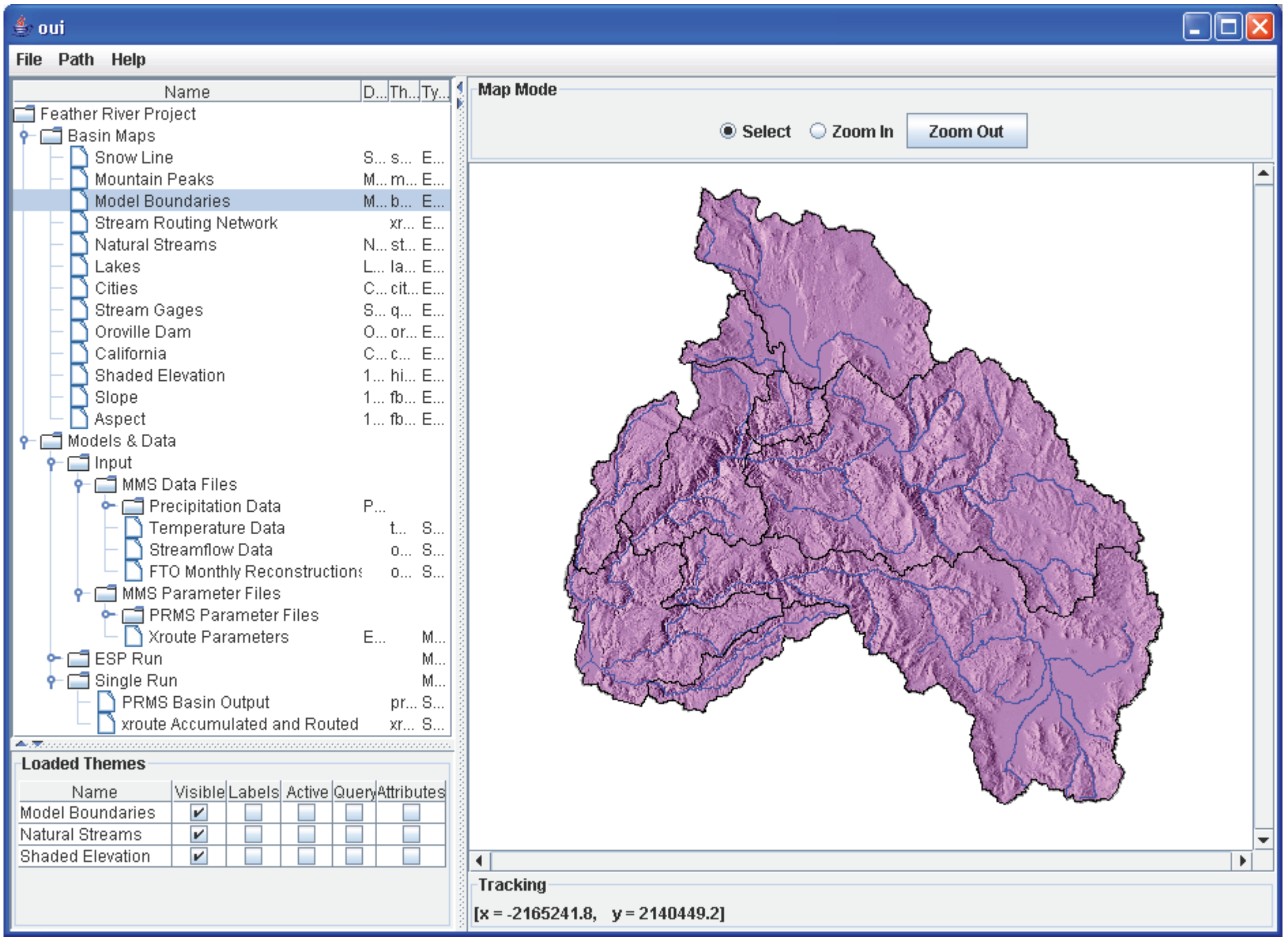
User’s Manual for the Object User Interface (OUI): An Environmental Resource Modeling Framework



Name	D...	Th...	Ty...
Feather River Project			
Basin Maps			
Snow Line	S...	s...	E...
Mountain Peaks	M...	m...	E...
Model Boundaries	M...	b...	E...
Stream Routing Network		xr...	E...
Natural Streams	N...	st...	E...
Lakes	L...	la...	E...
Cities	C...	cit...	E...
Stream Gages	S...	q...	E...
Oroville Dam	O...	or...	E...
California	C...	c...	E...
Shaded Elevation	1...	hi...	E...
Slope	1...	fb...	E...
Aspect	1...	fb...	E...
Models & Data			
Input			
MMS Data Files			
Precipitation Data	P...		
Temperature Data		t...	S...
Streamflow Data		o...	S...
FTO Monthly Reconstructions		o...	S...
MMS Parameter Files			
PRMS Parameter Files			
Xroute Parameters	E...		M...
ESP Run			M...
Single Run			M...
PRMS Basin Output		pr...	S...
xroute Accumulated and Routed		xr...	S...

