the grid points in an area.

THE NONPOINT-SOURCE POLLUTION EXAMPLE

The U.S. Environmental Protection Agency (USEPA) requires each state to maintain and routinely update a nonpoint-source pollution management plan. Each plan contains a list of water bodies affected by nonpoint-source pollution. Projects receiving USEPA nonpoint-source pollution grant monies are required to have the project site located at a water body on that list. IDEM's Nonpoint-Source Pollution Section, now the Watershed Management Section, created a task force of volunteers from Federal, State, and local governments and from the private sector to assist in revising Indiana's Nonpoint-Source Pollution Management Plan. In addition to a list of Indiana water bodies affected by nonpoint-source pollution, the task force wanted the list prioritized by the effects of nonpoint-source pollution on those water bodies. This prioritization would allow IDEM to identify water bodies in Indiana where the benefits of implementing a pollution management plan are likely to be greater in comparison to other areas. The task force formed the Prioritization of Water Bodies Subcommittee to produce this prioritized list of Indiana water bodies affected by nonpoint-source pollution.

The Prioritization of Water Bodies Subcommittee evaluated the assigned task and developed a qualitative technique for evaluating the potential for nonpoint-source pollution to affect a watershed. The qualitative technique involved analyzing data layers for which the differences among areas (the spatial variation), when combined, likely would reflect the spatial variation in the potential for nonpoint-source pollution to affect those watersheds. Each water body in a watershed was assigned the estimated potential for the watershed in which the water body was located. The Subcommittee then developed a process to allow an annual update to the estimated potential for nonpoint-source pollution to affect individual water bodies on the basis of an analysis of actual water-quality data.

Methodology

Several issues required resolution before the Subcommittee could apply this qualitative technique. The individual data layers showed information on a county or 8-digit-hydrologic-unit watershed basis. Because IDEM was interested in representing its findings based on watersheds, a process for converting data layers representing counties to data layers representing 8-digit hydrologic units was needed. This conversion was done, using an area-weighted average. That is, for each county in an 8-digit hydrologic unit, the data value for the county was multiplied by the area of the county within the 8-digit hydrologic unit and then divided by the area of the 8-digit hydrologic unit. The data value for the 8-digit hydrologic unit was the sum of the area-weighted county values within that 8-digit hydrologic unit.

Because each data layer had different units of measure, a method to standardize the units was needed. Rank values were used to give each data layer a unitless component that could be used to combine the data layers. The rank values were determined using the following process for each data layer. The data value for each geographical area, county, or watershed was normalized using the size of the area and the 95th, 66th, 50th, 33rd, and 5th percentiles were calculated using the normalized data values. Each geographical area in the data layer where the normalized value was equal to or greater than the 95th percentile was assigned a rank value of 6. A rank value of 5 was assigned to areas where the normalized value was greater than or equal to the 66th percentile but less than the 95th percentile. This assignment of rank values continued until areas where the normalized value was less than the 5th percentile were assigned a rank value of 1.

The data layers selected to represent indicators of nonpoint-source pollution were combined to create a single data layer that could be used to prioritize the list of Indiana's water bodies. The rank values for each county in the State were aggregated to create a new, but temporary, data layer. The aggregated county data layer was converted to the 8-digit-hydrologic-unit watersheds using an area-weighted average. The final data layer showed an estimate of the relative potential for nonpoint-source pollution to affect the watersheds of Indiana but did not show the actual potential for nonpoint-source pollution. The technique's real value to scientists and water managers is the relative difference among watersheds which, when analyzed, can be used to identify areas of potential concern.

Primary Sources of Nonpoint-Source Pollution

The Prioritization of Water Bodies Subcommittee of the Nonpoint-Source Pollution Management Plan Task Force decided the primary sources of nonpoint-source pollution in Indiana were related to population centers, agriculture, and erosion.

Population Centers

The Subcommittee thought runoff from increased impervious areas and increased usage of fertilizers and pesticides to maintain lawns, septic systems, and combined-sewer overflows were the primary sources for nonpoint-source pollution from population centers. To represent the potential effect of a population center on a watershed, population estimates for 1995 and the change in population between 1990 and 1995 were examined. The Subcommittee decided that population density alone did not give an accurate picture of the potential for nonpoint-source pollution to affect a county because, commonly, people live in one county and work in another. To account for this factor, a representation for the number of individuals employed in a county during 1993 was computed in each of the following categories:

Agriculture	Construction	Financial	Manufacturing
Mining	Nonclassified Businesses	Retail	Service Industry
Transportation	Wholesale	Total Employed in County	

The Subcommittee used the population density for 1995 and the representation of the number of individuals employed in a county to represent the potential effect of nonpoint-source pollution from population centers.

Agriculture

The potential for nonpoint-source pollution from agriculture was thought to come from two areas, row-crop production and livestock operations. The principal potential from row-crop production is related to applying fertilizers and pesticides, and the potential from livestock operations is related to animal waste and pasturing. To represent the potential effect from row-crop production, the Subcommittee examined the acres and percent of a county planted in the following crops during 1995:

Corn Wheat Soybeans Corn, soybeans, and wheat

Corn, wheat, and soybeans are the principal row crops in Indiana. After evaluating the various data layers, the Subcommittee decided that the percent of the county planted in corn, wheat, and soybeans best represents the potential effect of nonpoint-source pollution from row-crop production.

To qualify the potential for nonpoint-source pollution from livestock operations, the following data layers for 1992 were examined.

Head of:

Sheep Horses Swine Poultry Cattle Dairy cattle Non-dairy cattle

Factors estimating the quantity of water consumed by different types of livestock (Arvin, 1993) were used to calculate an estimate of the gallons of water consumed per day by swine, poultry, dairy cattle, and non-dairy cattle.

The following nutrients generated by animal waste also were examined. Pounds of:

Nitrogen Phosphorus Potassium Nitrogen, phosphorus, and potassium

The percentages of a crop's need for the following nutrients, met by animal waste produced in a county, also were examined.

Nitrogen Phosphorus Potassium

After examining the factors related to livestock operations, the potential benefits gained from application of manure as a fertilizer had to be considered. The Subcommittee decided that the pounds of nitrogen, phosphorus, and potassium generated by animal waste per acre of corn, soybeans, and wheat best reflected the potential for nonpoint-source pollution from livestock operations. Several assumptions were made. As the quantity of nutrients per planted acre decreases, the potential for excess nutrients decreases and the potential for nonpoint-source pollution decreases. The Subcommittee assumed that all animal waste was spread on fields and that the quantity of fertilizer was not affected. (This animal waste does not account for the application of manure transported to an area from other parts of the State.)

Erosion

The effects of erosion in Indiana were thought to be of such magnitude that erosion should be considered separately. Erosion was thought to be generated by two primary activities: urban growth and agriculture. No estimate has been made of erosion caused by urban growth in Indiana. The population change between 1990 and 1995 was thought to reflect the potential for erosion from urban growth. The assumption is that as population increases, the construction of homes, roads, businesses, and shopping centers increases the potential for erosion.

Because no-till practices are increasing in Indiana, the previously discussed data layers associated with row-crop production most likely do not adequately represent the potential for erosion from agriculture. Estimates for soil loss from agricultural areas throughout Indiana were used to qualify the potential for erosion from agricultural practices.

Additional Areas of Concern Regarding Nonpoint-Source Pollution

In addition to the potential effects of the previously discussed factors, withdrawals from surface- and ground-water sources were included to identify areas where the effect of nonpoint-source pollution could have a more serious effect. These areas are where the direct effect on human populations will most likely occur. Estimates of the surface water, ground water, and total water used for public supply for 1995 were reviewed, as was the percent of the population served by public and domestic supplies. The number of surface- and ground-water suppliers was identified.

Many suppliers of ground water in Indiana are small noncommunity suppliers who, in general, do not have access to the same testing and monitoring resources as larger community suppliers. The Subcommittee considers these small, noncommunity surface- and ground-water suppliers to be more vulnerable; therefore, these data were the most appropriate for the analysis.

SUMMARY

The U.S. Geological Survey (USGS) developed a geographical information system application to combine data layers showing natural resources, physical characteristics, and cultural features into a single data layer. The application was developed in cooperation with Indiana Department of Environmental Management (IDEM). Scientists and water managers can use the combined data layer to identify areas where the potential

cumulative effect of the data layers combined is relatively greater in one area when compared to other areas within the study area. The application can be used in many areas by changing the spatial focus of the data layers; the application can address other issues by changing the focus of the data layers included in the analysis.

The application developed using Environmental System Research Institute's ArcView with the Spatial Analyst extension:

- allows the selection of a watershed of interest,
- allows the selection of the number of classes to define the spatial variability of the themes,
- allows the assignment of weight factors to signify the relative importance of one theme compared to another,
- allows the selection of the grid-cell size used to convert themes to grids,
- converts the themes to a grid whose cell values are the product of the rank values and weight factors, and
- allows the selection of the statistics used to combine the grids and to summarize the grids into watersheds.

The application is based upon the work of IDEM's Prioritization of Water Bodies Subcommittee, Nonpoint-Source Management Plan Task Force.

SELECTED REFERENCES

Arvin, Donald V., 1993, Quality-assurance plan for the U.S. Geological Survey—Indiana District Water-Use Program, U.S. Geological Survey Open-File Report 93-88, 31 p.

Hutchinson, Scott and Daniel, Larry, 1997, Inside ArcView GIS: Santa Fe, N.M., OnWord Press, 474 p.

Mitchel, James E., 1993, A Characterization of the influence of (x,y) uncertainty on predicting the form of three-dimensional surfaces in American Water Resources Association Symposium Proceedings, March 14–17, 1993: Geographical Information Systems and Water Resources.

Mitchel, James E., 1996, Effect of spatial resolution on estimating hydrologic response and economic value of an urban forest in American Water Resources Association Symposium Proceedings, September 22–16, 1996: GIS and Water Resources.

Razavi, Amir H., 1997, ArcView GIS/Avenue developer's guide: Santa Fe, N.M., OnWord Press, 407 p.

Razavi, Amir H., and Warwick, Valerie, 1997, ArcView GIS/Avenue programmer's reference: Santa Fe, N.M., OnWord Press, 524 p.

U.S. Bureau of the Census, 1994, 1992 Census of agriculture, Geographic Area Series, Indiana State and County Data: U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census, v. 1, pt. 14.

U.S. Bureau of the Census, 2000, County population estimates and demographic components of population change: Annual Time Series, July 1, 1990 to July 1, 1999, from URL http://www.census.gov/population/estimates/county/co-99-8/99c8-18.txt, accessed April 3, 2000, text format.

U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual employment by industry: Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format.

APPENDIXES

APPENDIX 1. LISTING OF THE CODE

EV.AddTheme

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:29:46 2000

'This script causes "Add Theme" to look in the
'database accompanying this application the first
'time. "Add Theme" is executed after each start of the
'application.

'**********************************
'
'ev_data -- Variable holding the full pathname to the
'          data base that accompanies the application.
'_timesin -- Global variable used to count the number
'           of times "Add Theme" has been accessed
'           during this execution of the
'           application.
'
'**********************************

'First time "add theme" has been activated for this
'execution of the project. Look in the ev_data
'directory for the themes. Increment the global
'variable _timesin.


if (_timesin = 0) then
  ev_data = System.GetEnvVar("EV_DATA")
  ev_data.asFilename.SetCWD
  _timesin = _timesin + 1
end


'Working directory set, now run ESRI's View.Add.


av.Run("View.Add","")
```

EV.CreateTheme

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:29:58 2000

'This script is used to create a new theme using a
'base theme and a data file.

'*****************************************
'
'asrcnm -- Variable containing the source name, full
'         pathname, to the base theme.
'EV_DATA -- System environment variable holding the
'         full pathname to the data base that
'         accompanies the application.
'ev_data -- Variable holding the full pathname to
'         the data base that accompanies the
'         application.
'EV_HOME -- System environment variable holding the
'         full pathname to the user area where
'         files are saved.
'ev_home -- Variable holding the full pathname to
'         the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'         full pathname to the temporary work
'         area.
'ev_temp -- Variable holding the full pathname to
'         the temporary work area.
'evctBaBase -- The base theme, created using asrcnm.
'evctBaCodeField -- Field in the base theme
'         attribute table on which the table
'         join will occur.
'evctBaCodeSel -- Field type used during a county
'         join.
'evctBaList -- List containing the types of base
'         theme that can be used to create a new
'         theme.
'evctBaSel -- Type of base theme selected by the
'         user.
'evctBaTheme -- Name of the base theme shapefile.
'evctCntyFields -- List containing the allowable
'         field types on which a county join can
'         occur.
'evevenvDic -- Dictionary containing the system
'         environment variables, returned from
'         "EV.GetEnvVar."
'evctFile -- Individual file from the list "evctFiles,"
'         used to loop through all files in
'         "evctFiles."
'evctFileName -- Name of the join table, entered by
'         the user.
'evctFiles -- Files, selected by the user, that will
'         be joined to the base theme.
'evctFileVTab -- VTab created using a user defined
'         data file contained in evctFiles.
'evctFrField -- Field in the user defined data file that
'         will be used to join the data file to the
'         base theme.
'evctFrFields -- Fields contained in the user's data
'         file.
'evctjoinFTab -- Copy of the joined table used to
'         make the joined theme.
'evctJoinTheme -- Theme made using "evctjoinFTab."
'evctLabels -- List of file types that can be
'         used during a join.
'evctPatterns -- List of patterns used to find the
'         files that can be used during a join.
'evctToField -- Field in the base theme on which the
'         join will occur.
'evctTmpName -- Temporary name of the joined table,
'         created by the application.
'evctView -- Active view, the view in which the new
'         theme will be visible.
'evctVTab -- The base theme's FTab.
'frFieldPrec -- Precision of field in the data table
'         that will be used during the join.
'frFieldType -- Type of field in the data table that
'         will be used during the join.
'frFieldWidth -- Width of the field in the data table
'         that will be used during the join.
'toFieldPrec -- Precision of the field in the base theme
'         that will be used during the join.
'toFieldType -- Type of field in the base theme that
'         will be used during the join.
'toFieldWidth-- Width of the field in the base theme
'         that will be used during the join.
'
'*****************************************

'*****************************************
'
'Get the system environment variables and the view
'with which to work.
'
'*****************************************

'Get the system environment variables.


evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("EV_TEMP")


'Get the view with which to work.


evctView = av.Run("EV.ViewGet","")
```

```
'*****************************************
'
'Find and make the base theme.
'
'*****************************************

'Get the type of base theme that will be used to
'create a new theme.


evctBaList = {"County",
              "8-digit Hydrologic Units",
              "11-digit Hydrologic Units",
              "14-digit Hydrologic Units"}
evctBaSel = MsgBox.ListAsString(evctBaList,
            "Please Select the base area theme " +
            "you wish to use.",
            "Select type of base theme.")
if (evctBaSel = NIL) then
  MsgBox.Error("You can only create a new theme " +
              "with county or watershed data.",
              "Selecting type of base theme.")
  return NIL
end


'On the basis of the type of base theme selected,
'identify the name of the base theme and the name
'of the data field on which the table join will
'occur.


if (evctBaSel.Contains("14-digit")) then
  evctBaTheme = "in_hu14.shp"
  evctBaCodeField = "Huc_14"
elseif (evctBaSel.Contains("11-digit")) then
  evctBaTheme = "in_hu11.shp"
  evctBaCodeField = "Huc_11"
elseif (evctBaSel.Contains("8-digit")) then
  evctBaTheme = "in_hu8.shp"
  evctBaCodeField = "Huc_8"
else
  evctBaTheme = "county_utm.shp"
  evctCntyFields = {"FIPS State and County Code",
                    "FIPS County Code",
                    "State of Indiana Codes"}
  evctBaCodeSel = MsgBox.ListasString
                    (evctCntyFields,"Please " +
                    "select the type of county "+
                    "identifier you will " +
                    "use to join your data table.",
                    "Selecting join field.")
  if (evctBaCodeSel = NIL) then
    MsgBox.Error("You must identify the " +
                 "field to use during the " +
                 "table join.",
                 "Selecting join field.")
    return NIL
  end


'The user selected "county" as the type of base theme.
'Get the proper field type, State/County FIPS,
'County FIPS, and Indiana County numbers to use during
'the table join.


  if (evctBaCodeSel.contains
                  ("State and County")) then
    evctBaCodeField = "Statecty"
  elseif (evctBaCodeSel.contains
                    ("FIPS County")) then
    evctBaCodeField = "FIPS_Cnty"
  else
    evctBaCodeField = "IN_Cnty"
  end
end


'Get the base theme.


asrcnm = SrcName.Make(ev_data + "/" + evctBaTheme)
evctBaBase = Theme.Make(asrcnm)

'*****************************************
'
'Find the data file to join to the base theme
'
'*****************************************

'Get the file to join to the base theme.
'Set the current working directory to the user's
'home directory, show the user the available
'files, and let the user select a file.


ev_home.asFileName.SetCWD
evctPatterns = {"*.dbf", "arcdr9", "*.txt","*.cvs",
                "*.*"}
evctLabels = {"dBASE (*.dbf)", "INFO",
  "Delimited Text (*.txt)",
  "Comma Delimited (*.cvs)", "All Files (*.*)"}
```

```
evctFiles = FileDialog.ReturnFiles
  (evctPatterns, evctLabels, "Add Table", 0)



'If the user did not select a file, take no action
'and return to the calling function.


if (evctFiles.Count = 0) then
  MsgBox.Error("You must identify the file which " +
               "contains the data to be joined " +
               "the base area.","Creating themes")
  return NIL
end

'*****************************************
'
'Join the base theme's attribute table and the
'data file, and make the new theme visible in the
'active view.
'
'*****************************************

'Get the FTab for the base theme and the field
'used during the join.


evctVTab = evctBaBase.GetFTab
evctToField = evctVTab.FindField(evctBaCodeField)


'For each file selected by the user, join the
'file to the base theme.


for each evctFile in evctFiles


'Make the VTab of the file.


  evctFileVTab = VTab.Make(evctFile, FALSE, FALSE)


'Check for errors.


  if (evctFileVTab.HasError) then
    if (evctFile.HasLockError) then
      MsgBox.Error
        ("Unable to acquire Read Lock for file " +
          evctFile.GetBaseName, "")
    else
      MsgBox.Error
        ("The file '" + evctFile.GetBaseName +
        "' is not valid.", "")
    end
  else



'Get the name of the fields in the data file and
'have the user select the field to use in the
'join


    evctFrFields = evctFileVTab.GetFields
    evctFrField = MsgBox.ListasString(evctFrFields,
          "Please select the field in your " +
          "data table that corresponds to " +
          evctBaCodeField + " for " + nl +
          evctFile.asString,"Create Theme")


'User canceled the operation, get the next file.


    if (evctFrField = NIL) then
      MsgBox.Error("You keyed cancel. " +
            "Processing of the file " +
             evctFile.asString + " will stop.",
            "Creating Themes")
      continue
    end


'Check the to and from fields to make sure those fields
'have identical formats.

'Check the type.


    toFieldType = evctToField.GetType
    frFieldType = evctFrField.GetType
    if (NOT (toFieldType = frFieldType)) then
      MsgBox.Error("The field types of the " +
            "fields to be used in the join " +
            "differ. They must be the same. ",
            "Create Themes")
      return NIL
    end


'Check the width.


    toFieldWidth = evctToField.GetWidth
    frFieldWidth = evctFrField.GetWidth
```

```
    if (NOT (toFieldWidth = frFieldWidth)) then
      MsgBox.Error("The widths of the fields " +
            "to be used in the join differ. " +
            "They must be the same. ",
            "Create Themes")
      return NIL
    end


'Check the precision.

    toFieldPrec = evctToField.GetPrecision
    frFieldPrec = evctFrField.GetPrecision
    if (NOT (toFieldPrec = frFieldPrec)) then
      MsgBox.Error("The precision of the fields " +
            "to be used in the join differ. " +
            "They must be the same. ",
            "Create Themes")
      return NIL
    end


'Join the base theme attribute table and the
'data file.

    evctVTab.Join(evctToField,evctFileVTab,
              evctFrField)
  end
```

```
'Let the user give the new theme a name.

  evctTmpName = av.GetProject.MakeFileName
                ("wse_join","shp")
  evctFileName = FileDialog.Put(evctTmpName,
                "wse_j*.shp","Save " +
                evctFile.asString +
                " as Theme")


'Export the joined table to the new file.

  evctjoinFTab = evctVTab.Export(evctFileName
                          ,shape,FALSE)


'Make the joined table a theme and add it to the
'view.

  evctJoinTheme = FTheme.Make(evctjoinFTab)
  evctView.AddTheme(evctJoinTheme)
  evctJoinTheme.SetVisible(TRUE)
  evctJoinTheme.SetActive(TRUE)


'Remove the joins from the base theme table
'and get the next file.

  evctVTab.UnjoinAll
end
```

EV.Evaluate

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:30:16 2000

'This script combines the themes contained in a
'scenario file into a single theme. The themes must
'be in the active view's table of content.

'*****************************************
'
'aGridList -- List containing the grids to be combined.
'aStat -- The statistic chosen by the user to combine
'        the grids.
'aStatList -- List containing the statistics that
'        can be used to merge the grids.
'aTheme -- The theme being worked on. Used to loop
'        through "evevActiveThemes."
'aThemeName -- Name of the theme being converted to
'        a grid.
'aThemeWFDic -- Dictionary containing the themes and
'        weight factors.
'choice_list -- List of statistics that can be used
'        to summarized "theRstGrid."
'EV_DATA -- System environment variable holding the
'        full pathname to the data base that
'        accompanies the application.
'ev_data -- Variable holding the full pathname to
'        the data base that accompanies the
'        application.
'EV_HOME -- System environment variable holding the
'        full pathname to the user area where
'        files are saved.
'ev_home -- Variable holding the full pathname to
'        the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'        full pathname to the temporary work
'        area.
'ev_temp -- Variable holding the full pathname to
'        the temporary work area.
'evevActiveThemes -- A list of themes to be combined.
'evevAllThemes -- All themes in the view table of
'        content.
'evevenvDic -- Dictionary containing the system
'        environment variables, returned from
'        "EV.GetEnvVar."
'evevFactor -- Statistic used to make the watershed
'        of interest visible.
'evevField -- Field name containing "Rank_Value."
'evevFields -- Temporary list containing the fields
'        contained in the grid "theRstGrid" and the
'        watershed of interest.
'evevFN -- Temporary file used to summarize
'        "theRstGrid."
'evevFrField -- Field on which the summarization was
'        done, used to join the summarized table to
'        the watershed of interest.
'evevFTab -- A temporary FTab, containing ranks, of
'        the theme being converted to a grid and
'        the FTab of the watershed of interest.
'evevGrid -- Grid of the theme being processed.
'evevGridDic -- Dictionary containing the grids of the
'        themes included in the scenario being
'        evaluated.
'evevGthm -- "theRstGrid" converted to a grid theme.
'evevHash -- A hash used to identify the watershed of
'        interest.
'evevLegend -- Legend associated with "evevRTheme."
'evevMaxExt -- Maximum extent of the active view.
'evevMaxRank -- Maximum rank value, the maximum
'        number of classes used to define the
'        spatial variability of the themes in
'        "evevActiveThemes."
'evevMFTheme -- The theme created using the table
'        created when the watershed of interest and
'        the summarization tables were joined.
'evevPrj -- Map projection of the active view.
'evevRTheme -- The theme created when
"evevMFTheme"
'        is converted to a shapefile.
'evevStatus -- Status of the just converted grid.
'evevSumField -- Field name in the watershed of
'        interest that will be the basis for
'        summarizing the grid.
'evevSumVTab -- VTab containing the summarization of
'        "theRstGrid" using the watershed of
'        interest.
'evevToField -- Field containing the value that will
'        be used when the grid is summarized. Also,
'        the field used to join the summarized
'        table to the watershed of interest.
'evevView -- Active view, the view in which the themes
'        being combined exists, and the merged
'        theme will be visible.
'evwoiTheme -- Watershed of interest theme selected
'        by the user from "evwoiThemes."
'evwoiThemes -- List of themes that could be a
'        watershed of interest theme.
'findTheme -- True if the theme is in the view, false
'        if the theme is not in the view.
'passDic -- Dictionary used to pass information
'        to the scripts called in this script.
'theCellSize -- Cell size as entered by the user after
'        the cell has been checked for validity.
'theFirstGrid -- The first grid in the grid list. All
'        other grids in the list will be combined
'        with this grid using the enumeration
'        "theStat."
```

```
'theRstGrid -- The combined grid.
'theScenarios -- Name of the file that contains a
'          list of files and descriptions. These
'          files contain the list of themes and
'          weight factor that make up a scenario.
'theStat -- Enumeration of "aStat."
'tmpAns -- Temporary variable used in loops
'tmpBitMap -- Bit map containing the features
'          in the watershed of interest to be
'          used in the summarization.
'tmpCellSize -- Cell size as entered by the user.
'tmpDic -- Temporary dictionary containing the
'          contents of the dictionary returned from
'          a script.
'tmpExtent -- Extent of the theme being converted to
'          a grid.
'tmpfactor -- User selection from "choice_list" that
'          will be used to make the watershed of
'          interest theme visible.
'tmpVal -- Temporary value used to identify the huc
'          level of the watershed of interest.
'
'*****************************************