Loaded Themes

Name	Visible	Labels	Active	Query	Attributes
Model Boundaries	☑	□	□	□	□
Natural Streams	☑	□	□	□	□
Shaded Elevation	☑	□	□	□	□

[x = -2165241.8,   y = 2140449.2]

Open-File Report 2008–1120

U.S. Department of the Interior
U.S. Geological Survey

User’s Manual for the Object User Interface (OUI): An Environmental Resource Modeling Framework

By Steven L. Markstrom and Kathryn M. Koczot

Open-File Report 2008–1120

U.S. Department of the Interior
U.S. Geological Survey

U.S. Department of the Interior
DIRK KEMPTHORNE, Secretary

U.S. Geological Survey
Mark D. Myers, Director

U.S. Geological Survey, Reston, Virginia: 2008

For product and ordering information:
World Wide Web: http://www.usgs.gov/pubprod
Telephone: 1-888-ASK-USGS

For more information on the USGS—the Federal source for science about the Earth, its natural and living resources, natural hazards, and the environment:
World Wide Web: http://www.usgs.gov
Telephone: 1-888-ASK-USGS

Suggested citation:

Markstrom, S.L., and Koczot, K.M., 2008, User's manual for the object user interface (OUI)—An environmental resource modeling framework: U.S. Geological Survey Open-File Report 2008–1120, 39 p.

Any use of trade, product, or firm names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

PREFACE

This report describes the U.S. Geological Survey Object User Interface (OUI). The performance of the program has been tested in a variety of applications. Future applications, however, might reveal errors that were not detected in the test simulations. Users are requested to send notification of any errors found in this report or the model program to:

Office of Surface Water
U.S. Geological Survey
411 National Center
Reston, VA 20192
(703) 648–5001

The latest version of the model program and this report can be obtained using the Internet at address below (accessed March, 2008):

http://water.usgs.gov/lookup/get?crresearch/oui

Contents

Figures

Tables

User’s Manual for the Object User Interface (OUI): An Environmental Resource Modeling Framework

By Steven L. Markstrom and Kathryn M. Koczot

Abstract

The Object User Interface is a computer application that provides a framework for coupling environmental-resource models and for managing associated temporal and spatial data. The Object User Interface is designed to be easily extensible to incorporate models and data interfaces defined by the user. Additionally, the Object User Interface is highly configurable through the use of a user-modifiable, text-based control file that is written in the eXtensible Markup Language. The Object User Interface user’s manual provides (1) installation instructions, (2) an overview of the graphical user interface, (3) a description of the software tools, (4) a project example, and (5) specifications for user configuration and extension.

Introduction

The Object User Interface (OUI) is a map-based framework for models and modeling data. It provides a common interface for running models, as well as for acquiring, browsing, organizing, displaying, and selecting spatial and temporal data. As environmental-resource modeling has evolved, data availability and requirements, as well as the increased sophistication of algorithms that simulate the physical processes, have resulted in much more complex model setups. Organization of modeling projects can influence the success of these projects as much as any other factor. In addition, the necessity of effective and timely communication of simulation results has a great effect on the success of modeling projects. OUI was developed by the United States (U.S.) Geological Survey as a tool for addressing these issues.

Approach

Although OUI may look like a graphical user interface (GUI) for a geographical information system (GIS), there are several important distinctions:

- OUI provides a generic temporal and spatial interface for modeling projects.
- OUI provides a mechanism for deploying a modeling project to clients.
- OUI may be configured, extended, and adapted to many different needs.

OUI is written in the Java programming language and is configured with a control file that is written in the eXtensible Markup Language (XML). This language allows users a high degree of flexibility in hardware and user interface configuration. OUI can generate map displays of GIS feature data in line, point, or polygon shapefiles or as American Standard Code for Information Interchange (ASCII) raster formats. OUI also can display the contents of Modular Modeling System (MMS) format data files as they relate to the features in these maps (Leavesley and others, 1996a and 1996b). Tools developed for use in OUI run either as stand-alone applications or as an integrated system. This flexibility allows the user to configure OUI for specific modeling projects. Configuration and extension of OUI is possible without modifying the source code of the basic OUI system.

OUI can be integrated with other programming languages, most notably C and Fortran, on either the system or process level. Software compiled into a platform-specific executable program (referred to as native programming language, or native code) can be called by OUI through a system call. In this way native codes can continue to be used with minimal modification. Additionally, OUI also can invoke native code by using the Java Native Interface. In this case, the software compiled into a dynamically loaded library can be invoked as needed by OUI during execution.

Description of an OUI Project

An OUI project is a well-defined set of data files, executable programs, simulation models, and information about them, which are described by a configuration file written in XML. Although the actual name of this configuration file is arbitrary, it is subsequently referred to as *project.xml* in this report.

Project Tree and Tree Nodes

The *project.xml* file contains a structure called the project tree, which defines how the data files, executable programs, and simulation models are organized within the OUI project. The project is organized by adding blocks of code called tree nodes to the project tree. Each tree node associates a specific software action with the data and models in the project. This process is described in detail in the "OUI Project Configuration" section. The project tree is hierarchical, and a top-level project tree node must be defined for the entire project. All other tree nodes are contained within the project tree node. Any tree node may contain other tree nodes. This structure is similar to a file system on a computer, in which the root directory contains all of the other directories and files that are organized to meet the needs of the user. The project tree is displayed in the OUI upon startup.

Each tree node has an associated Java class that defines how the tree node will behave within OUI. For example, a tree node Java class can provide the functionality to run a simulation model. `OuiTreeNode` is the most basic tree node Java class. Any subclass of `OuiTreeNode` can be used as a tree node Java class, as described in the "OUI Project Configuration" and in the "OUI Project Extension" sections.

Data

Tree nodes may need data to accomplish their function. For example, when given a theme and time-series data, the tree node Java class `MMSDataTreeNode` will allow the user to select a feature from the theme and plot the associated time-series data for this feature. This type of class is called a data-management interface (DMI) and provides OUI with functionality to operate on these data types. The shapefile for the theme, the file that contains the time-series data, and the information which relates the features in the theme to the data in the time-series file is specified as metadata in the *project.xml* file. Specific information about how metadata relates to tree nodes is specified in the "OUI Project Configuration" and "OUI Project Extension" sections. Themes, time series, and parameters are basic types of data used by DMIs to support environmental-resource modeling and visualization.

A theme is a geographical dataset that may be loaded into OUI, either as ASCII raster data or as vector shapefiles. The shapefiles contain unique feature data, stored as points, lines, or polygons. All shapefiles have GIS attributes that describe the properties of the features. Themes can be displayed to provide geographical reference for an OUI project and allow selection of shapefile features. Selected features can then be used as input for simulation models or for data visualization. For display purposes, all geographical datasets in an OUI project must be in the same projection.

A time-series dataset is an ordered sequence of values at successive time increments. These values are values of state or flux, which vary through time at specific geographical locations. Time-series data can be either physical measurements or model-simulated values. Time-series data are displayed by OUI for selected features from a theme, or used as input for simulation models. The OUI project example documented in this report uses the MMS time-series file format for storage of time-series data (Leavesley and others, 1996b).

A parameter is a value that remains constant during a run of a simulation model. These values are usually physical or empirical constants, GIS attribute feature properties (such as vegetation type), or values derived from GIS attributes. Parameter sets can be displayed and edited by OUI for selected themes or used as input for simulation models.

Executable Programs and Simulation Models

Any simulation model written in the Java programming language, or which can be run without a graphical user interface from the command line, can be run within an OUI project. A model-management interface (MMI) is a tree node Java class that provides the functionality to run the simulation model. Time series, parameters, and selected themes and features are all data that are available to the MMI tree node as input to the simulation model. After the simulation model is finished with execution, any output data produced are available to DMI tree nodes for visualization, analysis, or export. Thus, the OUI project can be configured to provide the user with the necessary support for pre- and post-processing of model data, as well as control of the execution of the simulation model. The MMIs available for OUI to run simulation models are described in the "OUI Project Configuration" section.

Purpose and Scope

This report describes OUI, version 1.0. The report first describes how to install OUI and its hardware and software requirements, the path and file structure of the OUI system, and how to initiate OUI execution. An overview of the various graphical windows is presented in the section "Graphical User Interface Overview." A discussion of the integrated OUI software tools for modeling and data analysis are presented in the section "Software Tools." An annotated modeling project example is presented in the section "OUI Project Example: Feather River Basin." The final two sections, "OUI Project Configuration" and "OUI Project Extension," show users how to customize OUI for specific modeling applications.