'*****************************************
'
'Get the system environment variables and the view
'with which to work.
'
'*****************************************


'Get the system environment variables.


evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("TEMP")
ev_cwd = FileName.GetCWD
ev_pwd = av.GetProject.GetWorkDir
ev_home.asFileName.SetCWD
av.GetProject.SetWorkDir(ev_home.asFileName)


'Get the active view that contains the themes to be
'evaluated.


evevView = av.Run("EV.ViewGet","Evaluate")

'*****************************************
'
'Get the scenario file that contains the themes to
'be combined and their weight factors.
'
'*****************************************


'Get the file that contains the names of the files
'that contain the lists of themes and weight
'factors that will be used in this screening.


theScenarios = av.Run("EV.WFGetScenarios","Select")

if (theScenarios = NIL) then
  MsgBox.Error ("No list of scenarios was selected.",
          "Getting list of scenarios.")
  return NIL
end


'Get the list of themes and the weight factors
'selected by the user. If there are none, return to
'the calling script.

passDic = Dictionary.Make(2)
passDic.Set("Scenario",theScenarios)
passDic.Set("View",evevView)
tmpDic = av.Run("EV.WFGetTheme",passDic)
if (tmpDic.Count <= 0) then
  MsgBox.Error("Did not return any themes and " +
          "weight factors.","Retrieving " +
          "weight factors.")
  return NIL
end
passDic.Empty

'Get the dictionary that has the theme as the key
'and the weight factor as the value.


aThemeWFDic = tmpDic.Get("WFDIC")
if ((aThemeWFDic = NIL) or (aThemeWFDic.Count <=
0))
    then
  MsgBox.Error("Did not return any themes and " +
          "weight factors.","Getting " +
          "weight factors.")
  return NIL
end


'Show the themes and weight factors to the user
'and allow the user to change the weight factors.
'Changing the weight factors is done in the script
'"EV.WFAssignFactor."


tmpDic =
```

```
av.Run("EV.WFAssignFactor",aThemeWFDic)
if ((tmpDic = NIL) or (tmpDic.Count <= 0)) then
  MsgBox.Error("Did not return any themes and " +
            "weight factors.","Retrieving " +
            "weight factors.")

  return NIL
end
aThemeWFDic = tmpDic

'*****************************************
'
'Verify that the themes to be combined are in the
'active view.
'
'*****************************************

'Get the list of themes from the scenario file.


evevActiveThemes = aThemeWFDic.ReturnKeys

'Make sure the theme is in the view.


for each aTheme in evevActiveThemes
  findTheme = evevView.FindTheme(aTheme.asString)
  if (findTheme = NIL) then
    MsgBox.Error ("Could not find theme " +
              aTheme.asString + " in the " +
              "View " + evevView.asString +
              ". Please add the theme to " +
              "the view and try again.",
              "Checking Themes and Weight " +
              "Factors.")
    return NIL
  end
end

'*****************************************
'
'Establish the parameters that will control the
'combining of themes during this execution.
'Make a dictionary to pass this information to
'various scripts, which establish other parameters or
'convert the themes to grids.
'
'*****************************************

passDic.SetSize(6)
passDic.Set("View",evevView)
passDic.Set("ThWf",aThemeWFDic)


'Get the maximum rank value.

evevMaxRank = av.Run("EV.EvMaxRankValue",
                passDic)



'Make sure there is maximum rank value.


if (evevMaxRank = NIL) then
  MsgBox.Error("Unable to determine the " ++
            "maximum rank value.",
            "Finding maximum rank value.")
  return NIL
end
passDic.Set("MR",evevMaxRank)


'Get the watershed of interest theme that
'is used to represent the combined theme.
'First find all themes that contain the column WOI.


evwoiThemes = {}
evwoiThemes = av.Run("EV.WOIGetWOI",
evevView.GetThemes)

'There is at least one watershed of interest
'theme, that is the FTab contains a column with a
'field name of "WOI." Ask the user to select which
'theme to use.


evwoiTheme = NIL
if (evwoiThemes.Count = 1) then
  evwoiTheme = evwoiThemes.Get(0)
elseif (evwoiThemes.Count > 1) then
  evwoiTheme = MsgBox.List(evwoiThemes,
            "Which of the following " +
            "watershed of interest themes do " +
            "wish to use. Click Cancel to " +
            "end this execution.",
            "Selecting watershed of interest.")
end


'There are no watershed of interest themes.
'Exit the script.


if (evwoiTheme = NIL) then
  evevTheme = MsgBox.Error("There are no " +
            "watershed of interest themes," +
            "or you did not choose one." +
```

```
            "Click Cancel to " +
            "start over.",
            "No watershed of interest.")
  return NIL
end
passDic.Set("WT",evwoiTheme)



'Get the maximum extent of all active themes.
'This extent will be used to make the grids.


evevMaxExt = av.Run("EV.EvGetMaxExtent",passDic)
passDic.Set("ME",evevMaxExt)

'Ask the user to enter a cell size to be used when
'the theme is converted to a grid.


tmpAns = TRUE
while (tmpAns)
  tmpCellSize = MsgBox.Input("Please enter the " +
                "cell size to be used " +
                "when converting the theme " +
                "to a grid.",
                "Enter cell size.","500")
  if (tmpCellSize = NIL) then
    MsgBox.Error("A cell size must be entered.",
              "Entering cell size.")
    return NIL
  end
  if (tmpCellSize.IsNumber) then
    theCellSize = tmpCellSize.asNumber
    tmpAns = FALSE
  end
end
passDic.Set("CS",theCellSize)

'*****************************************
'
'Convert the themes to grids.
'
'*****************************************

'Create a dictionary, the key is the theme and the
'value is the grid for that theme.

evevGridDic = NIL
evevGridDic =
Dictionary.Make(evevActiveThemes.Count)


'Clear any selections previously made for this
'watershed of interest.

evwoiTheme.ClearSelection

'Start the loop and convert each theme in the
'scenario file to a grid.



for each aThemeName in evevActiveThemes

'Reset the extent of the theme seen in the view.


  aTheme = evevView.FindTheme(aThemeName)
  tmpExtent = aTheme.ReturnExtent
  evevView.GetDisplay.SetExtent(evevMaxExt.
    Scale(1.1))


'Assign the rank values to each object in the theme.


  passDic.Set("TM",aTheme)
    evevFTab = av.Run("EV.EvAssignRanks",passDic)


  passDic.Set("FT",evevFTab)


'Select the part of the theme that is part of the
'watershed of interest theme.


  aTheme.SelectByTheme(evwoiTheme,
            #FTAB_RELTYPE_INTERSECTS,
            0,
            #VTAB_SELTYPE_NEW)


'Convert the theme to a grid.


  evevGrid = NIL
  evevGrid = av.Run("EV.EvConv2Grid",passDic)


'Clear the theme selection.


  aTheme.ClearSelection
```

```
'Verify the Grid was created properly. If it was,
'add it to the grid dictionary.




 if (evevGrid.HasError) then
  MsgBox.Error(evevTheme.asString +
            "could not be converted " +
            "to a grid","Grid conversion error.")
  return NIL
 end
 evevStatus = Grid.GetVerify
 Grid.SetVerify(#GRID_VERIFY_OFF)
 Grid.SetVerify(evevStatus)
 evevGridDic.Set(aTheme,evevGrid)
 evevView.GetDisplay.SetExtent(tmpExtent.Scale(1.1))


' Remove the Rank Column from the table.


 av.Run("EV.RemoveRankField",aTheme)


' Get the next theme.


end


'*****************************************
'
'Combine the grids.
'
'*****************************************

'Choose which statistic to use to combine the grids.

aStatList = {
  "Majority - Value occurring most often within " +
    "the cells.",
  "Maximum - Maximum value for the cells.",
  "Mean - Mean value for the cells.",
  "Median - Median value for the cells.",
  "Minimum - Minimum value for the cells.",
  "Minority - Value occurring least often " +
    "within the cells.",
  "Range - Range in values occurring within the " +
    "cells.",
  "Standard Deviation - Standard Deviation of " +
    "the cell values.",
  "Sum - Sum of the cell values.",
  "Variety - Number of unique occurrences within " +
    "the cells."  }

aStat = MsgBox.ListasString(aStatList,
            "Select the statistic to use when " +
            "combining the grids.","Select " +
            "statistic used to combine grids.")
if (aStat = NIL) then


 MsgBox.Error ("You must choose a statistic for " +
            "the application to use when " +
            "combining grids.","Merge grids " +
            "statistic error.")
 return NIL
end

'Convert the user's selection to an enumeration.


if (aStat.Left(8) = "Majority") then
 theStat = "#GRID_STATYPE_MAJORITY"
elseif (aStat.Left(7) = "Maximum") then
 theStat = "#GRID_STATYPE_MAX"
elseif (aStat.Left(4) = "Mean") then
 theStat = "#GRID_STATYPE_MEAN"
elseif (aStat.Left(6) = "Median") then
 theStat = "#GRID_STATYPE_MEDIAN"
elseif (aStat.Left(7) = "Minimum") then
 theStat = "#GRID_STATYPE_MIN"
elseif (aStat.Left(8) = "Minority") then
 theStat = "#GRID_STATYPE_MINORITY"
elseif (aStat.Left(5) = "Range") then
 theStat = "#GRID_STATYPE_RANGE"
elseif (aStat.Left(8) = "Standard") then
 theStat = "#GRID_STATYPE_STD"
elseif (aStat.Left(3) = "Sum") then
 theStat = "#GRID_STATYPE_SUM"
elseif (aStat.Left(7) = "Variety") then
 theStat = "#GRID_STATYPE_VARIETY"
else
 theStat = "#GRID_STATYPE_MAX"
end


'Create a list of grids to be combined. Get the
'first grid in the list and combine the other grids
'with the first grid using the enumeration.
```

```
aGridList = {}
for each aTheme in (evevGridDic.ReturnKeys)
  aGridList.Add(evevGridDic.Get(aTheme))
end
theFirstGrid = aGridList.Get(aGridList.Count - 1)
aGridList.Remove(aGridList.Count - 1)
theRstGrid = theFirstGrid.
             LocalStats(theStat.asEnum,aGridList)

'Verify that the combined grid was created properly.


if (theRstGrid.HasError) then
  MsgBox.Error(evevTheme.asString +
             "could not be converted " +
             "to a grid","Conversion Error")
  return NIL
end
evevStatus = Grid.GetVerify
Grid.SetVerify(#GRID_VERIFY_OFF)
Grid.SetVerify(evevStatus)


if (theRstGrid.GetVTab <> NIL) then
  evevVTab = theRstGrid.GetVTab
  evevFields = evevVTab.GetFields
  evevField = evevVTab.FindField("Rank_Value")
  if (evevField.IsTypeNumber) then
    evevToField = evevVTab.FindField("Value")
  else
    MsgBox.Error("The field Rank_Value is not " +
              "a number field.",
              "Rank value error.")
    return NIL
  end
end

'Make the combined grid theme.

evevGthm = GTheme.Make(theRstGrid)


'The Grid theme was set to visible as part of the
'development process. It is left in for future
'debugging efforts.
'evevView.AddTheme(evevGthm)
'evevGthm.SetVisible(TRUE)

'*****************************************
'
'Summarize the combined grid and make a summary
'theme by joining the summarization table and the
'watershed of interest attribute table. Export
'that theme as a shapefile.
'
'*****************************************

'Find the field on which the summarization table will
'be joined.

evevFTab = evwoiTheme.GetFTab
evevFields = evevFTab.GetFields
tmpVal = 0
for each aField in evevFields
  if ((aField.asString = "Huc_8") AND (tmpVal < 8))
     then
    tmpVal = 8
    evevSumField = aField
  end
  if ((aField.asString = "Huc_11") AND (tmpVal < 11))
     then
    tmpVal = 11
    evevSumField = aField
  end


  if ((aField.asString = "Huc_14") AND (tmpVal < 14))
     then
    tmpVal = 14
    evevSumField = aField
  end
end


'Make sure the combined grid and the watershed of
'interest intersect.


if (evwoiTheme.ReturnExtent.Intersects
   (evevGthm.ReturnExtent).Not) then
  MsgBox.Error ("The themes in the scenario do not " +
             "overlay the watershed of interest.",
             "Summarizing grids.")
  return NIL
end


'Allow the user to choose which statistic will be
'used to display the theme after the combined grid
'has been summarized.

choice_list = { "Count - Number of cells within " +
             "the area being summarized.",
             "Minimum - Minimum cell value " +
             "within the area being summarized.",
             "Maximum - Maximum cell value " +
             "within the area being summarized.",
             "Range - Range of cell values " +
             "within the area being summarized.",
             "Mean - Mean cell value for the " +
             "area being summarized.",
             "Standard Deviation - Standard " +
```

```
            "Deviation of the cell values " +
            "within the area being summarized.",
            "Sum - Sum of the cell values " +
            "within the area being summarized." }
tmpfactor = MsgBox.ListasString(choice_list,
          "Select the item with which to " +
          "summarize the watersheds.",
          "Determining watershed screening " +
          "factor.")
if (tmpfactor.Left(3) = "Min") then
  evevFactor = "Min"
elseif (tmpfactor.Left(3) = "Max") then
  evevFactor = "Max"
elseif (tmpfactor.Left(5) = "Range") then
  evevFactor = "Range"
elseif (tmpfactor.Left(4) = "Mean") then
  evevFactor = "Mean"
elseif (tmpfactor.Left(8) = "Standard") then
  evevFactor = "Std"



elseif (tmpfactor.Left(4) = "Sum ") then
  evevFactor = "Sum"
else
  evevFactor = "Max"
end


'Get the remaining parameters needed to summarize
'the combined grid.


evevPrj = evevView.GetProjection
evevFN = av.GetProject.GetWorkDir.MakeTmp
        ("zstat","dbf")
tmpBitMap = BitMap.Make(evevFTab.GetNumRecords)
tmpBitMap.ClearAll
evevFTab.SetSelection(tmpBitMap)
evevSumVTab = theRstGrid.ZonalStatsTable
        (evevFTab,evevPrj,evevSumField,FALSE,
         evevFN)

'Verify the summary table was properly created and
'that it contains the fields needed to join the
'summary table to the attribute table for the
'watershed of interest.


if (evevSumVTab.HasError) then
  MsgBox.Error("Cannot summarize the combined grid",
            "Summarizing the combined grid.")
  return NIL
end


if (evevSumVTab.FindField(evevSumField.asString) =
    NIL) then
  MsgBox.Error ("Could not find " +
            evevSumField.asString +
            "in table " +
            evevSumVTab.GetName,
           "Summarizing the combined grid.")
else
  evevFrField = evevSumVTab.FindField
            (evevSumField.asString)
end
if (evevFTab.FindField(evevSumField.asString) =
    NIL) then
  MsgBox.Error ("Could not find " +
            evevSumField.asString +
            "in table " +
            evwoiTheme.GetName.asString,
           "Summarizing the combined grid.")
else
  evevToField = evevFTab.FindField
            (evevSumField.asString)
end


'Join the summary table to the FTab of the watershed
'of interest.

evevFTab.Join(evevToField,evevSumVTab,
           evevFrField)

av.PurgeObjects


‘*****************************************
'
'Export the theme and make the exported theme visible
'with 15 classes. Cleanup.
'
‘*****************************************

'Make the theme, export the theme, and make it
'visible.

evevMFTheme = Ftheme.Make(evevFTab)
passex = Dictionary.Make(2)
passex.Set("Theme",evevMFTheme)
passex.Set("View",evevView)


evevRTheme = av.Run("EV.Export",passex)
av.PurgeObjects
evevLegend = evevRTheme.GetLegend
evevLegend.SetLegendType(#LEGEND_TYPE_COLO
R)
evevLegend.Natural(evevRTheme,evevFactor,15)
evevRTheme.SetLegend(evevLegend)
```

```
evevRTheme.SetLegendVisible(TRUE)
evevRTheme.SetVisible(TRUE)


'Add the hash to identify the watershed of interest.

evevHash = av.Run ("EV.EvHash",evevRTheme)

if (NOT (evevHash = NIL)) then
  evevView.AddTheme(evevHash)
  evevHash.SetVisible(TRUE)
end

'Zoom to the combined and hash themes.

aRect = Rect.MakeEmpty
aRect = aRect.UnionWith(evevRTheme.
        ReturnExtent)
aRect = aRect.UnionWith(evevHash.
        ReturnExtent)
if (aRect.IsEmpty) then


 MsgBox.Info("Rectangle is empty.","")
  return NIL
elseif ( aRect.ReturnSize = (0@0) ) then
  evevView.GetDisplay.PanTo(aRect.ReturnOrigin)
else
  evevView.GetDisplay.SetExtent(aRect.Scale(1.1))
end

'Redraw the view.

av.Run("EV.RedrawView",evevView)

ev_cwd.SetCWD
av.GetProject.SetWorkDir(ev_pwd)

evwoiTheme.GetFTab.Unjoinall
```

EV.EvAssignRanks

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:30:30 2000

'Script is passed a theme and the maximum number
'of classifications. The Script looks at the
'theme legend. If the legend is a graduated color
'the script assigns a value to each feature in
'the theme-based class to which the feature is
'assigned.

'*****************************************
'
'aTheme -- A theme used to loop through
'         "evevarWFDic" to get the weight factor
'         for the theme passed to this script.
'evevarAttTable -- Attribute table for the theme
'         to which ranks are being assigned.
'evevarAttTableFields -- Fields in "evevarAttTable."
'evevarClassField -- Name of the field used, by the
'         legend, to classify the theme.
'evevarField -- The rank value field, float 7 digits
'         with three decimals.
'evevarIndex -- Counter used to loop through each
'         feature in the theme and assign a rank
'         to that feature.
'evevarIsOK -- True or False, "evevarAttTable" can
'         be edited.
'evevarMaxRank -- Maximum rank value for this
'         scenario. Passed from the calling
'         script.
'evevarNormField -- Name of the field used, by the
'         legend, to normalize the data.
'evevarRank -- Rank value for the feature in the
'         theme identified by "evevarIndex."
'evevarTheme -- Theme passed from the calling
'         script. Ranks assigned to this theme.
'evevarWFDic -- Dictionary containing the themes
'         and weight factors for this scenario.
'         Passed from the calling script.
'theClassField -- Data element used to classify
'         the theme.
'theClassIndex -- The class number to which the
'         feature identified by "evevarIndex" belongs.
'theClassValue -- Value of the class for feature
'         "evevarIndex" in the theme.
'theNormField -- Data field used to normalize the
'         data.
'theRankField -- Data field to which the rank is
'         assigned.
'
'*****************************************
'*****************************************
'
' Get the information passed to this script.
'
'*****************************************

'Get the theme, maximum rank value, and the weight
'factors.


evevarTheme = SELF.Get("TM")
evevarMaxRank = SELF.Get("MR")
evevarWFDic = SELF.GET("ThWf")
evevarWF = NIL
for each aTheme in evevarWFDic.ReturnKeys
  if (evevarTheme.asString = aTheme.asString) then
    evevarWF = evevarWFDic.Get(aTheme)
    break
  end
end
if (evevarWF = NIL) then
  MsgBox.Error("No weight factor was assigned to " +
            evevarTheme.asString + ".",
            "Assign rank value error.")
  return NIL
end

'*****************************************
'
' Get the classification and  normalization fields
' used in the legend. These fields will be needed to
' assign the rank values.
'
'*****************************************

'Get the classification and normalization fields.

evevarClassField = evevarTheme.GetLegend
                .GetFieldNames.Get(0)
if (evevarTheme.GetLegend.GetNormType.asString =
    "LEGEND_NORMTYPE_FIELD") then
  evevarNormField = evevarTheme.GetLegend
                .GetNormFieldName
else
  evevarNormField = NIL
end


'Get the list of fields that the legend is
'using to classify the theme. If the number
'of fields found is less than or more than
```

```
'one, stop assigning ranks.

if (evevarTheme.GetLegend.GetFieldNames.Count < 1)
  then
  MsgBox.Error ("There is no classification field" +
    "identified in the Legend.",
    "Assign rank value error.")
end
if (evevarTheme.GetLegend.GetFieldNames.Count > 1)
  then
  MsgBox.Error ("There are multiple " +
              "classification fields identified " +
              "in the legend. There can be only " +
              "one.","Assign rank value error.")
end

'*****************************************
'
' Modify the attribute table to contain a
' rank value field and assign the ranks.
'
'*****************************************

'Identify the table and make it editable.


evevarAttTable = evevarTheme.GetFTab
evevarAttTable.SetEditable (TRUE)


'Get the names of the fields in the tables.


evevarAttTableFields = evevarAttTable.GetFields

'Verify that the table is editable.

evevarIsOK = evevarAttTable.IsEditable
if (evevarIsOK.Not) then
  MsgBox.Error ("Cannot edit the table "
    ++evevarAttTable,"Assign rank value error.")
  return NIL
end


'Determine if the field "rank value" exists in the
'table. If not, add the field.


evevarField = evevarAttTable.FindField
            ("Rank_Value")
if (evevarField = NIL) then
  evevarRankField = Field.Make("Rank_Value",
                    #FIELD_FLOAT,7,3)


  evevarAttTable.AddFields({evevarRankField})
end


'Make sure there are records in the table.


if (evevarAttTable.GetNumRecords = 0) then
  MsgBox.Error ("There are no records to update.",
    "Assign rank value error.")
  return NIL
end


'Get the field names equivalent to the string names.


theClassField = evevarAttTable.FindField
              (evevarClassField)
if (NOT (evevarNormField = NIL)) then
  theNormField = evevarAttTable.FindField
               (evevarNormField)
end
theRankField = evevarAttTable.FindField
               ("Rank_Value")


'For each record in the table assign a rank
'on the basis of the legend.


for each evevarIndex in 0 ..
              (evevarAttTable.GetNumRecords - 1)
  theClassValue = evevarAttTable
                .ReturnValueNumber
                (theClassField,evevarIndex)
  if (NOT (evevarNormField = NIL)) then
    theNormValue = evevarAttTable
                 .ReturnValueNumber
                 (theNormField,evevarIndex)
    theClassIndex = evevarTheme.GetLegend.GetIndex
                 ({theNormValue, theClassValue})
  else
    theClassIndex = evevarTheme.GetLegend.GetIndex
                 ({theClassValue})
  end
  if (evevarTheme.GetLegend.GetNumClasses = 1) then
    evevarRank = evevarMaxRank
  else
    evevarRank = 1 + (theClassIndex *
```

```
                ((evevarMaxRank - 1)/
                 (evevarTheme.GetLegend
                 .GetNumClasses - 1)))
  end
  evevarRank = evevarRank * evevarWF

  evevarAttTable.SetValueNumber
          (theRankField,evevarIndex,evevarRank)
end
```

```
'Stop editing the table and return the attribute
'table with the ranks.


evevarAttTable.SetEditable (FALSE)
return evevarAttTable
```

EV.EvConv2Grid

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:30:42 2000

'Script to convert a theme to a grid.

‘*****************************************
'
'evevcgCellSize -- Grid cell size. Passed from the
'         calling script.
'evevcgField -- Loop variable, used with
'         "evevcgListFields" to check the FTab and
'         the field used to populate the grid.
'evevcgFTab -- FTab of the theme to be converted.
'evevcgGrid -- The grid of the theme passed to this
'         script.
'evevcgListFields -- Field in the FTab being
'         converted. Used to make sure the table
'         has a field that can be converted.
'evevcgPrj -- Map projection of the theme being
'         converted.
'evevcgRect -- Maximum extent to which the
'         the conversion will occur. Passed
'         from the calling script.
'evevcgStatus -- Status of the converted grid.
'evevcgTheme -- Theme being converted. Passed from
'         the calling script.
'
‘*****************************************

‘*****************************************
'
' Get the information passed from the calling
' script.
'
‘*****************************************

'Get the extent, FTab, theme, and cell size for
'this group of themes.


evevcgRect = SELF.Get("ME")
evevcgFTab = SELF.Get ("FT")
evevcgTheme = SELF.GET ("TM")
evevcgCellSize = SELF.GET("CS")
evevcgView = SELF.GET("View")


'Check the FTab passed to make sure a numeric
'field exists that can be used to populate the
'grid.

evevcgListFields = {}
for each evevcgField in evevcgFTab.GetFields
  if (evevcgField.IsVisible and
     (evevcgField.IsTypeNumber or
      evevcgField.IsTypeString)) then
    evevcgListFields.Add(evevcgField)
  end
end
if (evevcgListFields.Count = 0) then
  MsgBox.Error(evevcgTheme.GetName + " does " +
               "not contain a field to " +
               "populate the grid.",
               "Conversion to grid error.")
  return NIL
end

‘*****************************************
'
' Get the other information needed to convert
' the theme to a grid and generate the grid.
'
‘*****************************************

'Get the map projection and the rank value field.


evevcgPrj = evevcgView.GetProjection
evevcgField = evevcgFTab.FindField("Rank_Value")


'Make the grid.

evevcgGrid = Grid.MakeFromFTab
           (evevcgFTab,evevcgPrj,evevcgField,
            {evevcgCellSize, evevcgRect})
if (evevcgGrid.HasError) then
  MsgBox.Error (evevcgTheme.GetName ++
               "could not be converted to " +
               "a grid.","Grid conversion error.")
  return NIL
end
evevcgStatus = Grid.GetVerify
Grid.SetVerify(#GRID_VERIFY_OFF)
Grid.SetVerify(evevcgStatus)


'Check the grid and return it to the calling script.


if (evevcgGrid.GetVTab <> NIL) then
  evevcgVTab = evevcgGrid.GetVTab
  evevcgGthm = GTheme.Make(evevcgGrid)
  if (evevcgField.IsTypeNumber) then
```

```
 evevcgToField = evevcgVTab.FindField("Value")
else
  evevcgToField = evevcgVTab.FindField("S_Value")
  evevcgLegend = evevcgGthm.GetLegend
  evevcgLegend.Unique(evevcgGthm,"S_Value")
  evevcgGthm.UpdateLegend
end
  if (evevcgFTab.IsBase and
     evevcgFTab.IsBeingEditedWithRecovery.Not)
     then
   evevcgVTab.Join(evevcgToField,evevcgFTab,
              evevcgField)
  end
end
return evevcgGrid
```

EV.EvGetMaxExtent

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:30:56 2000

'Get the extent of the watershed of interest.
'This rectangle will be used to focus all
'output.

‘*****************************************
'
'evevgmeRecDic -- Dictionary containing information
'         passed from the calling script.
'evevgmeRect -- Extent used to limit the area of
'         a theme converted to a grid.
'evevgmeWOITheme -- Watershed of interest theme,
'         used to limit the extent of the themes
'         converted to grids.
‘*****************************************
'
‘*****************************************
'
' Get the information passed from the calling
' script. Get the extent of the watershed of
' interest theme and return that extend.
'
‘*****************************************
'Get the information passed from the calling script.


evevgmeRecDic = Self
evevgmeWOITheme = evevgmeRecDic.Get("WT")


'Set and return the extent.


evevgmeRect = NIL
evevgmeRect = evevgmeWOITheme.ReturnExtent
if (evevgmeRect.IsEmpty) then
  MsgBox.Error ("The watershed of interest does " +
            "not have any area.",
            "Watershed of interest extent " +
            " error.")
  return NIL
elseif (evevgmeRect.ReturnSize = (0@0)) then
  MsgBox.Error ("The watershed of interest does " +
            "not have any area.",
            "Watershed of interest extent " +
            " error.")

  return NIL
end
return evevgmeRect
```

EV.Evhash

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:31:06 2000

'This script makes a copy of the theme that was
'passed and uses a legend file that hashes any
'polygons that have the WOI column set to 99.

‘*****************************************
'
'EV_DATA -- Path to the accompanying data base.
'ev_data -- Contains the path to the data base
'         that accompanies the script.
'EV_HOME -- Path to the user's home directory.
'ev_home -- Contains the path to the user's
'         working area.
'EV_TEMP -- Temporary work space.
'ev_temp -- Temporary work space.
'evhLegend -- The hashed legend.
'evhLegendFile -- File containing the hashed legend.
'evevenvDic -- Dictionary containing the system
'         environment variables.
'tmpAns -- Logical variable.
'evhTheme -- Theme that was passed from the calling
'         script.
'
‘*****************************************

‘*****************************************
'
' Get the system environment variables.
'
‘*****************************************


evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("TEMP")

‘*****************************************
'
' Make the legend and get the hash file.
'
‘*****************************************


evhLegend = Legend.Make(#SYMBOL_FILL)
evhLegendFile = (ev_data + "/evhash.avl").asFileName
tmpAns = evhLegend.Load(evhLegendFile,
                  #LEGEND_LOADTYPE_ALL)
if (NOT (tmpAns)) then
  MsgBox.Error ("Could not load the legend file " +
            "needed to highlight the " +
            "watershed of interest after the " +
            "themes have been combined.",
            "Hashing watershed of interest " +
            "error.")
  return NIL
end

‘*****************************************
'
' Copy the theme and use the hashed legend to
' display the copy. SELF is the theme passed from
' the calling script.
'
‘*****************************************


evhTheme = SELF.Clone
evhTheme.SetName("WOI-" + evhTheme.GetName)
evhTheme.SetLegend(evhLegend)

'Return the hash theme.


return evhTheme
```

EV.EvMaxRankValue

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:31:14 2000

'Script finds the maximum rank value by looking at
'the legend entries.

‘*****************************************
'
'aThemeName -- Loop variable, used to loop through
'         "evevmrvThemeList."
'evevmrvDic -- Dictionary containing the information
'         passed by the calling script.
'evevmrvNumClass -- Number of classes in a theme.
'evevmrvRecDic -- Dictionary containing themes and
'         weight factors for this scenario.
'evevmrvTheme -- Theme from "evevmrvView" whose
'         legend is being examined.
'evevmrvThemeList -- List of themes in the
'         scenario.
'evevmrvView -- Active view, passed from the calling
'         script.
'
‘*****************************************

‘*****************************************
'
' Get the information passed from the calling
' script.
'
‘*****************************************

evevmrvDic = SELF
evevmrvView = evevmrvDic.Get("View")
evevmrvRecDic = evevmrvDic.Get("ThWf")
evevmrvThemeList = evevmrvRecDic.ReturnKeys

‘*****************************************
'
' Find the maximum number of classes any theme
' in the list passed from the calling script is
' divided into.
'
‘*****************************************

'Initiate the number of classifications.

evevmrvNumClass = NIL

'Look at each theme passed from the calling script.

for each aThemeName in evevmrvThemeList


'Get the theme.

  evevmrvTheme = evevmrvView.FindTheme(aTheme-
Name)


'Check the Legend to make sure it is type
'"Graduated Color."


  if (NOT(evevmrvTheme.GetLegend.GetLegendType.
        asString = "LEGEND_TYPE_COLOR")) then
    MsgBox.Error (evevmrvTheme.asString ++
      "does not have a Graduated Color Legend Type",
      "Assigning maximum rank value.")
    return NIL
  end


'Get the number of classifications in the theme.
'If the number of classifications in this theme
'is greater than "evevmrvNumClass" set
'"evevmrvNumClass" to the number of classifications
'in this theme.


  if (evevmrvNumClass = NIL) then
    evevmrvNumClass = evevmrvTheme.GetLegend.
                GetNumClasses
  elseif (evevmrvNumClass <
        evevmrvTheme.GetLegend.GetNumClasses)
      then
    evevmrvNumClass = evevmrvTheme.
              GetLegend.GetNumClasses
  end
end


'Return the maximum number of classes.


return evevmrvNumClass
```

EV.Export

```
'Modified from ESRI ArcView View.Export script
'Modified by James (Jay) L. Kiesler, Jr.
'Modified on Wed Apr 26 15:31:41 2000

'*****************************************
'
'evexAttribVis -- True/False is the attribute
'        field visible.
'evexField -- Loop variable, used to loop through
'        the fields in "evexTbl" to find if a
'        field other than shape is visible.
'evexFN -- File name used to save the exported
'        theme.
'evexFTab -- Exported FTab.
'evexFTheme -- Exported Theme.
'evexPrj -- Map projection of the view.
'evexShapeVis -- Is the shape field visible?
'evexShpfld -- Shape field in the "evexTheme."
'evexTbl -- FTab of the theme "evexTheme."
'evexTheme -- Theme to be exported. Passed from
'        the calling script.
'evexWasNotVisible -- Tells whether or not the
'        shape file was visible.
'
'*****************************************

'*****************************************
'
' Get the information passed from the calling
' script.
'
'*****************************************

evexDic = SELF
evexTheme = evexDic.Get("Theme")
evexView = evexDic.Get("View")
evexTbl = evexTheme.GetFTab
evexAttribVis = FALSE

'Get the system environment variables.


evctenvDic = av.Run("EV.GetEnvVar","")
ev_data = evctenvDic.Get("EV_DATA")
ev_home = evctenvDic.Get("EV_HOME")
ev_temp = evctenvDic.Get("TEMP")

ev_cwd = FileName.GetCWD
ev_pwd = av.GetProject.GetWorkDir

ev_home.asFileName.SetCWD
av.GetProject.SetWorkDir(ev_home.asFileName)


'*****************************************
'
' Make sure there is something to export.
'
'*****************************************

'Is there a visible field in the FTab?


for each evexField in evexTbl.GetFields
  if ((evexField.IsVisible) and
     not (evexField.IsTypeShape)) then
    evexAttribVis = TRUE
    break
  end
end


'Is the shape field visible?


evexShapeVis = evexTbl.FindField("Shape").IsVisible


'Nothing is visible. Stop.


if ((evexAttribVis and evexShapeVis).Not) then
  MsgBox.Error("Nothing from the theme is " +
            "visible.",
            "Exporting theme error.")
  return NIL
end

'*****************************************
'
' Export the theme passed from the calling
' script.
'
'*****************************************

'Make a file to hold the exported theme.


evexFN = av.GetProject.MakeFileName
          ("ws_eval", "shp")
if (evexFN = NIL) then
  MsgBox.Error ("Could not create file named " +
             evexFN.asString + "to store " +
             "the exported theme.",
             "Exporting theme error.")
```

```
  return NIL
end


'Find the shape field and determine if it is
'visible.


evexShpfld = (evexTbl.FindField("Shape"))
if (evexShpfld.IsVisible.Not) then
  evexShpfld.SetVisible(evexShpfld.IsVisible.Not)
  evexWasNotVisible = TRUE
else
  evexWasNotVisible = FALSE
end


'Is the view projected?

evexPrj = evexView.GetProjection

'Change the WOI column name to "WSofInt."

evexTbl.SetEditable(TRUE)
evexTbl.StartEditingWithRecovery
newWOIField = Field.Make
            ("WSofInt",#FIELD_SHORT,2,0)
fldList = { newWOIField }
oldWOIField = evexTbl.FindField("WOI")
evexTbl.AddFields(fldList)
numRec = evexTbl.GetNumRecords
for each recNum in 0 .. (numRec - 1)
  old_val = evexTbl.ReturnValue(oldWOIField,recNum)
  evexTbl.SetValue(newWOIField,recNum,old_val)
end
fldList = { oldWOIField }
evexTbl.RemoveFields(fldList)
evexTbl.StopEditingWithRecovery(TRUE)
evexTbl.SetEditable(FALSE)


if (evexPrj = NIL) then
  evexFTab = evexTbl.Export
           (evexFN, Shape,
            evexTbl.GetSelection.Count > 0)
else
  evexFTab = evexTbl.ExportProjected
           (evexFN, evexPrj,
            evexTbl.GetSelection.Count > 0)
end
if (evexFTab.HasError) then
  if (evexFTab.HasLockError) then
    MsgBox.Error("Unable to acquire Write " +
          "Lock for file " + evexFN.GetBaseName,
          "Exporting theme error.")
  else
    MsgBox.Error("Unable to create " +
          evexFN.GetBaseName,
          "Exporting theme error.")
  end
  return NIL
end

‘*****************************************
'
' Clean up the view and return.
'
‘*****************************************

if (evexWasNotVisible) then
  evexShpfld.SetVisible(FALSE)
end

'Build the spatial index, make the theme,
'add the theme to the view, and return the
'exported theme.

evexFTab.CreateIndex(evexShpfld)
evexFthm = FTheme.Make(evexFTab)
evexView.AddTheme(evexFthm)


'Change the WSofInt column name to "WOI."

evexTbl.SetEditable(TRUE)
evexTbl.StartEditingWithRecovery
newWOIField = Field.Make
            ("WOI",#FIELD_SHORT,2,0)
fldList = { newWOIField }
oldWOIField = evexTbl.FindField("WSofInt")
evexTbl.AddFields(fldList)
numRec = evexTbl.GetNumRecords
for each recNum in 0 .. (numRec - 1)
  old_val = evexTbl.ReturnValue(oldWOIField,recNum)
  evexTbl.SetValue(newWOIField,recNum,old_val)
end
fldList = { oldWOIField }
evexTbl.RemoveFields(fldList)
evexTbl.StopEditingWithRecovery(TRUE)
evexTbl.SetEditable(FALSE)


ev_cwd.SetCWD
av.GetProject.SetWorkDir(ev_pwd)
return evexFthm
```

EV.GetEnvVar

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:31:55 2000

'Script to read the system environment variables
'and return a dictionary containing the variable
'values to the calling script. These variables are
'set by the system.

‘*****************************************
'
'envDic -- Dictionary used to return environment
'         variables to the calling script.
'envvar -- System environment variable being
'         retrieved.
'envvarmsg -- Variable error message.
'envvarmsw -- Error message for Microsoft.
'envvarunix -- Error message for a Unix system.
'envvargen -- Generic error message.
'EV_DATA -- System environment variable containing
'          the path to the accompanying data base.
'ev_data -- Variable containing the path to the data
'         base that accompanies the application.
'EV_HOME -- System environment variables containing
'         the path to the user's home directory.
'ev_home -- Variable that contains the path to
'         the user's working area.
'EV_TEMP -- System environment variable containing
the
'         path to a temporary work space.
'ev_temp -- Variable containing the path to the
'         temporary work space.
'
‘*****************************************

‘*****************************************
'
'Make the dictionary used to return the system
'environment variables to the calling script and
'make the operating system specific portion of the
'error message.
'
‘*****************************************

envDic = Dictionary.Make(3)
envvarmsw = "Depending on the Microsoft " +
          "operating sysem environment " +
          "variables are set using the PATH " +
          "command or in the System " +
          "Properties menu."
envvarunix = "System environment variables are " +
           "set using the SETENV command."
envvargen = "See your computer specialist " +
          "if you need assistance."

‘*****************************************
'
'Get EV_DATA -- The path to the data base that
'accompanies this application.
'
‘*****************************************

'Get the environment variable.


ev_data = System.GetEnvVar("EV_DATA")


'If the environment variable did not exist, print
'an error message and end the script.


if (ev_data = NIL) then
  envvar = "EV_DATA"
  envvarmsg = "The environment variable " + envvar +
            " has not been set. " + envvar +
            " is a system environment variable " +
            "and must contain a value before " +
            "this application will function " +
            "properly.  "
  if (System.GetOS.asString = "SYSTEM_OS_MSW")
then
     MsgBox.Error(envvarmsg + envvarmsw,
              "Get environment variables.")
  elseif (System.GetOS.asString =
          "SYSTEM_OS_UNIX") then
    MsgBox.Error(envvarmsg + envvarunix,
              "Get environment variables.")
  else
    MsgBox.Error(envvarmsg + envvargen,
              "Get environment variables.")
  end
  exit
end

‘*****************************************
'
'Get EV_HOME -- The path to the user's work space.
'
‘*****************************************

'Get the environment variable.


ev_home = System.GetEnvVar("EV_HOME")
```

```
'If the environment variable did not exist, print
'an error message and end the project.


if (ev_home = NIL) then
  envvar = "EV_HOME"
  envvarmsg = "The environment variable " + envvar +
          " has not been set. " + envvar +
          " is a system environment variable " +
          "and must contain a value before " +
          "this application will function " +
          "properly.  "
  if (System.GetOS.asString = "SYSTEM_OS_MSW")
then
    MsgBox.Error(envvarmsg + envvarmsw,
              "Get environment variables.")
  elseif (System.GetOS.asString =
         "SYSTEM_OS_UNIX") then
    MsgBox.Error(envvarmsg + envvarunix,
              "Get environment variables.")
  else
    MsgBox.Error(envvarmsg + envvargen,
              "Get environment variables.")
  end
  exit
end


‘*****************************************
'
'Get TEMP or TMP -- The path to temporary
'work space.
'
‘*****************************************

'Get the environment variable.


ev_temp = System.GetEnvVar("EV_TEMP")
if (ev_temp = NIL) then
  ev_temp = System.GetEnvVar("TMP")
end

'If the environment variable did not exist, print
'an error message and end the project.


if (ev_temp = NIL) then
  envvar = "TEMP or TMP"
  envvarmsg = "The environment variable " + envvar +
          " has not been set. " + envvar +
          " is a system environment variable " +
          "and must contain a value before " +
          "this application will function " +
          "properly.  "
  if (System.GetOS.asString = "SYSTEM_OS_MSW")
then
    MsgBox.Error(envvarmsg + envvarmsw,
              "Get environment variables.")
  elseif (System.GetOS.asString =
         "SYSTEM_OS_UNIX") then
    MsgBox.Error(envvarmsg + envvarunix,
              "Get environment variables.")
  else
    MsgBox.Error(envvarmsg + envvargen,
              "Get environment variables.")
  end
  exit
end


‘*****************************************
'
'Populate the Dictionary and return to the
'calling script.
'
‘*****************************************

envDic.Set("EV_DATA",ev_data)
envDic.Set("EV_HOME",ev_home)
envDic.Set("EV_TEMP",ev_temp)
return envDic
```

EV.GetWeightFactors

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:31:55 2000

'Script that allows the user to enter weight
'factors.

'*****************************************
'
'getwfdic -- Dictionary used to pass arguments to
'         this script.
'themeDesc -- List of theme names.
'wfVal -- Default or existing weight factors
'         associated with the themes.
'wfactor -- List of weight factors entered by the
'         user.
'
'*****************************************

getwfdic = SELF
themeDesc = getwfdic.Get("Themes")
wfVal = getwfdic.Get("Weight")

'Show the user the themes and default weight
'factors. Allow the user to change the weight
'factors.


wfactor = MsgBox.MultiInput
  ("Enter the Weighting Factor for each theme",
   "Weighting Factors", themeDesc.asList,
    wfVal.asList)

return wfactor
```

EV.MapMaker

```
'Based on code from "ArcView GIS/Avenue
'Developer's Guide" by Amir H. Razavi

'Created by James (Jay) L. Kiesler, Jr
'Created on Wed Apr 26 15:40:06 2000

'This script creates a map (ArcView) Layout
'of the active view.

'*****************************************
'
'anArrow -- The north arrow shown on the layout.
'aPen -- Pen type used to draw lines.
'aRect -- Size of the rectangle needed to view all
'        visible themes.
'allTitles -- The two-line title for this layout.
'boxFill -- Fill used in the box used to identify
'        the primary view in the locator map.
'EV_DATA -- System environment variable holding the
'         full pathname to the data base that
'         accompanies the application.
'ev_data -- Variable holding the full pathname to
'         the data base that accompanies the
'         application.
'EV_HOME -- System environment variable holding the
'         full pathname to the user area where
'         files are saved.
'ev_home -- Variable holding the full pathname to
'         the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'         full pathname to the temporary work
'         area.
'ev_temp -- Variable holding the full pathname to
'         the temporary work area.
'evevenvDic -- Dictionary containing the system
'         environment variables, returned from
'         "EV.GetEnvVar."
'fullView -- Clone of the view seen in the layout
'        being prepared.
'fullViewGL -- List of graphics included in "fullView."
'infoBox -- Box that contains the scale bar and
'        north arrow.
'infoBoxGr -- Graphic that is the "infoBox."
'lBox -- Graphic box used to identify the primary
'        view in the locator map.
'lFrame -- View Frame that is the locator map.
'lgFrame -- View Frame that is the legend.
'lgRect -- Rectangle that defines where the legend
'        is displayed.
'line1 -- Line that separates the scale bar and
'        north arrow.
'line1Gr -- Graphic that is line 1.
'lRect -- Rectangle that defines the locator map.
'oPt-- Origin point of the layout. Point from
'        which all other points are defined.
'numClass -- Sum of the number of classes in each
'        visible theme.
'pTitle -- Graphic that is the first line of the
'        title.
'marginRect -- Active area of the layout.
'naGr -- Graphic that is the north arrow.
'naRect -- Rectangle that defines the north arrow.
'northArrowFile -- File containing the north arrow
'        definitions.
'northArrowODB -- Object data base of north arrows.
'northArrowList -- List of north arrows.
'sbRect -- Rectangle that defines the scale bar.
'sbFrame -- The graphic that is the scale bar.
'sFill -- Fill for the shadow box.
'shadowGr -- Graphic that is the shadow box.
'sRect -- Rectangle that defines the shadow box.
'subtitle -- Graphic that is the second line of the
'        title.
'subtitleSymbol -- Style of the second title line.
'theDispWidth -- Width of the layout. Used to
'        toggle between portrait and landscape.
'theDispHeight -- Height of the layout. Used to
'        toggle between portrait and landscape.
'theDispPS -- Page size.
'theLayout -- The Arcview layout that will contain
'        the image to be printed.
'theLayoutDisp -- Display for the layout.
'theLayoutGL -- List of graphic objects on the
'        layout.
'thelegxPT -- Starting point for the legend.
'thenarwPT0 -- Starting point of the north arrow.
'thenarwPT1 -- Size of the north arrow.
'theprimVPT -- Starting point for the primary view
'        frame.
'thescalPT0 -- Starting point of the scale bar.
'thescalPT1 -- Size of the scale bar.
'thescnaPT0 -- Corner point of the box to contain
'        the scale bar.
'thescnaPT1 -- Corner point of the box to contain
'        the scale bar.
'thescnaPT2 -- Corner point of the box to contain
'        the scale bar.
'thescnaPT3 -- Corner point of the box to contain
'        the scale bar.
'thesecVPT0 -- Starting point for the locator map.
'thesecVPT1 -- Size of the locator map.
'thesnalPT0 -- Starting point of the line between
'        the scale bar and north arrow.
'thesnalPT1 -- Ending point of the line between
'        the scale bar and north arrow.
'theThemes -- List of all visible themes in the
'        view.
'theTitlePT0 -- Starting point for the first line
```

```
'       of the layout title.

'theTitlePT1 -- Starting point for the second line
'       of the layout title.
'theView -- The active view.
'titleSymbol -- Style for the first line of the
'       title.
'tmpMapDis -- Gets the size of the view frame.
'VisExt -- Extent of the view frame that is visible.
'VisExtW -- Width of the view frame.
'vFill -- Type of fill used in the main view graphic.
'vFrame -- View frame defined using “vRect.”
'vFrameExtent -- Visual extent of the view.
'vRect -- Rectangle defining the main view area.
'vRectGr -- Main view graphic.
'
‘*****************************************

‘*****************************************
'
'Get the system environment variables.
'
‘*****************************************

'Get the system environment variables.


evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("TEMP")

‘*****************************************
'
' Get the active view, define the page type, and
' establish the starting points and size for
' the various components that make up the layout.
'
‘*****************************************

'Get the active view.


theView = av.Run("EV.ViewGet","MapMaker")

'Create the layout


theLayout = Layout.Make
theLayout.SetName (theView.GetName++"Layout")
theLayoutDisp = theLayout.GetDisplay
theLayoutGL = theLayout.GetGraphics


'Set the layout page properties. If the
'extent of the display is wider, then set up
'a landscape page, else use portrait. Use a
'0.50-inch margin.


theDispWidth = theView.GetDisplay.ReturnExtent.
            GetWidth
theDispHeight = theView.GetDisplay.ReturnExtent.
             GetHeight
if (theDispWidth > theDispHeight) then
  marginRect = Rect.MakeXY ( 0.5, 0.5, 10.5, 8.0)
  theDispPS = Point.Make(11.0,8.5)
  thetitlePT0 = Point.Make(1.0,7.0)
  thetitlePT1 = Point.Make(1.25,6.75)
  theprimVPT = Point.Make(7.5,4.0)
  thesecVPT0 = Point.Make(1.0,4.1)
  thesecVPT1 = Point.Make (3.0,2.5)
  thescnaPT0 = Point.Make (4.5,4.3)
  thescnaPT1 = Point.Make (4.5,6.3)
  thescnaPT2 = Point.Make (6.5,6.3)
  thescnaPT3 = Point.Make (6.5,4.3)
  thesnalPT0 = Point.Make (4.5,5.3)
  thesnalPT1 = Point.Make (6.5,5.3)
  thescalPT0 = Point.Make (4.55,4.5)
  thescalPT1 = Point.Make (1.5,0.5)
  thenarwPT0 = Point.Make (4.9,5.5)
  thenarwPT1 = Point.Make (1.0,0.8)
  thelegxPT = 7.8
else
  marginRect = Rect.MakeXY ( 0.5, 0.5, 8.0, 10.5)
  theDispPS = Point.Make(8.5,11)
  thetitlePT0 = Point.Make(1.0,9.5)
  thetitlePT1 = Point.Make(1.25,9.25)
  theprimVPT = Point.Make(5.0,6.0)
  thesecVPT0 = Point.Make(0.5,6.1)
  thesecVPT1 = Point.Make (2,3.0)
  thescnaPT0 = Point.Make (2.7,6.1)
  thescnaPT1 = Point.Make (2.7,8.1)
  thescnaPT2 = Point.Make (4.7,8.1)
  thescnaPT3 = Point.Make (4.7,6.1)
  thesnalPT0 = Point.Make (2.7,7.1)
  thesnalPT1 = Point.Make (4.7,7.1)
  thescalPT0 = Point.Make (2.75,6.3)
  thescalPT1 = Point.Make (1.5,0.5)
  thenarwPT0 = Point.Make (3.6,7.2)
  thenarwPT1 = Point.Make (1.0,0.8)
  thelegxPT = 5.3
end


'Further define the layout display


theLayoutDisp.SetUnits (#UNITS_LINEAR_INCHES)
theLayoutDisp.SetMargin (marginRect)
theLayoutDisp.SetMarginVisible (TRUE)
theLayoutDisp.SetPageSize (theDispPS)
```

```
theLayoutDisp.SetGridActive (FALSE)
theLayoutDisp.SetGridVisible (TRUE)
theLayoutDisp.SetGridMesh (Point.Make(1.0,1.0))



'*****************************************
'
' Get the origin point.
' Get the titles and create their graphics.
'
'*****************************************


'The coordinates are based on the Display frame
'and not the PageDisplay. In order to use the
'coordinates of PageDisplay, the lower left X and Y
'coordinates for the PageDisplay is obtained. In the
'next code line this coordinate is stored in the oPt
'(origin point) object.


oPt = theLayoutDisp.ReturnMarginExtent.ReturnOrigin


'Get the title from the user and add their graphics
'to the layout.  Use Times Roman font and sizes of
'24 and 18.


titleSymbol = TextSymbol.Make
titleSymbol.SetFont (
Font.Make("Times New Roman","Normal") )
titleSymbol.SetSize (24) 'Size is in points.
subtitleSymbol = titleSymbol.Clone
subtitleSymbol.SetSize (18)
allTitles = MsgBox.MultiInput ("Please enter","",
  {"Title:","Subtitle:"},
  {"This is map of",theView.GetName})
if (allTitles.Count > 0) then
   pTitle = GraphicText.Make (
   allTitles.Get(0), oPt+theTitlePT0 )
   pTitle.SetSymbols ({titleSymbol})
   pTitle.SetAngle (0)
   theLayoutGL.Add (pTitle)
   subtitle = GraphicText.Make (
   allTitles.Get(1), oPt+theTitlePT1 )
   subtitle.SetSymbols ({subtitleSymbol})
   subtitle.SetAngle (0)
   theLayoutGL.Add (subtitle)
end


'*****************************************
'
' Get the primary view and add it to the layout.
'
'*****************************************


'The primary-view frame shows the active view at
'its current extent, while the locator-view frame
'shows the active frame zoomed to its extent with
'a rectangle depicting the area of primary view.

' The primary view is placed at the lower left
'corner.


vFill = RasterFill.Make
vFill.SetStyle (#RASTERFILL_STYLE_SOLID)
vFill.SetColor (Color.GetWhite)
vFill.SetOutlined (TRUE)
vFill.SetOLColor (Color.GetBlack)
vRect = Rect.Make ( oPt+Point.Make (0.0,0.0),
                thePrimVPT )
vRectGr = GraphicShape.Make (vRect)
vRectGr.SetSymbol (vFill)
theLayoutGL.Add (vRectGr)
vFrame = ViewFrame.make (vRect)
vFrame.SetSymbol (vFill)
vFrame.SetView (theView,TRUE)
vFrame.SetScalePreserved (FALSE)
theLayoutGL.Add (vFrame)
tmpMapDis = vFrame.GetMapDisplay
VisExt = tmpMapDis.ReturnVisExtent
VisExtW = (VisExt.GetWidth) / 1609.3472


' Add a shadow box to the primary view.


sFill = vFill.Clone
sFill.SetColor (Color.GetBlack)
sRect = vFrame.GetBounds
shadowGr = GraphicShape.Make (sRect)
shadowGr.Offset (Point.Make(0.25,-0.25))
shadowGr.SetSymbol (sFill)
theLayoutGL.UnselectAll
shadowGr.SetSelected (TRUE)
theLayoutGL.Add (shadowGr)
theLayoutGL.MoveSelectedToBack
theLayoutGL.UnselectAll

'*****************************************
'
' Get the view and add the locator map and box
' to the layout.
'
'*****************************************


'Add the locator map at position 1" x 5.5# inches. The
'locator map is based on a cloned view that displays the
```

```
'view's full extent.




lRect = Rect.Make (
oPt+thesecVPT0,thesecVPT1 )
lFrame = ViewFrame.Make (lRect)
lFrame.SetSymbol (vFill)
fullView = theView.Clone
theThemes = fullView.GetVisibleThemes
aRect = Rect.MakeEmpty
numClass = 0
for each aTheme in theThemes
  aRect = aRect.UnionWith(aTheme.ReturnExtent)
  numClass = numClass + aTheme.GetLegend.
                        GetNumClasses
end
if (aRect.isEmpty) then
  fullView.GetDisplay.SetExtent
                  (fullView.ReturnExtent)
elseif (aRect.ReturnSize = (0@0)) then
  fullView.GetDisplay.PanTo(aRect.ReturnOrigin)
else
  fullView.GetDisplay.SetExtent(aRect.Scale(1.1))
end
fullView.SetName("Full View")
lFrame.SetView (fullView, TRUE)
lFrame.SetScalePreserved (FALSE)
theLayoutGL.Add (lFrame)


'Draw a box in the locator map to show
'the extent of the primary view. This box is
'actually drawn on the view display and
'seen on the layout document.


boxFill = vFill.Clone
boxFill.SetStyle (#RASTERFILL_STYLE_EMPTY)
boxFill.SetOutlined (TRUE)
boxFill.SetOlColor (Color.GetBlack)
boxFill.SetOlWidth (4)
vFrameExtent = theView.GetDisplay.ReturnVisExtent
lBox = GraphicShape.Make (vFrameExtent)
lBox.SetSymbol (boxFill)
fullViewGL = fullView.GetGraphics
fullViewGL.Add (lBox)

'*****************************************
'
```

```
' Get the scale bar and north arrow.
'
'*****************************************

'Draw boxes to hold the scale bar and
'the north arrow.


'Draw the box.


aPen = BasicPen.Make
aPen.SetColor (Color.GetBlack)
infoBox = Polygon.Make ( { {oPt+thescnaPT0,
        oPt+thescnaPT1,oPt+thescnaPT2,
        oPt+thescnaPT3 } } )
infoBoxGr = GraphicShape.make (infoBox)
infoBoxGr.SetSymbol (aPen)
theLayoutGL.AddBatch (infoBoxGr)


'Draw a line to separate the scale bar and the
'north arrow.


line1 = Line.Make (oPt+thesnalPT0,oPt+thesnalPT1)
line1Gr = GraphicShape.Make (line1)
line1Gr.SetSymbol (aPen)
theLayoutGL.AddBatch (line1Gr)


'Add scale bar.


sbRect = Rect.Make (oPt+thescalPT0,thescalPT1)
sbFrame = ScalebarFrame.Make (sbRect)
sbFrame.SetUnits (#UNITS_LINEAR_MILES)
sbFrame.SetStyle
(#SCALEBARFRAME_STYLE_ALTFILLED)
sbFrame.SetViewFrame (vFrame)
if (VisExtW <= 10) then
  sbFrame.SetInterval (1)
  sbFrame.SetIntervals (1)
elseif ((VisExtW > 10) AND (VisExtW <= 20)) then
  sbFrame.SetInterval (2)
  sbFrame.SetIntervals (1)
elseif ((VisExtW > 20) AND (VisExtW <= 100)) then
  sbFrame.SetInterval (5)
  sbFrame.SetIntervals (1)
else
  sbFrame.SetInterval (10)
  sbFrame.SetIntervals (1)
end
theLayoutGL.AddBatch (sbFrame)
```

```
'Add north arrow.

naRect = Rect.Make (oPt+thenarwPT0,thenarwPT1)
naGr = NorthArrow.Make (naRect)

'Retrieve a predefined north arrow from
'the north.def file.

northArrowFile = FileName.Make(ev_data +
                        "/north.def")
northArrowODB = ODB.Open (northArrowFile)
if (NIL = northArrowODB) then
  MsgBox.Error
  ("Unable to open north.def object data base",
  "Making a map error.")
  return NIL
end
northArrowList = northArrowODB.Get(0)
anArrow = northArrowList.Get(6)
if (anArrow = NIL) then
 anArrow = northArrowList.Get(0)
end
naGr.SetArrow (anArrow)

'AddBatch is used to add many items to a graphic
'list. The graphics are added to the layout when the
'end batch command is issued.

theLayoutGL.AddBatch (naGr)

'*****************************************
'
' Add the legend.
'
'*****************************************

if (numClass <= 6) then
 lgRect = Rect.Make (
   oPt+Point.Make (thelegxPT,0.0),
   Point.Make (2.2,1.5) )
elseif ((numClass > 6) AND (numClass <= 12)) then
 lgRect = Rect.Make (
   oPt+Point.Make (thelegxPT,0),
   Point.Make (2.2,3.0) )
elseif ((numClass > 12) AND (numClass <= 18)) then
 lgRect = Rect.Make (
   oPt+Point.Make (thelegxPT,0),
   Point.Make (2.2,4.5) )
else
 lgRect = Rect.Make (
   oPt+Point.Make (thelegxPT,0),
   Point.Make (2.2,6.0) )
end
lgFrame = LegendFrame.Make (lgRect)
lgFrame.SetViewFrame (vFrame)
theLayoutGL.AddBatch (lgFrame)

'End the AddBatch and put the graphics on the layout.

theLayoutGL.EndBatch

'*****************************************
'
' Make the layout visible.
'
'*****************************************

theLayoutWin = theLayout.GetWin
theLayoutWin.Open
theLayoutDisp.ZoomToPage
```

EV.OpeningViewShutDownApply

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:40:20 2000

'This script closes the opening view, if the
'view is open, and removes the tool used to close
'the opening view. The script is executed as the
'ApplyEvent for a tool. The tool is activated in the
'start-up script EV.Startup.

'*****************************************
'
'aDocGUI -- Loop variable to examine all GUIs in
'         "evovsdDocGUI."
'aTool -- Loop variable to examine all tools in
'         "evovsdTools."
'evovsdApplyScript -- Apply script for aTool.
'evovsdDocGUIs -- Document GUIs in the project.
'evovsdToolCut -- Tool to close the opening view.
'evovsdToolId -- ArcView's identity tool.
'evovsdTools -- Tools available on the View tool bar.
'evovsdViewWin -- Window that contains the view
'         name "Opening View."
'*****************************************

'*****************************************
'
' Find the opening-view window. If it is open, close
' it.
'
'*****************************************

evovsdViewWin = av.GetProject.FindDoc
            ("Opening View").GetWin

if (evovsdViewWin.IsOpen) then
  evovsdViewWin.Close
end


'*****************************************
'
' Remove the tool used to close the opening view.
'
'*****************************************

'Get a list of the GUIs in this project and save
'the View GUI.


evovsdDocGUIs = av.GetProject.GetGUIs
for each aDocGUI in evovsdDocGUIs
  if (aDocGUI.asString = "View") then


'Get a list of tools and find the tool that closes
'the opening view.


    evovsdTools = aDocGUI.GetToolBar.GetControls
    for each aTool in evovsdTools
      if (aTool.Is(Space)) then
        continue
      end
      evovsdApplyScript = aTool.GetApply
      if (evovsdApplyScript.contains
            ("EV.OpeningViewShut")) then
          evovsdToolCut = aTool
      end
      if (evovsdApplyScript.contains
        ("View.Identify")) then
          evovsdToolId = aTool
      end
    end


'Disable the tool that closes the opening view and
'activate the identity tool.


    evovsdToolCut.SetEnabled(FALSE)
    evovsdToolId.Select
    aDocGUI.Activate
  end


'Done with the view and other GUIs are not important.


  break
end
```

EV.OpeningViewShutDownUpdate

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:40:32 2000

'This script checks to see if "Opening View" is
'open. If "Opening View" is open, the script enables
'the tool needed to close that view. If the view is
'not open, the script disables the tool.

'*****************************************
'
'aDocGUI -- Loop variable to examine all GUIs in
'         "evovsdDocGUI."
'aTool -- Loop variable to examine all tools in
'         "evovsdTools."
'evovsdApplyScript -- Apply script for "aTool."
'evovsdDocGUIs -- Document GUIs in the project.
'evovsdGUI -- The opening view GUI.
'evovsdToolCut -- Tool to close the opening view.
'evovsdToolID -- ArcView's identity tool.
'evovsdTools -- Tools available on the view tool bar.
'evovsdViewWin -- Window that contains the view
'         name "Opening View."
'
'*****************************************

'*****************************************
'
' Get a list of the GUIs in this project and save
' the View GUI.
'
'*****************************************

evovsdDocGUIs = av.GetProject.GetGUIs
for each aDocGUI in evovsdDocGUIs
  if (aDocGUI.asString = "View") then
    evovsdGUI = aDocGUI
    break
  end
end

'*****************************************
'
' Get a list of tools on the view GUI and find the
' tool used to close the "Opening View."
'
'*****************************************

evovsdTools = evovsdGUI.GetToolBar.GetControls
evovsdToolCut = NIL
for each aTool in evovsdTools
  if (aTool.Is(Space)) then
    continue
  end
  evovsdApplyScript = aTool.GetApply
  if (evovsdApplyScript.contains
     ("EV.OpeningViewShut")) then
    evovsdToolCut = aTool
    break
  end
end


'*****************************************
'
' Find the "Opening View" window. Make the tool
' active if the tool exists and the view is
' open. Make the tool inactive if the view is
' not open. If the view is open and the tool
' does not exist, close the view.
'
'*****************************************

evovsdViewWin = av.GetProject.FindDoc
    ("Opening View").GetWin


'Look at the tool.


if (evovsdViewWin.IsOpen) then
  if (evovsdToolCut = NIL) then
    evovsdViewWin.Close
  else
    evovsdToolCut.SetEnabled(TRUE)
  end
else
  evovsdToolCut.SetEnabled(FALSE)
end
```

EV.RedrawView

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:40:40 2000

'*****************************************
'
'evrdvView -- View to be refreshed, passed from
'          the calling script.
'evrdvViewDis -- Display of the view.
'
'*****************************************

'Get the view and its display. Redraw the display.


evrdvView = Self
evrdvViewDis = evrdvView.GetDisplay
evrdvView.Draw(evrdvViewDis)
return NIL
```

EV.RemoveRankField

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:40:20 2000

'This script removes the Rank column from the
'data tables after the themes have been converted
'to a grid.

'*****************************************
'
'evrmTheme -- Theme passed from the calling script.
'evrmFTab -- FTab of "evrmTheme."
'evrmFields -- Fields in "evrmFTab."
'evrmField -- The "Rank_Value" field.
'evrmStatus -- Status of the change.
'
'*****************************************

evrmTheme = SELF
evrmFTab = evrmTheme.GetFTab
evrmFields = evrmFTab.GetFields
evrmField = evrmFTab.FindField("Rank_Value")
if (evrmField = NIL) then
  return NIL
else
  evrmFTab.StartEditingWithRecovery
  evrmFieldList = { evrmField }
  evrmFTab.SetEditable(TRUE)
  evrmFTab.RemoveFields(evrmFieldList)
  evrmStatus =
    evrmFTab.StopEditingWithRecovery
    (TRUE)
  evrmFTab.SetEditable(FALSE)
  if (NOT (evrmStatus)) then
    MsgBox.Error("Could not delete the Rank " +
                 "Value column",
                 "Removing Rank Field")
  end
end
```

EV.Startup

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:45:31 2000

'This is the start-up script for the IDEM and
'USGS ArcView Watershed Screening Tool.

'*****************************************
'
'aDocGUI -- Loop variable used to loop through
'       "evsuDocGUIs."
'EV_DATA -- System environment variable holding the
'       full pathname to the data base that
'       accompanies the application.
'ev_data -- Variable holding the full pathname to
'       the data base that accompanies the
'       application.
'EV_HOME -- System environment variable holding the
'       full pathname to the user area where the
'       files are saved.
'ev_home -- Variable holding the full pathname to
'       the user area where the files are saved.
'EV_TEMP -- System environment variable holding the
'       full pathname to the temporary work
'       area.
'ev_temp -- Variable holding the full pathname to
'       the temporary work area.
'evevenvDic -- Dictionary containing the system
'       environment variables, returned from
'       "EV.GetEnvVar."
'evsuDocGUIs -- GUIs in this project.
'evsuGUI -- The view GUI.
'evsuProject -- Project name.
'evsuTool -- Tool for closing the "Opening View."
'evsuTools -- Tools on the view GUI.
'evsuView -- "Opening View" document.
'evsuViewWin -- Window showing the opening view.
'_timesin -- Used to count the number of times
'       the "AddTheme" button is executed the
'       first time the data base accompaning this
'       application is used.
'
'*****************************************

'*****************************************
'
' Get the system environment variables, set the
' current working directory and the project
' working directory, and create "_timesin."
'
'*****************************************
evevenvDic = av.Run("EV.GetEnvVar","")
ev_home = evevenvDic.Get("EV_HOME")
ev_data = evevenvDic.Get("EV_DATA")
ev_temp = evevenvDic.Get("EV_TEMP")
ev_home.asFileName.SetCWD
av.GetProject.SetWorkDir(ev_home.asFileName)
_timesin = 0

'*****************************************
'
' If "Opening View" is not open, open it.
'
'*****************************************

evsuProject = av.GetProject
evsuView = evsuProject.FindDoc ("Opening View")
evsuViewWin = evsuView.GetWin
if (evsuViewWin.IsOpen.Not) then
  evsuViewWin.Open
end

'*****************************************
'
' Get the tools on the View tool bar and activate
' the tool for closing "Opening View."
'
'*****************************************

'Get the GUIs in the project and save the view GUI.


evsuDocGUIs = av.GetProject.GetGUIs
for each aDocGUI in evsuDocGUIs
  if (aDocGUI.asString = "View") then
    evsuGUI = aDocGUI
  end
end


'Get the tools from the View tool bar, and find
'the tool that closes "Opening View."


evsuTools = evsuGUI.GetToolBar.GetControls
for each aTool in evsuTools
  if (aTool.Is(Space)) then
    continue
  end
  if (aTool.GetApply.contains
     ("EV.OpeningViewShutDown")) then
    evsuTool = aTool
    break
  end
```

```
end

'Enable and select the tool for closing "Opening
'View."
                                        evsuTool.SetEnabled(TRUE)
                                        evsuTool.Select
```

EV.ViewGet

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:45:40 2000

'Script to find the view with which the user wishes
'to work. Assumes if only one view is active, that
'is the view with which the user wishes to work.

‘*****************************************
'
'actViews -- List of active views in the project.
'aDoc -- A loop variable used to look at each
'        document in "evgvDocList."
'evgvcalling -- Program calling this script.
'evgvDocList -- List of documents in this project.
'evgvViewDictionary -- Dictionary of views and
'        whether the views are open, closed, or
'        active.
'theViews -- List of all views in the project.
'
‘*****************************************

‘*****************************************
'
' Get the information passed to this script and a list
' of documents associated with the project, initiate
' a dictionary to hold the names of all views and
' whether the views are open, active, or just there.
'
‘*****************************************

evgvcalling = SELF
evgvDocList = av.GetProject.GetDocs
evgvViewDictionary = Dictionary.Make
                     (evgvDocList.Count)


‘*****************************************
'
' Look at each document, if it is a view and is not
' the "Opening View" then look at that view.
' Is it open, active, or what?  If it is not a view,
' get the next document. Then create a list of
' views and a list of active views.
'
‘*****************************************

for each aDoc in evgvDocList


'Is this document a view and is it named "Opening
View"?
  if (aDoc.GetGUI = "View") then
    if (aDoc.asString = "Opening View") then
      continue
    else


'If the view is active, add it to the dictionary
'as active.


      if (aDoc.IsActive) then
        evgvViewDictionary.Add(aDoc,"active")


'If the view is open, add it to the dictionary
'as open.


      elseif (aDoc.GetWin.IsOpen) then
        evgvViewDictionary.Add(aDoc,"visible")

'If the view is not active or open, just add as
'closed.


      else
        evgvViewDictionary.Add(aDoc,"closed")
      end
    end
  end
end

'Create a list of views.


theViews = evgvViewDictionary.ReturnKeys


'Create a list of active views.


actViews = {}
for each aView in theViews
  if (evgvViewDictionary.Get(aView) = "active")
      then
    actViews.Add(aView)
  end
end
```

```
'****************************************

'
' Open a view.
' If there is only one active view, return that view.
' If more than one view is active, show the user the
' list and ask them to select one view with which to
' work.
' If no views are active, show the user a list of
' all views and ask them to select one view.
' If there are no views, then return NIL.
'
'****************************************

'One active view, use that view.


if (actViews.Count = 1) then
  evgvView = actViews.Get(0)


'If more than one active view, ask the user to select
'which view to use.


elseif (actViews.Count > 1) then
  evgvView = MsgBox.ListasString(actViews,
          "Select a view with which to work.",
          "Finding a view with which to work.")


'No active views, show the list of all views and
'ask the user to select one.

elseif (theViews.Count > 0) then
  evgvView = MsgBox.ListasString(theViews,
          "Select a view with which to work.",
          "Finding a view with which to work.")


'No View error.


else
  MsgBox.Error("There are no views in this " +
           "project. Please create a view " +
           "before continuing.",
           "Getting view error.")
  evgvView = NIL
end


'Open and return the view.


if (NOT (evgvView = NIL)) then
  if (NOT (evgvView.GetWin.IsOpen)) then
    evgvView.GetWin.Open
    evgvView.GetWin.Activate
  end
end

if (evgvView = NIL) then
  exit
end
return evgvView
```

EV.WeightFactors

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:45:48 2000

'Script for assigning weight factors to themes.
'Each group of themes and weight factors can
'be saved as a scenario. A group of themes is
'defined to be the active themes in the current
'view.

'*****************************************
'
'actThemes -- Themes included in this scenario.
'aDesc -- Description for this scenario.
'aReturnDic -- Returned from the script that
'       allows the user to select a scenario.
'       This dictionary contains file name,
'       themes and weight factors, and
'       other information for the scenario
'       selected by the user.
'aScenario -- The file that contains the lists of
'       files containing the themes and weight
'       factors that make up a single scenario.
'aTheme -- Loop variable used to loop through
'       each theme.
'aThemeWFDic -- Dictionary containing the themes
'       and weight factors.
'aVTab -- Table containing the theme and weight
'       factors.
'defdesc -- Default description for the scenario
'       file. Blank or read from the file
'       containing the list of scenario files.
'defnm -- Default name for the file holding the
'       themes and weight factors.
'ev_cwd -- Current working directory.
'EV_DATA -- System environment variable holding the
'       full pathname to the data base that
'       accompanies the application.
'ev_data -- Variable holding the full pathname to
'       the data base that accompanies the
'       application.
'EV_HOME -- System environment variable holding the
'       full pathname to the user area where
'       files are saved.
'ev_home -- Variable holding the full pathname to
'       the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'       full pathname to the temporary work
'       area.
'ev_temp -- Variable holding the full pathname to
'       the temporary work area.
'evevenvDic -- Dictionary containing the system
'       environment variables, returned from
'       "EV.GetEnvVar."
'ExpFld -- Field in the table that contains the
'       explanation of the file holding the
'       themes and weight factors.
'FileFld -- Field in the table that contains
'       the name of the file holding the
'       themes and weight factors.
'found -- False, the file containing the themes and
'       weight factors was found. True, the
'       file was not found.
'RecNum -- Record number in a table where
'       information is to be written.
'rtnVal -- True, if the theme passed are polygons.
'       Project works only on polygons.
'theFileNm -- File name of the scenario file. Used
'       to save or reuse scenarios.
'themeFld -- Field in the table that contains the
'       theme where "WFField" is the weight
'       factor.
'theThemes -- Active themes in the view. This
'       script assumes the user wants to
'       assign a weight factor to each
'       active theme and the active theme
'       make up a scenario.
'theView -- View the user wishes to use.
'tmpAns -- Temporary variable for responding to a
'       request.
'tmpAns1 -- Temporary variable for responding to a
'       request.
'tmpAns2 -- Temporary variable for responding to a
'       request.
'tmpFileNmDef -- Temporary file name definition
'       used to save a new scenario.
'tmpmsg -- Message printed when saving the
'       scenario. Content depends on the
'       value of "wftype."
'wfdic -- Dictionary containing new weight factors.
'WFField -- Field in the table that contains the
'       weight factor where "themeFld" is the
'       theme.
'wftype -- Old or new, tells the application if
'       the scenario is a new or previously
'       used scenario.
'
'*****************************************

'*****************************************
'
' Get environment variables, active themes, and
' other information needed.
'
'*****************************************
```

```
'Get the system environment variables

evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("EV_TEMP")
ev_cwd = FileName.GetCWD
ev_pwd = av.GetProject.GetWorkDir
evSceneDir = ev_data + "\Scenes"
evScenarioDir = evSceneDir + "\Scenarios"


'Get the view.


theView = av.Run("EV.ViewGet","")


'Get the active themes.


theThemes = theView.GetActiveThemes

if (theThemes.Count = 0) then
  MsgBox.Warning("You must have themes in the " +
              "view to assign weight factors.",
              "Assigning weight factors.")
  return NIL
end

'Set the variable "theFileNm" to NIL. This variable
'may be populated by a scenario. The variable will be
'checked later. The script's actions will depend on
'whether the value is NIL or not.


theFileNm = NIL


'*****************************************
'
' Determine if the user wants to use an existing
' scenario or use the active themes. If
' an existing scenario is selected get the themes
' and weight factors. If a new scenario is created
' assign default of 1 as the weight factor.
'
'*****************************************

'Ask the user whether to use the active themes
'or get an existing scenario.


tmpAns0 = NIL
if (theThemes.Count >= 1) then
  tmpAns0 = MsgBox.ListasString(theThemes,"OK - " +
                "Use these themes to " +
                "create a new scenario." +
                nl + "Cancel - Use " +
                "existing scenarios or " +
                "halt execution.",
                "Select themes to include " +
                "in the scenario.")

end


'The user clicked on cancel. Check to see if the
'user wants to cancel this operation or use an
'existing scenario file.


if (tmpAns0 = NIL) then


'Show the user a list of files that contains the
'scenario files and descriptions. If the user
'keys cancel, end this script.


  aScenario = NIL
  av.ShowMsg("Retrieving file containing " +
          "scenarios ...")
  aScenario = av.Run("EV.WFGetScenarios","Select")
  av.ShowMsg(" ")


'The user keyed "cancel," kill the script.


  if (aScenario = NIL) then
    return NIL
  end


'Ask the user for the name of the scenario file
'that contains the themes and weight factors to
'use. Verify that the user wishes to quit if
'cancel is selected.


  tmpAns1 = TRUE
  while (tmpAns1)
    asendDic = Dictionary.Make(2)
    asendDic.Set("Scenario",aScenario)
```

```
  asendDic.Set("View",theView)
  aReturnDic = av.Run("EV.WFGetTheme",asendDic)


'The user did not select a scenario file. End
'this script.

  if (NOT(aReturnDic = NIL) AND
    (aReturnDic.Count > 0)) then
   tmpAns1 = FALSE
  else
   tmpAns2 = MsgBox.YesNo("It appears you " +
          "clicked on CANCEL or the file " +
          "you selected contains no " +
          "themes. Do you want to stop " +
          "the assignment of weight " +
          "factors?",
          "Selecting a scenario, " +
          "Assigning Weight Factors",TRUE)
    if (tmpAns2) then
     return NIL
    end
  end
 end


'The user selected a file name. Unload the
'file name of the dictionary containing the themes
'and weight factors. Set the list of active
'themes to the themes contained in the
'weight-factor dictionary. Set the weight-factor
'flag to old, i.e., it is an existing file.

 theFileNm = aReturnDic.Get("FileNm")
 defdesc = aReturnDic.Get("Desc")
 aThemeWFDic = aReturnDic.Get("WFDIC")
 actThemes = aThemeWFDic.ReturnKeys
 wftype = "OLD"

else

'The user has chosen to use the active theme.

 actThemes = theView.GetActiveThemes


'Check each theme to ensure the geometry is
'polygon.

 rtnVal = av.Run("EV.WFCheckThemes",actThemes)
 if (NOT(rtnVal)) then
  return NIL
 end


'Assign a default weight factor of "1" to each
'theme.

 aThemeWFDic = Dictionary.Make(actThemes.Count)
 for each aTheme in actThemes
  aThemeWFDic.Set(aTheme,1)
 end


'Ask the user to select a file in which to save the
'list of scenarios. Verify the answer if no file
'is selected.

 tmpAns1 = TRUE
 while (tmpAns1)
  aScenario = av.Run("EV.WFGetScenarios","Save")


'The user did not enter a file name.  End this
'script.

   if (NOT (aScenario = NIL)) then
    tmpAns1 = FALSE
   else
    tmpAns2 = MsgBox.YesNo ("You must " +
          "identify a scenario file to " +
          "maintain the list of themes " +
          "and weight factors. Are you " +
          "sure you DO NOT want to save " +
          "the themes and weight factors." +
          "You will lose the work you " +
          "just completed.",
          "Assigning Weight Factors",
          TRUE)
    if (tmpAns2) then
     return NIL
    end
   end
  end
  wftype = "NEW"
end

'*****************************************
'
' Allow the weight factor to be modified, and
' save the themes and weight factors.
'
```

```
'*****************************************


'Send the theme and default weight factor
'information to a script. The script shows the
'information to the user and allows the weight
'factors to be modified.

wfdic = av.Run("EV.WFAssignFactor",aThemeWFDic)

'No data were returned from the script.
'End this script.

if (wfdic.Count = 0) then
  MsgBox.Error("No weight factors were " +
    "assigned. Ending the " +
    "assignment of weight factors.",
    "Assigning weight factors")
  return NIL
end

'Save the weight factors to a file.
'Ask the user for the name of the file to hold
'the scenarios.

if (wftype = "NEW") then
  tmpmsg = "Enter the file name for saving the " +
         "themes and weight factors."
else
  tmpmsg = "Enter the file name for saving the " +
         "themes and weight factors. Cancel " +
         "to use the same name."
end

'If this is a new scenario, ensure that a file name
'has been given.

tmpAns0 = TRUE
tmpAns1 = TRUE
if (NOT (theFileNm = NIL)) then
  tmpFileNmDef = theFileNm.GetBaseName
else
  tmpFileNmDef = ""
end
while (tmpAns0)

'Ask for a file name and make sure it is valid.

  tmpFileNm = NIL
  aFileNm = NIL
  while (tmpAns1)
    tmpFileNm = MsgBox.Input(tmpmsg,
                  "Enter file name.",
                  tmpFileNmDef)
    if (NOT (tmpFileNm = NIL)) then
      aFileNm = tmpFileNm.Trim
      if (NOT (aFileNm.Contains(" "))) then
        tmpAns1 = FALSE
      else
        MsgBox.Error("The full pathname " +
                evScenarioDir + "\" + aFileNm +
                " contain spaces. Use " +
                "underscores in place " +
                "of spaces.","Bad File Name")
      end
      if ((aFileNm.Count + evScenarioDir.Count)
          > 150) then
        MsgBox.Error("The full pathname " +
                evScenarioDir + "\" + aFileNm +
                " exceeds 150 characters. " +
                "Shorten the file name.",
                "Bad File Name")
      end
    else
      if (wftype = "OLD") then
        tmpAns1 = FALSE
      else
        exit
      end
    end
    tmpFileNmDef = aFileNm.asString
  end
  if (wftype = "NEW") then
    if (NOT (aFileNm = NIL)) then
      tmpAns0 = FALSE
    end
  else
    tmpAns0 = FALSE
  end
end

'Check the file system. If the file exists, create
'a default name and let the user decide which
'file to use. The scenario files are stored in the
'home directory. Change the CWD to the home
```

```
'directory.


if (aFileNm <> NIL) then
  ev_cwd = FileName.GetCWD
  ev_pwd = av.GetProject.GetWorkDir
  evScenarioDir.asFileName.SetCWD
  av.GetProject.SetWorkDir
     (evScenarioDir.asFileName)
  if (NOT (aFileNm.Contains(".dbf"))) then



    aFileNm = aFileNm + ".dbf"
  end
  theFileNm = aFileNm.asFileName
  aBaseNm = theFileNm.GetBaseName

  if (File.Exists(theFileNm)) then
    tmpAns = MsgBox.YesNo("The file " + aBaseNm +
             " exists. Do you wish to " +
             "overwrite the file?",
             "Saving weight factors.",FALSE)
    if (NOT (tmpAns)) then
      defnm = av.GetProject.MakeFileName
                    ("weight_f","dbf")
      theFileNm = FileDialog.Put(defnm,"*.*",
             theFileNm.asString + " exists. " +
             "Save themes and weight " +
             "factors to:")
      if (theFileNm = NIL) then
        theFileNm = aFileNm.asFileName
      end
    end
  else
    theFileNm = aFileNm.asFileName
    theOS = System.GetOS
    if (theOS = "#SYSTEM_OS_UNIX") then
      theFileNm.SetSaveAsUNIX
    end
  end


'Ask the user for a description of the scenarios.
'Make sure the scenario description is 150
'characters or less.

  defdesc = ""
  tmpAns = TRUE
  while (tmpAns)
    aDesc = MsgBox.Input("Enter a short " +
            "description of this scenario, " +
            "maximum length is " +
            "150 characters. If a scenario " +
            "file was used, that description " +
            "is shown","Enter scenario",
            defdesc)
    if (aDesc = NIL) then
      tmpAns1 = MsgBox.YesNo("You just keyed " +
                "CANCEL. Do you wish to " +
                "cancel the weight " +
                "factor entry? No will " +
                "cancel just the " +
                "description entry.",
                "Cancel weight factor entry?",
                FALSE)
      aDesc = defdesc
    end



    aDescLen = aDesc.Count
    if (aDescLen <= 150) then
      tmpAns = FALSE
    else
      MsgBox.Error("Your description of the " +
              "scenario must be 150 " +
              "characters or less.",
              "Enter scenario description.")
    end
  end


'Save the file name and description to the file
'of scenarios. Making the changes to the table
'makes the changes to the disk file.


  FileFld = aScenario.FindField("File_name")
  ExpFld = aScenario.FindField("Scenario")
  aScenario.SetEditable(TRUE)


'If the file name already exists in the scenario
'file replace the description entry.


  found = TRUE
  if (aScenario.GetNumRecords > 0) then
    for each rec in 0 ..
                (aScenario.GetNumRecords - 1)
      existfilenm = aScenario.ReturnValue
                      (FileFld,rec)
      if (existfilenm = theFileNm.GetBaseName) then
        aScenario.SetValue(ExpFld,rec,aDesc)
        found = FALSE
        break
      end
    end
  end
```

```
'The file name was not found in the scenario
'file, add this is a new scenario.

  if (found) then

'Add a record to the table of scenarios.

    RecNum = aScenario.AddRecord

'Add the entry

    aScenario.SetValue(FileFld,RecNum,
          theFileNm.GetBaseName)
    aScenario.SetValue(ExpFld,RecNum,aDesc)
    aScenario.SetEditable(FALSE)
  end

'Make a virtual table to contain the themes and
'weight factors.

  aVTab = VTab.MakeNew(theFileNm,dBASE)
  themeFld = Field.Make("Theme_name",
                #FIELD_VCHAR,60,0)
  WFField =
Field.Make("W_Factor",#FIELD_FLOAT,9,3)
  aVTab.AddFields( {themeFld, WFField})

'Save the theme names and weight factors to the
'virtual table.

  aVTab.SetEditable(TRUE)
  for each aTheme in wfdic.ReturnKeys

'Add a record to the virtual table.

    RecNum = aVTab.AddRecord
    aVTab.SetValue(themeFld,RecNum,aTheme.asString)
    aVTab.SetValue(WFField,RecNum,wfdic.
    Get(aTheme))
  end
  aVTab.SetEditable(FALSE)
else

'Modify the file with new values.

  aVTab = VTab.Make(theFileNm,TRUE,FALSE)
  themeFld = aVTab.FindField("Theme_name")
  WFField = aVTab.FindField("W_Factor")
  aVTab.SetEditable(TRUE)
  for each aTheme in wfdic.ReturnKeys
    for each tmpJ in 0 .. (aVTab.GetNumRecords - 1)
      filTheme = aVTab.ReturnValue(themeFld,tmpJ)
      if (filTheme = aTheme) then
        aVTab.SetValue(WFField,tmpJ,
                wfdic.Get(aTheme))
      end
    end
  end
  aVTab.SetEditable(FALSE)
end

'Reset the current working directory.

ev_cwd.SetCWD
av.GetProject.SetWorkDir(ev_pwd)
av.ShowMsg("Have assigned weight factors.")
```

EV.WFAssignFactor

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:45:56 2000

'Script to assign weight factors to a list of themes
'Existing weight factors or defaults of 1
'are placed in a dictionary along with the themes.

'*****************************************
'
'aTheme -- Loop variable used to loop through all
'         themes passed to this script.
'recDic -- Dictionary passed from the calling script
'         that contains the themes and default
'         weight factors.
'themeDesc -- Variable used to display the theme
'         names to the user.
'tmpI -- Temporary counter.
'wfactor -- Weight factors entered by the user.
'wfVal -- Variable used to display the weight
'         factors to the user.
'
'*****************************************

'*****************************************
'
' Get the information passed from the calling script
' and initialize variables.
'
'*****************************************

recDic = SELF
themeDesc = " "
wfVal = " "
tmpI = 0

'*****************************************
'
' Show the user the themes and weight factors,
' and allow the user to change the weight factors.
'
'*****************************************


'Build the variables needed to show the user
'a list of themes and default weight factors.
'OK, so you looked and you wondered what was this
'fool thinking? Chalk it up to some of my first
'Avenue code and forgive me.


firsttheme = NIL
for each aTheme in recDic.ReturnKeys
  if (firsttheme = NIL) then
    firsttheme = aTheme
  end
  themeDesc = themeDesc + " " + aTheme.asString
  wfVal = wfVal + " " + recDic.
                    Get(aTheme).asString
end


'Show the user the themes and default weight
'factors. Allow the user to change the weight
'factors.


passDic = Dictionary.Make(2)
passDic.Set("Themes",themeDesc)
passDic.Set("Weight",wfVal)
wfactor = av.Run("EV.GetWeightFactors",passDic)


'*****************************************
'
' Return the weight factors to the calling script.
'
'*****************************************

'The user keyed "cancel," exit the script.


if (wfactor.Count = 0) then
  exit
end


' Return the revised weight factors.

tmpAns = TRUE
while (tmpAns)
  tmpI = 0
  for each tmpI in 0 .. (themeDesc.asList.Count - 1)
    for each aTheme in recDic.ReturnKeys
      if (themeDesc.asList.Get(tmpI) =
          aTheme.asString) then
        if (wfactor.Get(tmpI).isNumber) then
          recDic.Set(aTheme,wfactor.
                     Get(tmpI).asNumber)
        else
          MsgBox.Error("Weight factors must " +
                       "be numeric. Theme " +
                       aTheme.asString + " was " +
                       "assigned a weight " +
                       "factor of " +
```

```
            wfactor.Get(tmpI),

"Found nonnumeric " +
            "weight factor.")
  wfactor = av.Run("Ev.GetWeightFactors",
              passDic)
  break
end
```

```
    end
   end
  end
  tmpAns = FALSE
end

return recDic
```

EV.WFCheckThemes

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:03 2000

'This script is used to pass a list of themes. The
'script then checks each theme to ensure the
'geometry is polygon. The merge theme works on only
'polygons, not lines and points. If all themes passed
'are polygon, the script returns TRUE, else it returns
'FALSE.

‘*****************************************
'
'actThemes -- List of themes passed to this script.
'aField -- The shape field.
'aFTab -- Theme's FTab.
'aShape -- The shape value.
'aTheme -- Loop variable used to loop through the
'         list of themes.
'theName -- The theme name.
'theShapeType -- Type of shape.
'tmpI -- Temporary counter.
'tmpJ -- Number of themes passed.
'tmpK -- Temporary counter.
'
‘*****************************************

‘*****************************************
'
' Loop through each passed theme. If the shape
' type is not polygon, print a message and stop
' this script.
'
‘*****************************************

'Get the themes passed and clear the status bar.
actThemes = SELF
av.ClearMsg
av.ShowMsg("Checking to ensure the themes " +
        "selected are valid.")


'Loop through each record and look at the shape type

tmpI = 1
tmpJ = actThemes.Count
for each aTheme in actThemes
  aFTab = aTheme.GetFTab
  aField = aFTab.FindField("Shape")
  for each tmpI in 0 .. (aFTab.GetNumRecords - 1)
    aShape = aFTab.ReturnValue(aField,tmpI)
    theShapeType = aShape.GetDimension
    if (theShapeType < 2) then
      theName = aTheme.GetName
      MsgBox.Error ("The theme " + theName +
          "contains shapes other than " +
          "polygons. This application can " +
          "process only polygons.",
          "Assigning weight factors")
      return FALSE
    end
  end


'Update the staus bar.


  tmpK = (tmpI / tmpJ) * 100
  av.SetStatus(tmpK)
  tmpI = tmpI + 1
end
av.ClearMsg
return TRUE
```

EV.WFGetScenarios

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:12 2000

'This script allows the user to select a file
'containing a list of scenarios. A scenario
'consists of the themes used, their weight factors,
'and the name of the resulting theme.

‘*****************************************
'
'aVTab -- Table of scenario files and descriptions.
'ev_cwd -- Current working directory.
'EV_DATA -- System environment variable holding the
'         full pathname to the data base that
'         accompanies the application.
'ev_data -- Variable holding the full pathname to
'         the data base that accompanies the
'         application.
'EV_HOME -- System environment variable holding the
'         full pathname to the user area where
'         files are saved.
'ev_home -- Variable holding the full pathname to
'         the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'         full pathname to the temporary work
'         area.
'ev_temp -- Variable holding the full pathname to
'         the temporary work area.
'evevenvDic -- Dictionary containing the system
'         environment variables, returned from
'         "EV.GetEnvVar."
'fdmsg -- Message asking the user to select a file.
'fileNm -- Name of the file the user selected.
'tmpAns -- Used to check user responses.
'usetype -- Used to select message, select file
'         to save, or select scenario file.
'
‘*****************************************

‘*****************************************
'
' Get the information passed from the calling
' script and the system environment variables.
'
‘*****************************************

usetype = SELF


'Get the system environment variables.
evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("TEMP")

‘*****************************************
'
' Get the file containing the scenario files.
' Create a VTab of the scenario files and
' descriptions.
'
‘*****************************************

'The files containing the scenarios are in the
'scenes directory. Get the current working directory,
'then  change the current working directory to the
'home directory.


ev_cwd = FileName.GetCWD
ev_pwd = av.GetProject.GetWorkDir
evSceneDir = ev_data + "/Scenes"
evSceneDir.asFileName.SetCWD


'Look for the files containing the scenarios.
'These files contain a list of file names and
'descriptions. Each file contains the themes and
'weight factors for each theme. If a file
'containing scenarios exists, show the user the
'file(s).


if (File.Exists("Scenes1.dbf".asFileName)) then
  av.ShowMsg("Found a file that contains " +
          "scenarios.")
  if (usetype = "Save") then
    fdmsg = "Select the file to contain the " +
          "scenario file. Cancel to " +
          "create a new file."
  end
  if (usetype = "Select") then
    fdmsg = "Select the file from which " +
          "the scenario file and description " +
          "can be selected."
  end
  fileNm = FileDialog.Show("Scene*","Scenes",fdmsg)
  av.ShowMsg("")
  if ((fileNm = NIL) and (usetype = "Save")) then
    tmpAns = MsgBox.YesNo("Do you wish to " +
                  "create a new file " +
                  "for maintaining the " +
                  "scenario files?",
```

```
                    "New File?",TRUE)

    if (tmpAns) then
      fileNm = "new"
    else
      exit
    end
  end
else
  fileNm = "new"
end


'A new operation, create the first scene file.


if (fileNm = "new") then
  aVTab = av.Run("EV.WFMakeScenario","")
else


'No file name. If this is the selected operation, then
'stop this script.

  if (fileNm = NIL) then
    if (usetype = "Select") then
      return NIL
    end


'No file name and this is a save operation.
'Create the file to maintain the scenario
'file names and descriptions. Write the column
'headings to the file.
    tmpAns = MsgBox.YesNo("Do you wish to create " +
             "a new file for maintaining the " +
             "list of scenarios?",
             "New File?",FALSE)
    if (tmpAns) then
      aVTab = av.Run("EV.WFMakeScenario","")
    else
      return NIL
    end
  else


'The scenario file exists, use it.


    if
(TextFile.Make(fileNm,#File_Perm_READ).GetSize
         = 0) then
      MsgBox.Error("There are no records in the " +
                "file " + fileNm.asString,
                "Get Scenario")
      ev_cwd.SetCWD
      exit
    end
    aVTab = VTAB.Make(fileNm,TRUE,FALSE)
  end
end

'Return the CWD to its original value.


ev_cwd.SetCWD
av.GetProject.SetWorkDir(ev_pwd)
return aVTab
```

EV.WFGetTheme

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:25 2000

'This script creates the list of themes that the user
'uses to create a scenario.

‘*****************************************
'
'aFile -- Name of the file in the record number
'        "recNum."
'aFileNm -- Name of the file containing the themes
'        and weight factors, the scenario file.
'aScenario -- Name of the passed file. This file
'        contains the list of files that contain
'        the themes and weight factors.
'aVTab -- VTab of themes and weight factors.
'ev_cwd -- Current working directory.
'EV_DATA -- System environment variable holding the
'        full pathname to the data base that
'        accompanies the application.
'ev_data -- Variable holding the full pathname to
'        the data base that accompanies the
'        application.
'EV_HOME -- System environment variable holding the
'        full pathname to the user area where
'        files are saved.
'ev_home -- Variable holding the full pathname to
'        the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'        full pathname to the temporary work
'        area.
'ev_temp -- Variable holding the full pathname to
'        the temporary work area.
'evevenvDic -- Dictionary containing the system
'        environment variables, returned from
'        "EV.GetEnvVar."
'expFld -- Field containing the explanation of the
'        scenario file.
'fileFld -- Field containing the name of the
'        scenario file.
'fileNmLen -- Length of the file name containing
'        the themes and weight factors.
'maxLength -- Length of the longest file name in
'       "aScenario."
'recNum -- Record number, used to loop through
'        the records in a table.
'thedesc -- List of descriptions contained in
'       "aScenario."
'theFilDesc -- Used to show the files and file
'       descriptions to the user.
'thefiles -- List of files contained in "aScenario."
'theFileNm -- The scenario file.
'thelength -- Length of the file name contained
'        in record "recNum."
'ThemeFld -- Field containing the theme.
'theRetDic -- Variable that contains "theWFDic" and
'        other information passed back to the calling
'        script.
'theWFDic -- Dictionary used to pass themes and
'        weight factors between scripts.
'tmpI -- Temporary counter.
'WFField -- Field containing the weight factor.
'
‘*****************************************

‘*****************************************
'
' Get the information passed from the calling script,
' the system environment variables, and the current
' working directory.
'
‘*****************************************


'Get the name of the file containing the list
'of scenario files.


passDic = SELF
aScenario = passDic.Get("Scenario")
theView = passDic.Get("View")


'Get the system environment variables.


evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("TEMP")


'The files are stored in the user's home directory.
'Change the current working directory to the user's
'home directory. Maintain the CWD so that it can be
'changed back before exiting the script.


ev_cwd = FileName.GetCWD
ev_pwd = av.GetProject.GetWorkDir
evScenarioDir = ev_data + "\Scenes\Scenarios"
evScenarioDir.asFileName.SetCWD
```

```
'*****************************************
'
' Create the list of scenario files and allow the
' user to select the desired scenario file.
'
'*****************************************


'Get the field names from the file containing the
'list of scenario files.

fileFld = aScenario.FindField("File_Name")
expFld = aScenario.FindField("Scenario")


'Create a list of the files and their descriptions.


thefiles = {}
thedesc = {}
if (aScenario.GetNumRecords < 1) then
  MsgBox.Error("There are no scenario files in " +
            aScenario.asString,"Error finding " +
            "scenario file.")
  exit
end
for each recNum in 0 ..
                (aScenario.GetNumRecords - 1)
  thefiles.Add(aScenario.ReturnValue
                 (fileFld,recNum).asString)
  thedesc.Add(aScenario.ReturnValue(expFld,recNum)
               .asString)
end


'Build the list to show the user.


theFilDesc = {}
theFilDesc.Add("The file name --> The file description")
theFilDesc.Add("  ")
for each recNum in 0 .. (thefiles.Count - 1)


'Build the entry.

  aFile = thefiles.Get(recNum)
  theFilDesc.Add(aFile + " --> " + thedesc.Get(recNum))
end


'Show the user the list of scenarios and let the
'user select the file to use. If the user selects one of
'the column headings or a blank column, do nothing.

tmpAns = TRUE


while (tmpAns)
  aFile = MsgBox.ListasString(theFilDesc,
              "Select the " +
              "desired scenario file.",
              "Select Scenario.")
  if (aFile = NIL) then
    exit
  end
  for each entNum in 0 .. (theFilDesc.Count - 1)
    if (aFile = theFilDesc.Get(entNum)) then
      if (entNum > 1) then
        tmpAns = FALSE
        break
      end
    end
  end
end


'User selected "cancel," end script.


if (aFile = NIL) then
  exit
end

'*****************************************
'
' Get the file containing the themes and
' weight factors and return the information to the
' calling script.
'
'*****************************************

'Get the name of the file containing the themes and
'weight factors.

for each tmpI in 0 .. (aFile.Count - 1)
  if (aFile.Middle(tmpI,1) = " ") then
    fileNmLen = tmpI + 1
    break
  end
end


'Open the file and create the VTab of themes and
'weight factors.


aFileNm = aFile.Left(fileNmLen)
defdesc = aFile.Right(aFile.Count - (fileNmLen +
4)).Trim
```

```
theFileNm = FileName.Make(aFileNm.Trim)
if (System.GetOS = "#SYSTEM_OS_UNIX") then
   theFileNm.SetSaveAsUNIX
end


aVTab = VTab.Make(theFileNm,FALSE,FALSE)


'Get the field names needed to read the VTab.


ThemeFld = aVTab.FindField("Theme_name")
WFField = aVTab.FindField("W_factor")


'Make and populate a dictionary where the key is
'the theme and the value is the weight factor.


theWFDic = Dictionary.Make(aVTab.GetNumRecords)
if (aVTab.GetNumRecords < 1) then
  MsgBox.Error ("There are no themes in the " +
             "file " + aFileNm,
             "Error opening the scenario file.")
  exit
end
for each recNum in 0 .. (aVTab.GetNumRecords - 1)
  aTheme = aVTab.ReturnValue(ThemeFld,recNum)
  inview = theView.FindTheme(aTheme)
  if (inview = Nil) then
    MsgBox.Error("The theme " + aTheme.asString +
              " is not in view " +
              theView.asString,"Getting Themes")
    errList = aVTab.ReturnValue(ThemeFld,0).asString
    if (aVTab.GetNumRecords > 1) then
      for each errNum in 1 ..
                    (aVTab.GetNumRecords -1)
        errList = errList + nl +
                aVTab.ReturnValue(ThemeFld,errNum)
      end
    end
    MsgBox.Report(errList,
             "Expecting the following themes.")
    exit
  else
    wf = aVTab.ReturnValue(WFField,recNum)
    theWFDic.Set(aTheme,wf)
  end
end


'Make another dictionary to return the file name,
'dictionary of themes and weight factors, and the
'VTab of themes and weight factors.


theRetDic = Dictionary.Make(4)
theRetDic.Set("FileNm",theFileNm)
theRetDic.Set("Desc",defdesc)
theRetDic.Set("WFDIC",theWFDic)
theRetDic.Set("VTAB",aVTab)


'Return the CWD to its original values.


ev_cwd.SetCWD
av.GetProject.SetWorkDir(ev_pwd)
return theRetDic
```

EV.WFMakeScenario

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:36 2000

'Script to make the file that holds the scenario
'files and their descriptions. The scenario files
'contain the list of themes and weight factors
'to use when screening a watershed.

'*****************************************
'
'aVTab -- VTab of file names and descriptions.
'ev_cwd -- Current working directory.
'EV_DATA -- System environment variable holding the
'       full pathname to the data base that
'       accompanies the application.
'ev_data -- Variable holding the full pathname to
'       the data base that accompanies the
'       application.
'EV_HOME -- System environment variable holding the
'       full pathname to the user area where
'       files are saved.
'ev_home -- Variable holding the full pathname to
'       the user area where files are saved.
'ev_pwd -- The project working directory.
'EV_TEMP -- System environment variable holding the
'       full pathname to the temporary work
'       area.
'ev_temp -- Variable holding the full pathname to
'       the temporary work area.
'evevenvDic -- Dictionary containing the system
'       environment variables, returned from
'       "EV.GetEnvVar."
'fld1 -- Field containing the the name of the
'       scenario file.
'fld2 -- Field containing the explanation of the
'       scenario file.
'fldLst -- List of fields to add to aVTab.
'fileNm -- Name of the file containing the list of
'       scenario files.
'
'*****************************************

'Get the system environment variables

evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("EV_TEMP")

'Get the current working directory. Change it to
'the home directory, if necessary. The file containing
'the list of scenarios is assumed to be in
'the home directory.

evSceneDir = ev_data + "\Scenes"
ev_pwd = av.GetProject.GetWorkDir
ev_cwd = FileName.GetCWD
if (ev_cwd.asString <> evSceneDir) then
  evSceneDir.asFileName.SetCWD
end
av.GetProject.SetWorkDir(evSceneDir.asFileName)


'Make the file name.


fileNm = av.GetProject.MakeFileName("scenes","dbf")
if (System.GetOS = "#SYSTEM_OS_UNIX")  then
  fileNm.SetSaveAsUNIX
end



'Create the file and VTab.


aVTab= VTab.MakeNew(fileNm,dBase)


'Add the fields to the table.


aVTab.SetEditable(TRUE)
fld1 =
Field.Make("File_Name",#FIELD_VCHAR,150,0)
fld2 = Field.Make("Scenario",#FIELD_VCHAR,150,0)
fldLst = { fld1, fld2 }
aVTab.AddFields(fldLst)
aVTab.SetEditable(FALSE)


'Return the VTab.
'Return the CWD and PWD to their original values.

av.GetProject.SetWorkDir(ev_pwd)
ev_cwd.SetCWD

return aVTab
```

EV.AOIApply

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:44 2000

'This is the "Apply" script for the Identify Watershed
'of Interest Tool. The logic of this script is
'from ESRI's View.Identify script.

'*****************************************
'
'aBitMap -- Bit map used to select records.
'aBitMapSuc -- Bit map resulting from a query.
'aGUI -- Variable used to loop through a list of GUIs.
'aVTab -- VTab of file names and descriptions.
'ev_cwd -- Current working directory.
'EV_DATA -- System environment variable holding the
'        full pathname to the data base that
'        accompanies the application.
'ev_data -- Variable holding the full pathname to
'        the data base that accompanies the
'        application.
'EV_HOME -- System environment variable holding the
'        full pathname to the user area where
'        files are saved.
'ev_home -- Variable holding the full pathname to
'        the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'        full pathname to the temporary work
'        area.
'evevenvDic -- Dictionary containing the system
'        environment variables, returned from
'        "EV.GetEnvVar."
'evwoiField -- The field used to identify the
'        watershed of interest.
'evwoiFileNm -- Name of the file containing the
'        watershed of interest.
'evwoiFound -- True, at least one theme contains
'        the point "evwoiPoint."
'evwoiFTab -- FTab for themes being examined.
'evwoiFTabFields -- Fields in the FTab "evwoiFTab."
'evwoiFTheme -- Theme created from the FTab that
'         resulted from the user selecting a
'         watershed of interest.
'evwoiGUI -- List of GUIs for this project. Used
'        to reset the tool bar.
'evwoiHuc -- Theme containing the next level of
'        hydrologic units to examine.
'evwoiHuc8c -- Eight-digit hydrologic unit code.
'evwoiKey -- First feature in the list "evwoiKeys"
'evwoiKeys -- List of features containing
'        "evwoiPoint" in the themes contained in
'        "evwoiThemes."
'evwoiLabel -- Hydrologic unit for the feature
'        containing the point "evwoiPoint."
'evwoiLabelField -- Field in "evwoiFTab" that contain
'        the primary hydrologic unit code
'        for the theme "evwoiTheme."
'evwoiNm -- Name of the hydrologic unit that
'        contains the point "evwoiPoint."
'evwoiNmField -- Field containing the name of the
'        hydrologic units.
'evwoiNumRec -- Number of records containing the
'        8-digit hydrolgic-unit code.
'evwoiPoint -- Location the user pointed to and
'        clicked on.
'evwoiRect -- Rectangle used to focus view after
'        a watershed of interest has been
'        selected.
'evwoiSrcNm -- Name of the shape file containing
'        the next hydrologic unit theme at which
'        to  look.
'evwoiQuery -- Query used to select records from
'        "evwoiTheme" VTab.
'evwoiTheme -- First theme in the list
'        "evwoiThemes" should be the watershed
'        of interest theme.
'evwoiThemes -- List of themes in "evwoiView."
'evwoiTmp -- Temporary variable used to identify
'        the primary hydrologic unit code for
'        theme "evwoiTheme."
'evwoiVal -- The value of the WOI field for the
'        "evwoiKey" feature.
'evwoiView -- View with which to work.
'evwoiWOIField -- Variable used to create the WOI
'        field in tables.
'recNum -- Record number found in a query.
'theField -- Field containing the themes' primary
'        hydrologic unit code.
'tmpAns -- Response to a message.
'tmpList -- Temporary list used to add columns to
'        tables.
'
'*****************************************

'*****************************************
'
' Get the system environment variables, the active
' view, the point entered by the user, and the
' watershed of interest theme. Initialize
' variables and make sure the features on the
' theme can be selected by pointing.
' WOICLICK allows the user to select the view,
' thus, the only active document should be the
' view with which the user wishes to work.
'
```

```
‘*****************************************

'Get the system environment variables


evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("EV_TEMP")
ev_cwd = FileName.GetCWD


'Get the view and location of the point where
'the mouse was clicked.


evwoiView = av.GetActiveDoc
evwoiPoint = evwoiView.GetDisplay.ReturnUserPoint

'Get the first theme in the list of themes.
'The GetThemes command returns a list of all themes
'in a View ordered as they are seen in the table of
'contents. EV.WOIClick moves the watershed of
'interest to the first position in this list.
'Thus, the first theme in the list is the watershed
'of interest the user has selected.


evwoiThemes = evwoiView.GetThemes
evwoiTheme = evwoiThemes.Get(0)

'Examine the "evwoiTheme," if it has been used before
'reset the WOI column to allow a new watershed of
'interest to be identified.

evwoiTheme.GetFTab.SetEditable(TRUE)
evwoitestfld = evwoiTheme.GetFTab.FindField("WOI")
if (evwoiTheme.GetFTab.FindField("Huc_14") = NIL)
then
  if (evwoiTheme.GetFTab.FindField("Huc_11") = NIL)
then
    repval = 8
  else
    repval = 11
  end
else
  repval = 14
end

if (Not (evwoitestfld = NIL)) then
  evwoiNumRec =
evwoiTheme.GetFTab.GetNumRecords
  for each aRecNum in 0 .. (evwoiNumRec - 1)
    avalue = evwoiTheme.GetFTab.ReturnValue
           (evwoitestfld,aRecNum)
    if (avalue = 99) then
      if (evwoiTheme.GetFTab.ReturnValue


         (evwoiTheme.GetFTab.FindField("Hu_name"),
          aRecNum).Contains("Part of")) then
           theval = repval * 2
      else
       theval = repval
      end
      evwoiTheme.GetFTab.SetValueNumber(
         evwoitestfld,aRecNum,theval)
    end
  end
end
evwoiTheme.getftab.seteditable(FALSE)



'If the features on this theme can be found by a
'point then get the feature associated with the
'point "evwoiPoint."


evwoiFound = FALSE
evwoiLabel = NIL
if (evwoiTheme.CanFindByPoint) then

‘*****************************************
'
' Get the information needed to identify the
' hydrologic unit the user identified and
' determine the hydrologic unit level of this
' theme.
'
‘*****************************************

   evwoiKeys = evwoiTheme.FindByPoint(evwoiPoint)

'There is one or more keys. Use the first key to
'get the label associated with the feature. In
'this case the label is the appropriate
'hydrologic unit code.

  if (evwoiKeys.Count > 0) then
    evwoiFound = TRUE
    evwoiKey = evwoiKeys.Get(0)

'Get the FTab associated with the theme. Use the
'fields in that table to determine which field
'should be used as the label field (The
'hydrologic unit code) for the proper level of
'hydrologic unit.

    evwoiFTab = evwoiTheme.GetFTab
```

```
  evwoiFTabFields = evwoiFTab.GetFields
  evwoiTmp = 0
  evwoiLabelField = NIL
  for each aField in evwoiFTabFields


    if ((aField.asString = "Huc_8") or
       (aField.asString = "Huc8")) then
      if (evwoiTmp < 8) then
        evwoiTmp = 8
        evwoiLabelField = aField
      end
    end
    if ((aField.asString = "Huc_11") or
       (aField.asString = "Huc11")) then
      if (evwoiTmp < 11) then
        evwoiTmp = 11
        evwoiLabelField = aField
      end
    end
    if ((aField.asString = "Huc_14") or
       (aField.asString = "Huc14")) then
      if (evwoiTmp < 14) then
        evwoiTmp = 14
        evwoiLabelField = aField
      end
    end
  end

‘*****************************************
'
' Set the label field and get the hydrologic unit
' code and name.
'
‘*****************************************

'Set the label field for the theme. Then get the
'value for that field (the hydrologic unit code
'for the watershed on which the user just clicks.


  evwoiTheme.SetLabelField(evwoiLabelField)
  evwoiLabel = evwoiTheme.ReturnValueString
    (evwoiTheme.GetLabelField.GetName,evwoiKey)


'Get the name of the watershed selected.


  evwoiQuery = "[ " + evwoiTheme.GetLabelField.
           asString + " ] = "  +
           evwoiLabel.asString.Quote
  aBitMap = BitMap.Make(evwoiTheme.GetFTab.
                GetNumRecords)
  aBitMap.ClearAll
  aBitMapSuc = evwoiTheme.GetFTab.Query
           (evwoiQuery,aBitMap,
            #VTAB_SELTYPE_NEW)
  if (aBitMapSuc.Not) then
    return NIL
  end
  recNum = aBitMap.GetNextSet(0)


  evwoiNmField = evwoiTheme.GetFTab.
           FindField("Name")
  if (evwoiNmField = NIL) then
    evwoiNmField = evwoiTheme.GetFTab.FindField
           ("Hu_name")
  end
  if (evwoiNmField = NIL) then
    evwoiNm = "Unnamed "
    if (evwoiTmp = 8) then
      evwoiNm = evwoiNm + "8-digit watershed"
    elseif (evwoiTmp = 11) then
      evwoiNm = evwoiNm + "11-digit watershed"
    else
      evwoiNm = evwoiNm + "14-digit watershed"
    end
  else
    evwoiNm = evwoiTheme.GetFTab.
         ReturnValueString
       (evwoiNmField,recNum)
  end


‘*****************************************
'
' Ask the user to decide what to do next.
'
‘*****************************************

  tmpAns = MsgBox.YesNoCancel("Watershed " +
           evwoiLabel.asString + nl +
           evwoiNm + nl +
           "Yes -- Use this " +
           "watershed." + nl +
           "No -- Select smaller " +
           "watersheds" + nl +
           "Cancel -- Select " +
           "another watershed." ,
           "Is this your " +
           "watershed of interest?" ,
           "TRUE")
  if (tmpAns = NIL) then


'User wants to select another watershed.


    return NIL
  elseif (tmpAns) then
```

```
'User wants to use this watershed as the
'watershed of interest. Get the WOI field, set
'the value to 99, and zoom to that watershed.

    evwoiField = evwoiTheme.GetFTab.FindField
                 ("WOI")
    evwoiTheme.GetFTab.SetEditable(TRUE)
    evwoiVal = evwoiTheme.GetFTab.ReturnValue
               (evwoiField,evwoiKey)
    evwoiTheme.GetFTab.SetValue(evwoiField,
                  evwoiKey,99)
    evwoiTheme.GetFTab.SetEditable(FALSE)
    aBitMap.ClearAll
    aBitMap.Set(evwoiKey)
    evwoiTheme.GetFTab.SetSelection(aBitMap)
    av.Run("EV.WOIZoom",evwoiTheme)
    av.Run("EV.WOIStreams",evwoiView)
    evwoiGUIs = av.GetProject.GetGUIs
    for each aGUI in evwoiGUIs
      if (aGUI.asString = "View") then
        aGUI.GetToolBar.SelectDefault
        break
      end
    end
    return NIL
  end

'*****************************************
'
' User wants to select a smaller watershed of
' interest. The hydrologic unit at which the user
' pointed is known. Make the next level of hydrologic
' units visible.
'
'*****************************************

'8-digit hydrologic units are visible, and the user
'clicked within an 8-digit hydrologic unit.

  if (evwoiTmp = 8) then

'Get the FTab for the 11-digit hydrologic units.

    evwoiSrcNm = SrcName.Make
          (ev_data + "\In_hu11.shp")
    evwoiHuc = Theme.Make(evwoiSrcNm)
    evwoiFTab = evwoiHuc.GetFTab

'Create the query needed to get the 11-digit
'hydrologic units associated with the 8-digit
'hydrologic unit the user selected.

    evwoiQuery = "[Huc_8] = " +
      evwoiLabel.asString.Quote
    theBitMap = evwoiFTab.GetSelection

    evwoiFTab.Query(evwoiQuery,theBitMap,
      #VTAB_SELTYPE_NEW)
    evwoiFTab.UpdateSelection

'Use the file name associated with the watershed
'of interest theme currently being used to
'save the selected data. Delete the watershed
'of interest theme and redraw the view.

    evwoiFileNm = FileName.Make
      (evwoiTheme.GetSrcName.asString)
    evwoiView.DeleteTheme(evwoiTheme)
    evwoiView.Invalidate
    File.Delete(evwoiFileNm)

'Export the selected 11-digit hydrologic units
'to the file name associated with the watershed of
'interest theme currently being used. Add the WOI
'field to the table.

    evwoiFTab = evwoiHuc.ExportToFTab
              (evwoiFileNm)
    evwoiWOIField = Field.Make
       ("WOI",#FIELD_SHORT,2,0)
    tmpList = {}
    tmpList.Add(evwoiWOIField)
    evwoiFTab.SetEditable(TRUE)
    evwoiFTab.AddFields(tmpList)
    evwoiFTab.SetEditable(FALSE)

'Make the theme from the FTab, set the legend to
'uniform color where the color is cyan, and display
'that theme.

    evwoiFTheme = FTheme.Make(evwoiFTab)
    evwoiView.AddTheme(evwoiFTheme)
    av.Run ("EV.WOIColor",evwoiFTheme)
    evwoiFTheme.SetActive(TRUE)
```

```
    evwoiFTheme.SetVisible(TRUE)
    evwoiRect = Rect.MakeEmpty
    evwoiRect = evwoiRect.UnionWith
      (evwoiFTheme.ReturnExtent)
    if (evwoiRect.ReturnSize = (0@0) ) then
      evwoiView.GetDisplay.PanTo
        (evwoiRect.ReturnOrigin)
    else
      evwoiView.GetDisplay.SetExtent
        (evwoiRect.Scale(1.1))
    end
    evwoiFTheme.GetFTab.SetEditable(TRUE)


    evwoiField = evwoiFTheme.GetFTab.FindField
            ("WOI")
    for each tmpI in 0 .. (evwoiFTheme.GetFTab
                    .GetNumRecords - 1)
      evwoiFTheme.GetFTab.SetValue
            (evwoiField,tmpI,8)
    end
    evwoiFTheme.GetFtab.SetEditable(FALSE)


'11-digit hydrologic units are visible, and the user
'clicked within an 11-digit hydrologic unit.


  elseif (evwoiTmp = 11) then


'Get the 8-digit hydrologic unit code associated
'with the 11-digit hydrologic unit watersheds
'currently displayed. All 14-digit hydrologic units
'within the 8 are used during the evaluating process.


    evwoiFTab = evwoiTheme.GetFtab
    evwoiField = evwoiFTab.FindField("Huc_8")
    evwoiHuc8c = evwoiFTab.ReturnValue
            (evwoiField,1)


'Get the FTab for the 14-digit hydrologic units.


    evwoiSrcNm = SrcName.Make
        (ev_data + "/In_hu14.shp")
    evwoiHuc = Theme.Make(evwoiSrcNm)
    evwoiFTab = evwoiHuc.GetFTab


'Use the file name associated with the watershed
'of interest theme currently being used to
'save the selected data. Delete the watershed
'of interest theme and redraw the theme.


    evwoiFileNm = FileName.Make
      (evwoiTheme.GetSrcName.asString)
    evwoiView.DeleteTheme(evwoiTheme)
    evwoiView.Invalidate


'Create the query needed to get the 14-digit
'hydrologic units associated with the 8-digit
'hydrologic unit the user selected.


    evwoiQuery = "[Huc_8] = " +
      evwoiHuc8c.Quote
    theBitMap = evwoiFTab.GetSelection
    evwoiFTab.Query(evwoiQuery,theBitMap,
      #VTAB_SELTYPE_NEW)
    evwoiFTab.UpdateSelection


'Export the selected 14-digit hydrologic units
'to the file name associated with the watershed of
'interest theme currently being used. Add the WOI
'field to the table.


    evwoiFTab = evwoiHuc.ExportToFTab
            (evwoiFileNm)
    evwoiWOIField = Field.Make
      ("WOI",#FIELD_SHORT,2,0)
    tmpList = {}
    tmpList.Add(evwoiWOIField)
    evwoiFTab.SetEditable(TRUE)
    evwoiFTab.AddFields(tmpList)
    evwoiFTab.SetEditable(FALSE)
    evwoiFTheme = FTheme.Make(evwoiFTab)
    evwoiView.AddTheme(evwoiFTheme)
    av.Run ("EV.WOIColor",evwoiFTheme)
    evwoiFTheme.SetActive(TRUE)
    evwoiFTheme.SetVisible(TRUE)


'Build a query to display only the 14-digit
'hydrologic unit watershed.


    evwoiQuery = "[Huc_11] = " +
      evwoiLabel.asString.Quote
    theBitMap = evwoiFTab.GetSelection
    evwoiFTab.Query(evwoiQuery,theBitMap,
      #VTAB_SELTYPE_NEW)
    evwoiFTab.UpdateSelection
```

```
    evwoiFTheme.SetActive(TRUE)
    evwoiFTheme.SetVisible(TRUE)
    av.Run("EV.WOIZoom",evwoiFTheme)


'Set the field WOI to identify the location of
'each 14-digit hydrologic unit. Set the field to 8
'if the 14-digit hydrologic units are not in the
'11-digit hydrologic unit selected by the user. Set
'the value to 11 for the 14-digit hydrologic units
'within the 11-digit hydrologic unit selected by the
'user.



    evwoiFTab.SetEditable(TRUE)
    evwoiNumRec = evwoiFTab.GetNumRecords
    tmpI = 0
    theField = evwoiFTab.FindField("Huc_11")
    while (tmpI < evwoiNumRec)
      evwoiHuc11c = evwoiFTab.ReturnValueString
        (theField,tmpI)
      if (evwoiHuc11c = evwoiLabel.asString) then
        evwoiFTab.SetValueNumber
          (evwoiWOIField,tmpI,11)
      else
        evwoiFTab.SetValueNumber
          (evwoiWOIField,tmpI,8)
      end
      tmpI = tmpI + 1
    end
    evwoiFTab.setEditable(FALSE)


'User selected a 14-digit hydrologic unit watershed.
'Center on that and add the 1:100,000 streams.


  elseif (evwoiTmp = 14) then


'Get the 8-digit hydrologic unit code associated
'with the 11-digit hydrologic unit watersheds
'currently displayed.
    evwoiTheme.ClearSelection
    evwoiFTab = evwoiTheme.GetFTab
    evwoiField = evwoiFTab.FindField("WOI")
    evwoiQuery = "[Huc_14] = " +
      evwoiLabel.asString.Quote
    theBitMap = evwoiFTab.GetSelection
    evwoiFTab.Query(evwoiQuery,theBitMap,
      #VTAB_SELTYPE_NEW)
    evwoiFTab.UpdateSelection
    evwoiTheme.SetActive(TRUE)
    evwoiTheme.SetVisible(TRUE)
    av.Run("EV.WOIZoom",evwoiTheme)
    av.Run("EV.WOIStreams",evwoiView)
    evwoiFTab.SetEditable(TRUE)


'Set the value of WOI to 14 for the 14-digit
'hydrologic unit selected by the user.


    evwoiNumRec = evwoiFTab.GetNumRecords
    tmpI = 0
    theField = evwoiFTab.FindField("Huc_14")
    while (tmpI < evwoiNumRec)
      evwoiHuc14c = evwoiFTab.ReturnValueString
        (theField,tmpI)
      if (evwoiHuc14c = evwoiLabel.asString) then
        evwoiFTab.SetValueNumber
          (evwoiField,tmpI,14)
          break
      end
      tmpI = tmpI + 1
    end
    evwoiFtab.setEditable(FALSE)
  end
 end
end


'Beep if the feature could not be found by pointing.


if (NOT evwoiFound) then
  System.Beep
end
```

EV.WOIClick

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:51 2000

'Script executed when the WS icon is clicked.
'Allows the user to select an existing watershed
'of interest theme or create a new watershed of
'interest theme.

‘*****************************************
'
'actApplyScript -- Apply script currently active.
'aFTab -- FTab for a theme.
'aGUI -- Loop variable used to loop through the
'        GUIs in the project.
'ev_cwd -- Current working directory.
'EV_DATA -- System environment variable holding the
'        full pathname to the data base that
'        accompanies the application.
'ev_data -- Variable holding the full pathname to
'        the data base that accompanies the
'        application.
'EV_HOME -- System environment variable holding the
'        full pathname to the user area where
'        files are saved.
'ev_home -- Variable holding the full pathname to
'        the user area where files are saved.
'EV_TEMP -- System environment variable holding the
'        full pathname to the temporary work
'        area.
'evevenvDic -- Dictionary containing the system
'        environment variables, returned from
'        "EV.GetEnvVar."
'evwoiAllThemes -- List of all themes in
'        "evwoiView."
'evwoiFileNm -- File name for the watershed of
'        interest theme.
'evwoiFTab -- 8-digit hydrologic unit FTab.
'evwoiGUIs -- List of GUIs in the project.
'evwoiHuc -- 8-digit hydrologic unit theme.
'evwoiSrcNm -- Source name for the 8-digit
'        hydrologic units.
'evwoiTheme -- The watershed of interest theme.
'evwoiThemes -- Themes that contain the WOI field.
'evwoiTmpFileNm -- Default file name for the
'        watershed of interest theme.
'evwoiView -- View containing the watershed of
'        interest.
'evwoiWOIField -- Field used to identify the
'        watershed of interest.
'tmpAns -- Response to a command.
'tmpI -- Temporary counter.
'tmpMsg -- Message for the message box.
'tmpList -- List used to add fields to the tables.
'
‘*****************************************

‘*****************************************
'
' Get the system environment variables and the
' view.
'
‘*****************************************

evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("TEMP")
ev_cwd = FileName.GetCWD
evwoiView = av.Run("EV.ViewGet",
            "Watershed of Interest " +
            "Click Event")

‘*****************************************
'
' Get a list of themes, determine which
' themes show watersheds of interest, and get the
' active apply script.
'
‘*****************************************


'Find the script name of the active tool, which is
'the tool on which the user clicked. If this
'script was activated by split watershed or
'by cut watershed just move the stream coverage
'on top. Moving the stream coverage will be done
'later in the script.


evwoiGUIs = av.GetProject.GetGUIs
for each aGUI in evwoiGUIs
  if (aGUI.asString = "View") then
    actApplyScript = aGUI.GetToolBar.GetActive.
                GetApply
    break
  end
end
```

```
'*****************************************
'
' Select a watershed of interest.
'
'*****************************************

evwoiAllThemes = evwoiView.GetThemes
evwoiThemes = {}
evwoiTheme = NIL
evwoiThemes = av.Run("EV.WOIGetWOI",
                evwoiView.GetThemes)

'There is at least one watershed of interest
'theme. Ask the user to select a theme to use.
'Depending on the active tool, the CANCEL will
'create a new watershed of interest theme or
'return from this script.


if (evwoiThemes.Count > 0) then
  aThemesString = NIL
  for each aTheme in evwoiThemes
    if (aThemesString = NIL) then
      aThemesString = aTheme.asString
    else
      aThemesString = aThemesString + nl +
aTheme.asString
    end
  end
  tmpAns = MsgBox.YesNo(aThemesString,
                  "Do you wish to use " +
                  "one of the following " +
                  "watersheds of interest?",
                  TRUE)

  while (tmpAns)
    evwoiTheme = MsgBox.List(evwoiThemes,
        "Select one of the following " +
        "watershed of interest themes.",
        "Select a watershed of interest.")
    if (NOT (evwoiTheme = NIL)) then
      tmpAns = FALSE
    else
      tmpAns = MsgBox.YesNo("Do you really " +
                  "want to quit?",
                  "Identifying watershed of " +
                  "interest",TRUE)
      if (tmpAns) then
        tmpAns = FALSE
      end
    end
  end
end

'There are no watershed of interest themes or the
'user wishes to create a new watershed of interest
'theme. Create a new theme using the 8-digit
'hydrologic units, add the WOI field to the theme,
'and make the theme active and visible.


if (evwoiTheme = NIL) then


'If the tool for selecting watershed of interest
'is not the watershed selection tool, return
'from the script and do nothing.


  if (NOT (actApplyScript = "EV.WOIApply")) then
    return NIL
  end


'Create the theme by physically making a copy of
'the 8-digit hydrologic unit shape file and
'adding the WOI column to the attribute table.
'The WOI column will be used to identify the
'8-, 11-, and 14-digit hydrologic units selected.
'Physically copy the 8-digit theme to an area
'selected by the user.

  evwoiSrcNm = SrcName.Make
            (ev_data + "/In_hu8.shp")
  evwoiHuc = Theme.Make(evwoiSrcNm)
  evwoiFTab = evwoiHuc.GetFTab
  evwoiTmpFileNm = av.GetProject.MakeFileName
              ("WOI", "shp")
  evwoiFileNm = FileDialog.Put(evwoiTmpFileNm,
              "WOI*.shp",
              "Create a new watershed of interest?")
  if (evwoiFileNm = NIL) then
    return NIL
  end


'Add the WOI column to the theme, add the theme to the
'view, and make the theme visible and active.


  evwoiFTab = evwoiHuc.ExportToFTab(evwoiFileNm)
  evwoiWOIField = Field.Make("WOI",
                      #FIELD_SHORT,2,0)
  tmpList = {}
  tmpList.Add(evwoiWOIField)
  evwoiFTab.SetEditable(TRUE)
  evwoiFTab.AddFields(tmpList)
  evwoiFTab.SetEditable(FALSE)
  evwoiFTheme = FTheme.Make(evwoiFTab)
```

```
  evwoiView.AddTheme(evwoiFTheme)
  evwoiFTheme.SetActive(TRUE)
  evwoiFTheme.SetVisible(TRUE)



'Set the legend to uniform color. Color of cyan.
'Zoom to the proper resolution.


  av.Run ("EV.WOIColor",evwoiFTheme)
  passDic = Dictionary.Make(2)
  passDic.Set("Theme",evwoiFTheme)
  passDic.Set("View",evwoiView)
  av.Run("EV.Zoom2FullExtent",passDic)


else


'The user selected a theme. Make that theme active
'and visible. Zoom to the selected records.


  tmpI = -1
  for each aTheme in evwoiAllThemes
    tmpI = tmpI + 1
    if (aTheme = evwoiTheme) then
      break
    end
  end
  evwoiAllThemes.Shuffle
            (evwoiAllThemes.Get(tmpI),0)
  evwoiView.InvalidateTOC(NIL)
  evwoiTheme.UpdateLegend
  evwoiView.Invalidate
  evwoiTheme.SetActive(TRUE)
  evwoiTheme.SetVisible(TRUE)


'Set the legend of the theme to a uniform color,
'with the color cyan.


  av.Run("EV.WOIColor",evwoiTheme)


'Zoom in to the proper HUC.


  av.Run("EV.WOIZoom",evwoiTheme)
end


'If the split or cut tools got to this script,
'add the stream coverage.


if ((actApplyScript.Contains("SplitWater")) or
    (actApplyScript.Contains("CutWater"))) then
  av.Run("EV.WOIStreams","")
end
```

EV.WOIColor

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:46:59 2000

'Script is passed a theme. The script then sets
'the theme legend to uniform color. The color is
'cyan.

'*****************************************
'
'evwoiCTheme -- Theme passed from the calling script.
'evwoiSymbolList -- List of symbols in the legend.
'evwoiColor -- The color applied to the legend.

'*****************************************
'
' Get the theme passed and the theme legend.
' Set the legend to a uniform color of cyan.
'
'*****************************************

evwoiCTheme = SELF

evwoiSymbolList = SymbolList.FromList
        (evwoiCTheme.GetLegend.GetSymbols)
evwoiColor = Color.GetCyan
evwoiSymbolList.UniformColor(evwoiColor)
```

EV.WOICut

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:47:07 2000

'A script allowing the user to cut a piece of a
'watershed and use that piece as the watershed
'of interest.

‘*****************************************
'
'actThemes -- List of active themes in "theView."
'aTheme -- Loop variable used to loop through all
'         the themes in "theView."
'diffshp -- The watershed shape with the user entered
'         polygon removed.
'newRec -- Record number of the next record added
'         to theFTab.
'numRec -- Number of features in the watershed of
'         interest theme.
'passDic -- Dictionary used to pass information
'         between scripts.
'recNum -- Record number of the feature containing
'         "userPoly."
'shpField -- Field in the table that contains the geometry
'         of the features in the watershed of interest
'         theme.
'theFTab -- FTab for the watershed of interest theme.
'theTheme -- The watershed of interest theme.
'theView -- The active view.
'tmpField -- Temporary variable used to find a
'         particular field in a table.
'userPoly -- The outline of the cut entered by the
'         user.
'wtdpoly -- The geometry of the feature being
'         examined.
'
‘*****************************************

‘*****************************************
'
' Get the view, the list of themes, the active
' themes, the watershed of interest theme, and
' the user entered polygon.
'
‘*****************************************

'Get the view. “WOIClick” asks the user to identify
'the View and, thus, the WOI is in the active
'document.


theView = av.GetActiveDoc


'Get the first theme in the list that is a
'watershed of interest theme. The “WOIClick”
'event moves the selected watershed of interest to
'the top of the table of contents unless a stream
'coverage is the only theme listed before the
'watershed of interest.


for each aTheme in theView.GetThemes
  tmpField = aTheme.GetFTab.FindField("WOI")
  if (tmpField <> NIL) then
    theTheme = aTheme
    break
  end
end


'Make all themes inactive except the WOI.

actThemes = { }
for each aTheme in theView.GetThemes
  if (aTheme.IsActive) then
    aTheme.SetActive(FALSE)
    actThemes.Add(aTheme)
  end
end


'Get the polygon entered by the user.


userPoly = theView.ReturnUserPolygon


‘*****************************************
'
' Cut the watershed.
'
‘*****************************************

'If no user polygon is entered, then return with no
'action.



if (userPoly.IsNull) then
  return NIL
end


'Find the record that contains the user entered
'polygon.
```

```
numRec = theTheme.GetFTab.GetNumRecords
shpField = theTheme.GetFTab.FindField("Shape")
recNum = NIL
for each tmpI in 0 .. (numRec - 1)
  wtdpoly = theTheme.GetFTab.ReturnValue
          (shpField,tmpI)
  if (wtdpoly.contains(userPoly)) then
    recNum = tmpI
    break
  end
end

if (recNum = NIL) then
  MsgBox.Error("The graphic you drew is not " +
            "contained in any of the " +
            "watersheds in the theme " +
            theTheme.asString,
            "Cuting a watershed.")
  return NIL
end


'Make the selected theme active, get the FTab
'and make it editable


theTheme.SetActive(TRUE)
theFTab = theTheme.GetFTab
theFTab.SetEditable(TRUE)
theFTab.BeginTransaction


'Add the user polygon to the FTab as a shape.

newRec = theFTab.AddRecord
theFTab.SetValue(shpField,newRec,userPoly)


'Remove the new shape from the watershed and
'save the change.

diffshp = wtdpoly.ReturnDifference(userPoly)
theFTab.SetValue(shpField,recNum,diffshp)


'Create the spatial index.


theFTab.CreateIndex(shpField)


'End the edit process.


theTheme.GetFTab.EndTransaction
theTheme.GetFTab.SetEditable(FALSE)


'Update the new record in the FTab.

passDic = Dictionary.Make(3)
passDic.Set("THEME",theTheme)
passDic.Set("OLD",recNum)
passDic.Set("NEW",newRec)
av.Run("EV.WOISetVal",passDic)


'Repaint the theme.


theTheme.Invalidate(TRUE)


'Return all themes to their original state.


for each aTheme in theView.GetThemes
  if (aTheme.IsActive) then
    aTheme.SetActive(FALSE)
  end
end
for each aTheme in actThemes
  aTheme.SetActive(TRUE)
end
```

EV.WOIGetWOI

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:47:07 2000

'Script that finds the watershed of interest
'themes in the active view.

'*****************************************
'
'theThemes -- List of themes passed from the calling
'         script.
'rtnThemes -- List of watershed of interest themes
'         found by the script.
'aTheme -- Loop variable used to loop through all

"         themes passed by the calling script.

'*****************************************
theThemes = SELF

rtnThemes = {}
for each aTheme in theThemes
  if (aTheme.CanSelect) then
    if (NOT (aTheme.GetFTab.FindField("WOI")
          = NIL)) then
      if (NOT (aTheme.asString.Contains("Hash_"))) then
        rtnThemes.Add(aTheme)
      end
    end
  end
end
return rtnThemes
```

EV.WOIHuclevel

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:50:42 2000

‘*****************************************
'
'aField -- Loop variable used to loop through
'         "evwoiFTabFields."
'aTheme -- Theme passed from the calling program.
'evwoiFTab -- FTab for the theme passed from the
'         calling program.
'evwoiFTabFields -- Fields in "evwoiFTab."
'evwoiTmp -- Temporary counter signifying the
'         hydrologic unit level of this theme.
'evwoiLabelField -- Field containing the hydrologic
'         unit code.
'passDic -- Dictionary containing the information
'         passed from the calling script.
'newRec -- Record number added to the theme's FTab.
'recNum -- Record number of the modified feature.
'rtnDic -- Dictionary used to return information
'         to the calling script.
'theTheme -- Theme passed from the calling script.
'
‘*****************************************

‘*****************************************
'
' Get the theme passed from the calling script,
' the FTab, and the fields in the FTab.
'
‘*****************************************

aTheme = SELF
evwoiFTab = aTheme.GetFTab
evwoiFTabFields = evwoiFTab.GetFields

‘*****************************************
'
' Use the fields in the table to determine which
' field should be used as the label field
' (the hydrologic unit code). Set the indicator
' to the proper level of hydrologic unit.
'
‘*****************************************

evwoiTmp = 0
evwoiLabelField = NIL
for each aField in evwoiFTabFields
  if (aField.asString = "Huc_8") then
    if (evwoiTmp < 8) then
      evwoiTmp = 8
      evwoiLabelField = aField
    end
  end
  if (aField.asString = "Huc_11") then
    if (evwoiTmp < 11) then
      evwoiTmp = 11
      evwoiLabelField = aField
    end
  end
  if (aField.asString = "Huc_14") then
    if (evwoiTmp < 14) then
      evwoiTmp = 14
      evwoiLabelField = aField
    end
  end
end


‘*****************************************
'
' Build the return dictionary and return.
'
‘*****************************************


rtnDic = Dictionary.Make(2)
rtnDic.Set("LABEL",evwoiLabelField)
rtnDic.SET("TMP",evwoiTmp)
return rtnDic
```

EV.WOISetVal

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:50:51 2000

'Update the other entries in FTab after a cut or
'split watershed operation.

'*****************************************
'
'aField -- Loop variable used to look at all
'         fields in "theFields."
'evtmp -- Huc level for this theme.
'passDic -- Dictionary containing the information
'         passed from the calling script.
'newName -- Name of the new feature.
'newRec -- Record number added to the theme's FTab.
'oldVal -- Value of the field for the watershed that
'         was modified.
'recNum -- Record number of the modified feature.
'rtnDic -- Dictionary containing the theme's huc
'         level.
'theFields -- Fields in "theFTab."
'theFTab -- FTab for "theTheme."
'theTheme -- Theme passed from the calling script.
'wtrshed -- Geometry of the new feature.
'
'*****************************************

'*****************************************
'
' Get the information passed to this script,
' establish variable values, and determine the
' huc level of the watershed of interest theme.
'
'*****************************************

passDic = SELF
theTheme = passDic.Get("THEME")
recNum = passDic.Get("OLD")
newRec = passDic.Get("NEW")


'Determine the huc level.
rtnDic = av.Run("EV.WOIHuclevel",theTheme)
evtmp = rtnDic.Get("TMP")

'*****************************************
'
'Update each field in the FTab
'
'*****************************************

'Get the FTab and the fields. Make the table
'editable.


theFTab = theTheme.GetFTab
theFTab.SetEditable(TRUE)
theFields = { }
theFields = theFTab.GetFields

'Loop through each field and populate the fields
'for the newly created feature.

for each aField in theFields
  if (aField.asString = "Shape") then
    wtrshed = theFTab.ReturnValue(aField,newRec)
  elseif (aField.asString = "Area") then
    theFTab.SetValue(aField,newRec,
                  wtrshed.ReturnArea)
  elseif (aField.asString = "Perimeter") then
    theFTab.SetValue(aField,newRec,
                  wtrshed.ReturnLength)
  elseif (aField.asString.Contains("name")) then
    newName = "Part of " +
            theFTab.ReturnValue(aField,recNum)
    theFTab.SetValue(aField,newRec,newName)
  elseif (aField.asString = "WOI") then
    theFTab.SetValue(aField,newRec,(evtmp*2))
  else
    oldVal = theFTab.ReturnValue(aField,recNum)
    theFTab.SetValue(aField,newRec,oldVal)
  end
end
theFTab.SetEditable(FALSE)
return NIL
```

EV.WOISplitWatershed

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:51:03 2000

'Script to split a feature into two pieces.

‘*****************************************
'
'aBitMap -- Bit map used to indicate which records
'         have been used or which records to select.
'abitlist -- List used to create and modify a
'         bit map.
'actThemes -- List of active themes in "theView."
'aLine -- The line entered by the user that is
'         used to split the feature into two pieces.
'aTheme -- The first theme in the View that
'         contains the field "WOI," (the theme
'         being split).
'evwoiFTab -- FTab for the theme being split,
'         "aTheme."
'evwoiLabelField -- Field containing the hydrologic
'         unit code.
'evwoiTmp -- Temporary variable that contains the huc
'        level indicator.
'newNm -- Name of the portion of the hydrologic
'         unit that has been split.
'nmField -- Name field in "evwoiFTab."
'recNm -- Name of the hydrologic unit being split.
'recNum -- Record number in a table.
'result -- Contains the geometry after the feature
'         has been split.
'rtnDict -- Dictionary returned from a script
'         containing the huc level for "aTheme."
'shpField -- Field containing the geometry of the
'         features in "aTheme."
'theTheme -- Loop variable used to loop through
'         the themes in the view.
'theType -- Type of geometry contained in shpField.
'theView -- The active view.
'tmpAns -- Response to a command.
'tmpHuc -- Hydrologic unit code of the record
'         being modified.
'tmpField -- Variable used to see if "WOI" is a field
'         in theFTab for "theTheme."
'woiField -- "WOI" field in the "evwoiFTab."
'
‘*****************************************

‘*****************************************
'
' Get the view, the line entered by the user,
' the watershed of interest theme, and a list of
' of active themes.
'
‘*****************************************

theView = av.GetActiveDoc
av.ShowMsg("Getting line used to divide the " +
        "watersheds.")
aLine = theView.ReturnUserPolyLine
if (aLine.isNull) then
  return NIL
end


'Get the first theme in the list. The “WOIClick”
'event moves the selected watershed of interest to
'the top of the table of contents unless a stream
'coverage overlies it.


av.ShowMsg("Getting the watershed of interest.")
for each theTheme in theView.GetThemes
  tmpField = theTheme.GetFTab.FindField("WOI")
  if (tmpField <> NIL) then
    aTheme = theTheme
    break
  end
end


'Make all themes inactive except the WOI.


actThemes = { }
for each theTheme in theView.GetThemes
  if (theTheme.IsActive) then
    theTheme.SetActive(FALSE)
    actThemes.Add(theTheme)
  end
end
aTheme.SetActive(TRUE)
if (aTheme.CanEdit.Not)then
   MsgBox.Error (aTheme.asString +
            " can not be edited. See your " +
            "System Administrator for " +
            "assistance.","EV.WOISplitWatershed")
end

‘*****************************************
'
' Split the watershed.
'
‘*****************************************
```

```
av.ShowMsg("Splitting the watershed.")
av.UseWaitCursor

'Using the line entered by the user to split the
'watershed.


theView.SetEditableTheme(aTheme)
evwoiFTab = aTheme.GetFTab
evwoiFTab.StartEditingWithRecovery
evwoiFTab.BeginTransaction
shpField = evwoiFTab.FindField("Shape")
theType = shpField.GetType
if ((theType = #FIELD_SHAPEPOLY) or
    (theType = #FIELD_SHAPELINE)) then
  result = aTheme.Split(aLine)
end
evwoiFTab.EndTransaction
evwoiFTab.StopEditingWithRecovery(TRUE)


‘*****************************************
'
' Populate the data fields for the newly
' created feature.
'
‘*****************************************

'Determine the level of HUCs in this theme.


rtn_Dic = av.Run("EV.WOIHuclevel",aTheme)
evwoiLabelField = rtn_Dic.Get("LABEL")
evwoiTmp = rtn_Dic.Get("TMP")


'Find the huc that was modified and set values.


aBitMap = evwoiFTab.GetSelection
recNum = aBitMap.GetNextSet(0)
tmpHuc = evwoiFTab.ReturnValue
              (evwoiLabelField,recNum)


'Find the records that need to be updated.


aQuery = "[ " + evwoiLabelField.asString + " ] = " +
       tmpHuc.asString.Quote
aBitMap = BitMap.Make(evwoiFTab.GetNumRecords)
aBitMap.ClearAll
evwoiFTab.SetSelection(aBitMap)
tmpAns = evwoiFTab.Query(aQuery,aBitMap,
                                  #VTAB_SELTYPE_NEW)

if (tmpAns) then
  evwoiFTab.SetSelection(aBitMap)
  theView.Invalidate
end


'Update the WOI and name fields.


recNum = 0
woiField = evwoiFTab.FindField("WOI")
nmField = evwoiFTab.FindField("Name")
if (nmField = NIL) then
  nmField = evwoiFTab.FindField("Hu_name")
end
evwoiFTab.SetEditable(TRUE)
abitlist = {}
for each abit in aBitMap
  abitlist.Add(abit)
end
while (recNum >= 0)
  recNum = aBitMap.GetNextSet(recNum)
  if (recNum > 0 ) then
    evwoiFTab.SetValue(woiField,
                  recNum,evwoiTmp*2)
    recNm = evwoiFTab.ReturnValue(nmField,recNum)
    newNm = "Part of " + recNm
    evwoiFTab.SetValue(nmField,recNum,newNm)
  end
end


'Create a shape index.


shpField = evwoiFTab.FindField("Shape")
evwoiFTab.CreateIndex(shpField)
evwoiFTab.SetEditable(FALSE)


'Zoom to the selected watersheds.


av.Run("EV.WOIZoom",aTheme)


'Return themes to their original state.


for each aTheme in theView.GetThemes
  if (aTheme.IsActive) then
    aTheme.SetActive(FALSE)
  end
end
```

```
for each aTheme in actThemes                    end
  aTheme.SetActive(TRUE)
```

EV.WOIStreams

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:51:11 2000

'This script loads the 1:100,000 stream coverage
'into the view as an aid to the user. If the
'coverage already exists in the view, the script
'will move the coverage to the top of the table of
'contents and make the coverage visible.

‘*****************************************
'
'aTheme -- Loop variable used to look at each
'         theme in "evwoistrView."
'EV_DATA -- System environment variable holding the
'         full pathname to the data base that
'         accompanies the application.
'ev_data -- Variable holding the full pathname to
'         the data base that accompanies the
'         application.
'EV_HOME -- System environment variable holding the
'         full pathname to the user area where
'         files are saved.
'ev_home -- Variable holding the full pathname to
'         the user area where files are saved.
'ev_pwd -- Project working directory.
'EV_TEMP -- System environment variable holding the
'         full pathname to the temporary work
'         area.
'ev_temp -- Variable holding the full pathname to
'         the temporary work area.
'evevenvDic -- Dictionary containing the system
'         environment variables, returned from
'         "EV.GetEnvVar."
'evwoiColor -- Color being assigned to the legend.
'evwoiSymbolList -- List of graphic symbols in the
'         view.
'evwoiStreams -- 1:100,000 stream theme.
'evwoistrSrcNm -- File name of the shape file
'         containing the 1:100,000 stream
'         coverage.
'evwoistrThemes -- Themes in "evwoistrView."
'evwoistrView -- Active view.
'tmpCnt -- Temporary counter.
'tmpSrcNm -- Source name of the theme being
'         examined.
'
‘*****************************************
'
' Get the system environment variables and
' the view.
'
‘*****************************************

'Get the system environment variables

evevenvDic = av.Run("EV.GetEnvVar","")
ev_data = evevenvDic.Get("EV_DATA")
ev_home = evevenvDic.Get("EV_HOME")
ev_temp = evevenvDic.Get("EV_TEMP")

'The view is passed from the calling script.

evwoistrView = av.GetActiveDoc

‘*****************************************
'
' Determine if the 1:100,000 theme is already
' part of the view.
'
‘*****************************************

'Get a list of all themes in the view. Determine if
'the 1:100,000 stream coverage exists. The variable
'"tmpCnt" will be used to reorder the table of
'contents, if the theme exists in the table of
'contents.

evwoistrThemes = evwoistrView.GetThemes
evwoistrSrcNm = NIL
tmpCnt = -1
for each aTheme in evwoistrThemes
  tmpSrcNm = aTheme.GetSrcName
  tmpCnt = tmpCnt + 1
  if (tmpSrcNm.asString.Contains("Stream100.shp"))
       then
    evwoistrSrcNm = tmpSrcNm
    aTheme.SetVisible(TRUE)
    break
  end
end

‘*****************************************
'
' Add or move the 1:100,000 stream theme.
'
```

```
'*****************************************

'If the 1:100,000 stream theme does not exists in
'the view, add the theme, make the lines green, and
'make the theme visible.

if (evwoistrSrcNm = NIL) then
  evwoistrSrcNm = SrcName.Make
        (ev_data + "/stream100.shp")
  evwoiStreams = Theme.Make(evwoistrSrcNm)
  evwoistrView.AddTheme(evwoiStreams)
  evwoiSymbolList = SymbolList.FromList
            (evwoiStreams.GetLegend.GetSymbols)
  evwoiColor = Color.GetGreen
  evwoiSymbolList.UniformColor(evwoiColor)
  evwoiStreams.SetVisible(TRUE)
else

'The 1:100,000 stream theme exists. Move the theme to
'the top of the table of contents and redraw the view.

  evwoiStreams = Theme.Make(evwoistrSrcNm)
  evwoistrThemes.Shuffle
    (evwoistrThemes.Get(tmpCnt),0)
  evwoistrView.InvalidateTOC(NIL)
  evwoiStreams.UpdateLegend
  evwoistrView.Invalidate
end
```

EV.WOIZoom

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:51:21 2000

'Modified ArcView's "View.ZoomSelected" to look
'at the theme passed.

‘*****************************************
'
'evwoiRect -- Area that will fill the view.
'evwoiZoomDis -- Display showing the view.
'evwoiZoomTheme -- Theme passed from the calling
'         script.
'evwoiZoomView -- View containing the theme.
'
‘*****************************************

‘*****************************************
'
' Get the theme and the view passed. Make sure
' the theme can be selected then get the view area
' and zoom to the view area.
'
‘*****************************************

evwoiZoomTheme = SELF
evwoiZoomView = evwoiZoomTheme.GetView

for each aTheme in evwoiZoomView.GetThemes
  if (NOT (aTheme = evwoiZoomTheme)) then
    aTheme.SetActive(FALSE)
    aTheme.SetVisible(FALSE)
  end
end
evwoiRect = Rect.MakeEmpty
evwoiRect = evwoiRect.UnionWith
    (evwoiZoomTheme.GetSelectedExtent)


'Zoom to the appropriate area.


if (evwoiRect.IsEmpty) then
  evwoiRect = evwoiZoomTheme.ReturnExtent
  evwoiZoomView.GetDisplay.SetExtent
    (evwoiRect.Scale(1.1))
elseif (evwoiRect.ReturnSize = (0@0) ) then
  evwoiZoomView.GetDisplay.PanTo
    (evwoiRect.ReturnOrigin)
else
  evwoiZoomView.GetDisplay.SetExtent
    (evwoiRect.Scale(1.1))
end


'Redraw the view.


evwoiZoomDis = evwoiZoomView.GetDisplay
evwoiZoomView.Draw(evwoiZoomDis)
```

EV.Zoom2FullExtent

```
'Created by James (Jay) L. Kiesler, Jr.
'Created on Wed Apr 26 15:51:29 2000

'Script to zoom to the full extent of the the view
'passed to this script. Modification of ESRI's
'"View.ZoomFullExtent." Only look at the view passed
'to the script and not the active document.

‘*****************************************
'
'evz2feView -- View passed from the calling script.
'evz2feExtent -- Extent of the view.
'
‘*****************************************

‘*****************************************
'
' Get the view passed from the calling script
' and reset the extent to the full extent of the
' view.
'
‘*****************************************

'Get the view passed to the script and the extent of
'that view.
```

```
theDic = SELF
evz2feView = theDic.Get("View")
evz2feTheme = theDic.Get("Theme")
for each aTheme in evz2feView.GetThemes
  if (NOT (aTheme = evz2feTheme)) then
    aTheme.SetActive(FALSE)
    aTheme.SetVisible(FALSE)
  end
end
evz2feExtent = evz2feView.ReturnExtent
'Zoom to the full extent.


if (evz2feExtent.IsEmpty) then
  return NIL

elseif ( evz2feExtent.ReturnSize = (0@0) ) then
  evz2feView.GetDisplay.PanTo
       (evz2feExtent.ReturnOrigin)
else
  evz2feView.GetDisplay.SetExtent
       (evz2feExtent.Scale(1.1))
  av.GetProject.SetModified(true)
end
evwoiZoomDis = evz2feView.GetDisplay
evz2feView.Draw(evwoiZoomDis)
```

APPENDIX 2. LISTING OF DATA LAYERS AND METADATA ABOUT EACH LAYER

County.shp

Table county.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
Fips_cnty – Federal Information Processing County Code
In_cty – Two-digit State of Indiana County Code
St – Two-letter Postal abbreviation for the state
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles

In_hu8.shp

Table in_hu8.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Name – Name of the hydrologic unit
States – States in which the hydrologic unit lie
Published– Published area of the hydrologic unit, in many cases this includes area outside Indiana
Huc_8 – 8-digit hydrologic unit code identifying the hydrologic unit

In_hu11.shp

Table in_hu11.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Huc_11 – 11-digit hydrologic unit code identifying the 11-digit hydrologic unit
Huc_8 – 8-digit hydrologic unit code within which the 11-digit hydrologic unit is

In_hu14.shp

Table in_hu14.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Hu_acres – Area of the feature, in acres
Noncontrib – Area of the feature that does not directly contribute to surface-water runoff, in acres
Sq_miles – Area of the feature, in square miles
Noncon_sqm – Area of the feature that does not directly contribute to surface-water runoff, in square miles
Huc_14 – 14-digit hydrologic unit code identifying the hydrologic unit
Huc_11 – 11-digit hydrologic unit code within which the 14-digit hydrologic unit is
Huc_8 – 8-digit hydrologic unit code within which the 14-digit hydrologic unit is

Acres_corn.shp

Source of corn values – Written Communication, September 17, 1996, U.S. Department of Agriculture, National Agricultural Statistics Service, Indiana Agricultural Statistics Service Agricultural Experiment Station, Purdue University

Table acres_corn.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Corn – Acres of the feature planted in corn

Acres_soy.shp

Source of soybean values – Written Communication, September 17, 1996, U.S. Department of Agriculture, National Agricultural Statistics Service, Indiana Agricultural Statistics Service Agricultural Experiment Station, Purdue University

Table acres_soy.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Soybean – Acres of the feature planted in soybeans

Acres_wheat.shp

Source of wheat values – Written Communication, September 17, 1996, U.S. Department of Agriculture, National Agricultural Statistics Service, Indiana Agricultural Statistics Service Agricultural Experiment Station, Purdue University

Table acres_wheat.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes

St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Wheat – Acres of the feature planted in wheat

Acres_planted.shp

Source of values – Written Communication, September 17, 1996, U.S. Department of Agriculture, National Agricultural Statistics Service, Indiana Agricultural Statistics Service Agricultural Experiment Station, Purdue University

Table acres_planted.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Plt_acres – Acres of the feature planted in corn, soybeans, and wheat. Determined by summing the values in the corn, soybean, and wheat themes.
Zcnty – Percentage of the county planted in corn, soybeans, and wheat. Determined by dividing Plt_acres by acres.

Crop_per_n.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table crop_per_n.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Crop_nit – Percentage of the crop needs for nitrogen met by animal waste produced in the county

Crop_per_p.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table crop_per_p.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Crop_pho – Percentage of the crop needs for phosphorus met by animal waste produced in the county

Crop_per_k.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table crop_per_k.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Crop_pot – Percentage of the crop needs for potassium met by animal waste produced in the county

Animal_was_n.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table animal_was_n.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Waste_pot – Estimate of the pounds of potassium contained in animal waste produced in the county

Animal_was_p.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table animal_was_p.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Waste_pho – Estimate of the pounds of phosphorus contained in animal waste produced in the county

Animal_was_k.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table animal_was_k.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Waste_pot – Estimate of the pounds of potassium contained in animal waste produced in the county

Animal_was_tnut.shp

Source of values – Written Communication, February 1997, Barbara Vining, U.S. Department of Agriculture, National Resource Conservation Service of work done by the U.S. Environmental Protection Agency and the U.S. Department of Agriculture, National Resource Conservation Service, Economic Research Service

Table animal_was_tnut.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Total_nut – Estimate of the pounds of nitrogen, phosphorus, and potassium contained in animal waste produced in the county. Determined by summing the values in the pounds of nitrogen, phosphorus, and potassium themes.

Horses.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table horses.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Horses – Head of horses in a county

Cattle.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table cattle.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Cattle – Head of cattle in a county

Cattle_dairy.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table cattle_dairy.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Dairy – Head of dairy cattle in a county

Cattle_ndairy.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table cattle_ndairy.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Non_dairy – Head of nondairy cattle in a county

Poultry.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table poultry.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Poultry – Head of poultry in a county

Sheep.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table sheep.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Sheep – Head of sheep in a county

Swine.shp

Source of values – U.S. Bureau of the Census, 1994, 1992 Census of Agriculture, Volume 1 Geographic Area Series, Part 14 Indiana State and County Data, U.S. Department of Commerce, Economics and Statistics Administration, Bureau of the Census

Table swine.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Swine – Head of swine in a county

Population.shp

Source of values – U.S. Bureau of the Census, 2000, County Population Estimates and Demographic Components of Population Change: Annual Time Series, July 1, 1990 to July 1, 1999, from URL http://www.census.gov/population/estimates/county/co-99-8/99c8-18.txt, accessed April 3, 2000, text format.

Table population.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z990_base – 1990 base census population
Z990_est – Estimated population, July 1, 1990
Z991_est – Estimated population, July 1, 1991
Z992_est – Estimated population, July 1, 1992
Z993_est – Estimated population, July 1, 1993
Z994_est – Estimated population, July 1, 1994
Z995_est – Estimated population, July 1, 1995
Z996_est – Estimated population, July 1, 1996
Z997_est – Estimated population, July 1, 1997
Z998_est – Estimated population, July 1, 1998
Z999_est – Estimated population, July 1, 1999

Population_cng.shp

Source of values – U.S. Bureau of Census, 2000, County Population Estimates and Demographic Components of Population Change: Annual Time Series, July 1, 1990 to July 1, 1999, from URL http://www.census.gov/population/estimates/county/co-99-8/99c8-18.txt, accessed April 3, 2000, text format

Table population_cng.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z990_base – April 1, 1990, base population
Z0_91_cng – Estimated population change, July 1, 1990, to July 1, 1991
Z1_92_cng – Estimated population change, July 1, 1991, to July 1, 1992
Z2_93_cng – Estimated population change, July 1, 1992, to July 1, 1993
Z3_94_cng – Estimated population change, July 1, 1993, to July 1, 1994
Z4_95_cng – Estimated population change, July 1, 1994, to July 1, 1995
Z5_96_cng – Estimated population change, July 1, 1995, to July 1, 1996
Z6_97_cng – Estimated population change, July 1, 1996, to July 1, 1997
Z7_98_cng – Estimated population change, July 1, 1997, to July 1, 1998
Z8_99_cng – Estimated population change, July 1, 1998, to July 1, 1999
Z5_99_cng – Estimated population change, July 1, 1995, to July 1, 1999
Z0_99_cng – Estimated population change, July 1, 1990, to July 1, 1999

Emp_ag_serv_forest.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_ag_serv_forest.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_construction.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_construction.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_farm.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_farm.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_nonfarm.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_nonfarm.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_proprietors.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_proprietors.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_farm_proprietors.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_farm_proprietors.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_nonfarm_proprietor.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_nonfarm_proprietor.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_government.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_government.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_federal.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_federal.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_state_local.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_state_local.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_state.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_state.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_local.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_local.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_military.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_military.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_finance.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_finance.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_manufacture.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_manufacture.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_mining.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_mining.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_private.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_private.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_retail.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_retail.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_service.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_service.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_transportation.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_transportation.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_wholesale.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_wholesale.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_wage_salary.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_wage_salary.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Emp_full_parttm.shp

Source of values – U.S. Department of Commerce, Bureau of Economic Analysis, 2000, Annual Employment by Industry, Indiana Business Research Center, from URL http://www.stats.indiana.edu/bea/e.asi, accessed April 4, 2000, spreadsheet format

Table emp_full_parttm.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Z987 – Estimated individuals employed during 1987
Z988 – Estimated individuals employed during 1988
Z989 – Estimated individuals employed during 1989
Z990 – Estimated individuals employed during 1990
Z991 – Estimated individuals employed during 1991
Z992 – Estimated individuals employed during 1992
Z993 – Estimated individuals employed during 1993
Z994 – Estimated individuals employed during 1994
Z995 – Estimated individuals employed during 1995
Z996 – Estimated individuals employed during 1996
Z997 – Estimated individuals employed during 1997

Wu_withdrawals.shp

Source of values – U.S. Geological Survey

Table wu_withdrawals.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Pop – Estimated population, in thousands
Gw_with – Withdrawals of fresh ground water, in million gallons per day
Sw_with – Withdrawals of fresh surface water, in million gallons per day
Total_with – Sum of fresh ground- and surface-water withdrawals, in million gallons per day
Pc_with – Per capita withdrawals of fresh water, Total_with divided by Pop

Wu_withdrawals_cat.shp
Source of values – U.S. Geological Survey
Table wu_withdrawals_cat.dbf
Column Heading – Column Description
Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Pub_sup – Withdrawals of fresh water for public supply, in million gallons per day
Commercial – Withdrawals of fresh water for commercial uses, in million gallons per day
Domestic – Withdrawals of fresh water for domestic uses, in million gallons per day
Industrial – Withdrawals of fresh water for industrial uses, in million gallons per day
Therm_elect – Withdrawals of fresh water for thermoelectric power generation, in million gallons per day
Mining – Withdrawals of fresh water for mining uses, in million gallons per day
Livestock – Withdrawals of fresh water for livestock uses, in million gallons per day
Irrigation – Withdrawals of fresh water for irrigation uses, in million gallons per day
Total_with – Total withdrawal of fresh water, sum of all withdrawals, in million gallons per day

Wu_with_sw.shp
Source of values – U.S. Geological Survey
Table wu_with_sw.dbf
Column Heading – Column Description
Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Pub_sup – Withdrawals of fresh surface water for public supply, in million gallons per day
Commercial – Withdrawals of fresh surface water for commercial uses, in million gallons per day
Domestic – Withdrawals of fresh surface water for domestic uses, in million gallons per day
Industrical – Withdrawals of fresh surface water for industrial uses, in million gallons per day
Therm_elect – Withdrawals of fresh surface water for thermoelectric power generation, in million gallons per day
Mining – Withdrawals of fresh surface water for mining uses, in million gallons per day
Livestock – Withdrawals of fresh surface water for livestock uses, in million gallons per day
Irrigation – Withdrawals of fresh surface water for irrigation uses, in million gallons per day
Total_with – Total withdrawal of fresh surface water, sum of all withdrawals, in million gallons per day

Wu_with_gw.shp

Source of values – U.S. Geological Survey

Table wu_with_gw.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Pub_sup – Withdrawals of fresh ground water for public supply, in million gallons per day
Commercial – Withdrawals of fresh ground water for commercial uses, in million gallons per day
Domestic – Withdrawals of fresh ground water for domestic uses, in million gallons per day
Industrial – Withdrawals of fresh gound water for industrial uses, in million gallons per day
Therm_elect – Withdrawals of fresh ground water for thermoelectric power generation, in million gallons per day
Mining – Withdrawals of fresh ground water for mining uses, in million gallons per day
Livestock – Withdrawals of fresh ground water for livestock uses, in million gallons per day
Irrigation – Withdrawals of fresh ground water for irrigation uses, in million gallons per day
Total_with – Total withdrawal of fresh ground water, sum of all withdrawals, in million gallons per day

Wu_consump_use.shp

Source of values – U.S. Geological Survey

Table wu_consump_use.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Consum_use – Estimated quantity of water consumed, in million gallons per day
Conv_loss – Estimated quantity of water lost during movement from one location to another, in million gallons per day
Use_reclaimed – Estimated quantity of reclaimed wastewater, in million gallons per day

Wu_public_supply.shp
Source of values – U.S. Geological Survey
Table wu_public_supply.dbf
Column Heading – Column Description
Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Pop_srv_gw – Estimated population served by public suppliers getting water from ground-water sources
Pop_srv_sw – Estimated population served by public suppliers getting water from surface-water sources
Pop_srv_to – Estimated population served by public suppliers
With_gw – Withdrawals of fresh ground water for use by public suppliers, in million gallons per day
With_sw – Withdrawals of fresh surface water for use by public suppliers, in million gallons per day
With_tot – Total withdrawals of water for use by public suppliers, in million gallons per day
Dom_del – Estimated deliveries by public suppliers for domestic uses, in million gallons per day
Com_del – Estimated deliveries by public suppliers for commercial uses, in million gallons per day
Ind_del – Estimated deliveries by public suppliers for industrial uses, in million gallons per day
Thermo_del – Estimated deliveries by public suppliers for thermoelectric uses, in millions gallons per day
Total_del – Total deliveries by public suppliers, sum of all deliveries, in million gallons per day
Use_loss – Public use and loss, With_tot minus Total_del, in million gallons per day
Percap_use – Per capita use, Use_loss divided by Pop_srv_to, in gallons per day
Use_reclai – Reclaimed wastewater, in million gallons per day

Wu_domestic.shp

Source of values – U.S. Geological Survey

Table wu_domestic.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ss_gw_with – Self-supplied withdrawals of fresh ground water for domestic use, in million gallons per day
Ss_sw_with – Self-supplied withdrawals of fresh surface water for domestic use, in million gallons per day
Ss_tot_with – Total self-supplied withdrawals of water for domestic use, in million gallons per day
Ss_pop – Estimated population served by self-supplied withdrawals for domestic use
Ss_per_cap – Per capita withdrawals of water for domestic use, Ss_tot_with divided by Ss_pop, in gallons per day
Ps_del – Deliveries of water from public suppliers for domestic uses, in million gallons per day
Ps_pop – Estimated population served by public-supply deliveries of water for domestic use
Ps_per_cap – Per capita deliveries of water for domestic use, Ps_del divided by Ps_pop, in gallons per day
With_del – Sum of the self-supplied withdrawals and public-supply deliveries for domestic use
Consum_use – Consumptive use of water for domestic use

Wu_commercial.shp

Source of values – U.S. Geological Survey

Table wu_commercial.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ss_gw_with – Self-supplied withdrawals of fresh ground water for commercial use, in million gallons per day
Ss_sw_with – Self-supplied withdrawals of fresh surface water for commercial use, in million gallons per day
Ss_tot_with – Total self-supplied withdrawals of water for commercial use, in million gallons per day
Ps_del – Deliveries of water from public suppliers for commercial use, in million gallons per day
With_del – Sum of the self-supplied withdrawals and public-supply deliveries for commercial use
Consum_use – Consumptive use of water for commercial use
Use_reclai – Reclaimed wastewater, in million gallons per day

Wu_industrial.shp

Source of values – U.S. Geological Survey

Table wu_industrial.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ss_gw_with – Self-supplied withdrawals of fresh ground water for industrial use, in million gallons per day
Ss_sw_with – Self-supplied withdrawals of fresh surface water for industrial use, in million gallons per day
Ss_tot_with – Total self-supplied withdrawals of water for industrial use, in million gallons per day
Ps_del – Deliveries of water from public suppliers for industrial use, in million gallons per day
With_del – Sum of the self-supplied withdrawals and public-supply deliveries for industrial use
Consum_use – Consumptive use of water for industrial use
Use_reclai – Reclaimed wastewater, in million gallons per day

Wu_mining.shp

Source of values – U.S. Geological Survey

Table wu_mining.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Gw_with – Withdrawals of fresh ground water for mining use, in million gallons per day
Sw_with – Withdrawals of fresh surface water for mining use, in million gallons per day
Tot_with – Total withdrawals of water for mining use, in million gallons per day
Consum_use – Consumptive use of water for mining use
Use_reclai – Reclaimed wastewater, in million gallons per day

Wu_thermoelectric.shp
Source of values – U.S. Geological Survey
Table wu_thermoelectric.dbf
Column Heading – Column Description
Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ss_gw_with – Self-supplied withdrawals of fresh ground water for thermoelectric power generation, in million gallons per day
Ss_sw_with – Self-supplied withdrawals of fresh surface water for thermoelectric power generation, in million gallons per day
Ss_tot_with – Total self-supplied withdrawals of water for thermoelectric power generation, in million gallons per day
Ps_del – Deliveries of water from public suppliers for thermoelectric power generation, in million gallons per day.
With_del – Sum of the self-supplied withdrawals and public-supply deliveries for thermoelectric power generation
Consum_use – Consumptive use of water for thermoelectric power generation
Use_reclai – Reclaimed wastewater, in million gallons per day
Power_gen – Estimated power generated, in million kilowatt hours

Wu_ff_thermoelectric.shp
Source of values – U.S. Geological Survey
Table wu_ff_thermoelectric.dbf
Column Heading – Column Description
Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ss_gw_with – Self-supplied withdrawals of fresh ground water for fossil-fuel thermoelectric power generation, in million gallons per day
Ss_sw_with – Self-supplied withdrawals of fresh surface water for fossil-fuel thermoelectric power generation, in million gallons per day
Ss_tot_with – Total self-supplied withdrawals of water for fossil-fuel thermoelectric power generation, in million gallons per day
Ps_del – Deliveries of water from public suppliers for fossil-fuel thermoelectric power generation, in million gallons per day
With_del – Sum of the self-supplied withdrawals and public-supply deliveries for fossil-fuel thermoelectric power generation
Consum_use – Consumptive use of water for fossil-fuel thermoelectric power generation
Use_reclai – Reclaimed wastewater
Power_gen – Estimated power generated, in million kilowatt hours

Wu_hydroelectric.shp

Source of values – U.S. Geological Survey

Table wu_hydroelectric.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Instr_mgd – Quantity of instream water used by hydroelectric facilities to generate power, in million gallons per day
Instr_acft – Quantity of instream water used by hydroelectric facilities to generate power, in thousands of acre-feet per year
Offstr_with – Withdrawals of fresh surface water used by hydroelectric facilities to generate power, in million gallons per day
Instr_pwge – Power generated using instream water, in million kilowatt hours
Ofstr_pwgen – Power generated using withdrawals of fresh surface water, in million kilowatt hours
Pwr_gen – Total power generated using instream and offstream withdrawals, in million kilowatt hours
Instoffstr – Total water used by hydroelectric facilities, sum Instr_mgd and Offstr_with, in million gallons per day

Wu_irrigation.shp

Source of values – U.S. Geological Survey

Table wu_irrigation.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles miles
Gw_with – Withdrawals of fresh ground water for irrigation, in million gallons per day
Sw_with – Withdrawals of fresh surface water for irrigation, in million gallons per day
Tot_with – Total withdrawals of water for irrigation, in million gallons per day
Consum_use – Consumptive use of water for mining use, in million gallons per day
Conv_loss – Estimated quantity of water lost during movement from one location to another, in million gallons per day
Use_reclai – Reclaimed wastewater, in million gallons per day
Sprinkler – Estimated area irrigated by sprinklers, in thousands of acres
Micro – Estimated area irrigated by micro-irrigation methods, in thousands of acres
Surface – Estimated area irrigated by surface-irrigation methods, in thousands of acres
Tot_acres – Estimated area irrigated, in acres

Wu_livestock.shp

Source of values – U.S. Geological Survey

Table wu_livestock.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ls_gw_with – Withdrawal of fresh ground water for stock and animal specialty uses, in million gallons per day
Ls_sw_with – Withdrawal of fresh surface water for stock and animal specialty uses, in million gallons per day
Ls_tot_with – Withdrawal of fresh water for stock and animal specialty uses, in million gallons per day
Sk_gw_with – Withdrawal of fresh ground water for stock uses, in million gallons per day
Sk_sw_with – Withdrawal of fresh surface water for stock uses, in million gallons per day
Sk_tot_with – Withdrawal of fresh water for stock uses, in million gallons per day
As_gw_with – Withdrawal of fresh ground water for animal specialty uses, in million gallons per day
As_sw_with – Withdrawal of fresh surface water for animal specialty uses, in million gallons per day
As_tot_with – Withdrawal of fresh water for animal specialty uses, in million gallons per day
Ls_consuse – Consumptive use of water for stock and animal specialty uses, in million gallons per day
Sk_consuse – Consumptive use of water for stock uses, in million gallons per day
As_consuse – Consumptive use of water for animal specialty uses, in million gallons per day

Wu_wwtp.shp

Source of values – U.S. Geological Survey

Table wu_wwtp.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Pub_fac – Number of public wastewater-treatment plants
Other_fac – Number of nonpublic wastewater treatment plants
Total_fac – Total number of wastewater treatment plants
Rtn_wtr – Public releases of water from wastewater-treatment plants, in million gallons per day

Cerlis.shp

Source of values – Written Communication, November 1996, Douglas Campbell, Indiana Department of Environmental Management

Table cerlis.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Cerlist – Number of cerlis sites

ww.shp

Table ww.dbf

Column Heading – Column Description

Area – Area of the feature, in map units (square meters)
Perimeter – Perimeter of the feature, in map units (meters)
Statecty – Concatenation of the Federal Information Processing State and County Codes
St – Two-letter Postal abbreviation for the state
In_cty – Two-digit State of Indiana County Code
Cntyname – Name of the county
Acres – Area of the feature, in acres
Sq_miles – Area of the feature, in square miles
Ww_sup – Number of noncommunity ground-water suppliers