Three levels of OUI users are considered in this manual: (1) managers who will run preconfigured models; (2) modelers who will configure OUI-based modeling projects; and (3) developers who will extend OUI functionality by writing new Java source code. As one moves from the first through the third levels, an increasing amount of knowledge is required about the structure and concepts of OUI, XML, and the Java programming language. This manual is organized in the same fashion. While all users will gain useful information by reading the entire document, the "OUI Project Configuration" section of this report is written specifically for modelers, and the "OUI Project Extension" section is written specifically for developers.

Acknowledgments

OUI was developed through the U.S. Geological Survey Watershed and River System Management Program and National Research Program. The Feather River OUI example was constructed in cooperation with the California Department of Water Resources Cooperative Snow Surveys Program. The authors thank George Leavesley, Roland Viger, and R. Steven Regan (U.S. Geological Survey National Research Program, Lakewood, Colorado), Douglas Boyle and Gregg Lamorey (Desert Research Institute, Reno, Nevada), Harry Lins (U.S. Geological Survey Office of Surface Water, Reston, Virginia), Frank Gehrke and Michael Mierzwa (California Department of Water Resources, Sacramento, California), and Donald Frevert and David King (U.S. Bureau of Reclamation, Denver, Colorado) for their support, testing, and encouragement throughout the development of OUI.

Installation Instructions

The latest version of OUI is available from the USGS OUI web page (*http://water.usgs.gov/lookup/get?crresearch/oui*, accessed March, 2008) and is distributed as a zip file. About 190 megabytes of disk space is required for installation. This total includes approximately 30 megabytes for OUI libraries and documentation and another 160 megabytes for the Feather River project example. Of the 160 megabytes, about 100 megabytes is required to hold output data from the example model runs.

OUI is installed into a top-level directory named *oui/* (fig. 1). Four subdirectories *doc/, lib/, src/* and *feather/* are located within the *oui/* directory. The directory *doc/* contains a copy of this report and the javadoc documentation of the OUI source code; *lib/* contains the libraries necessary to run OUI; *src/* contains the source code used to compile OUI; and *feather/* contains the files for the Feather River example project.

Hardware Requirements

OUI is developed for Windows- and Linux-based personal computers. OUI installation on other computer systems can require platform-dependent modifications. Minimum system requirements include Windows 2000, Windows XP, or the Linux operating system; 800 megahertz Intel Pentium III processor (or equivalent); 256 megabytes random access memory; display capabilities greater than 256 color depth; and 1 gigabyte of hard disk space. However, system requirements depend on the size of the data sets being used. For example, the Feather River OUI project example was developed on a Windows XP personal computer with an AMD 3000+ Athlon processor, 2 gigabytes of random-access memory, and 60 gigabytes of hard disk space. OUI performance is more dependent on data access speed than on numerical processing speed; thus, it is comparable to other programs that handle large data sets, such as relational data bases or GIS systems.

Software Requirements

OUI requires Java version 1.6 or later. Optionally, OUI extensions may require (1) Java3D; (2) MMS to run models built with MMS (Leavesley and others, 2005); and (3) a GIS system that produces shapefiles to create new spatial data sets.

Minimally, to run OUI, the Java Runtime Environment must be installed. Developers requiring a Java compiler need to install the Java Development Kit. Java is usually installed as a shared-system resource, which may require system administrator privileges. Check with your system administrator before attempting to install Java. Java is freely available from the Sun Microsystems web page (*http://java.sun.com*, accessed March, 2008).

The Java3D library is required to display three-dimensional animations of model results; however, if this option is not desired, Java3D does not need to be installed. Java3D is freely available from the Java3D web page (*https://java3d.dev.java.net*, accessed March, 2008).

MMS is a modeling framework for building and running simulation models. It is freely available from the USGS MMS web page (*http://water.usgs.gov/lookup/get?crresearch/mms*, accessed March, 2008).

OUI Execution

Start files are provided in the OUI project directory *oui/feather/*. These files specify the input arguments to Java. These arguments may need to be edited to reflect the specific paths of a particular project configuration. Generally, these files will not need to be changed. A batch file (*oui.bat*) is provided with the example project folder. Windows users should run this batch file to start OUI. A Borne shell script file (*oui.sh*) is provided with the example project directory. Linux users should run this script to start OUI.

OUI Path Specification

Paths may be specified as either relative or absolute. There are two locations in an OUI project where paths are specified—the OUI start files (*oui.sh* and *oui.bat*) and in the *project.xml* file. All other path specifications are relative to these. If a project directory is moved to a new location, and absolute paths are used, these paths must be edited. The path specifications in the Feather River example project are relative and should not be edited.

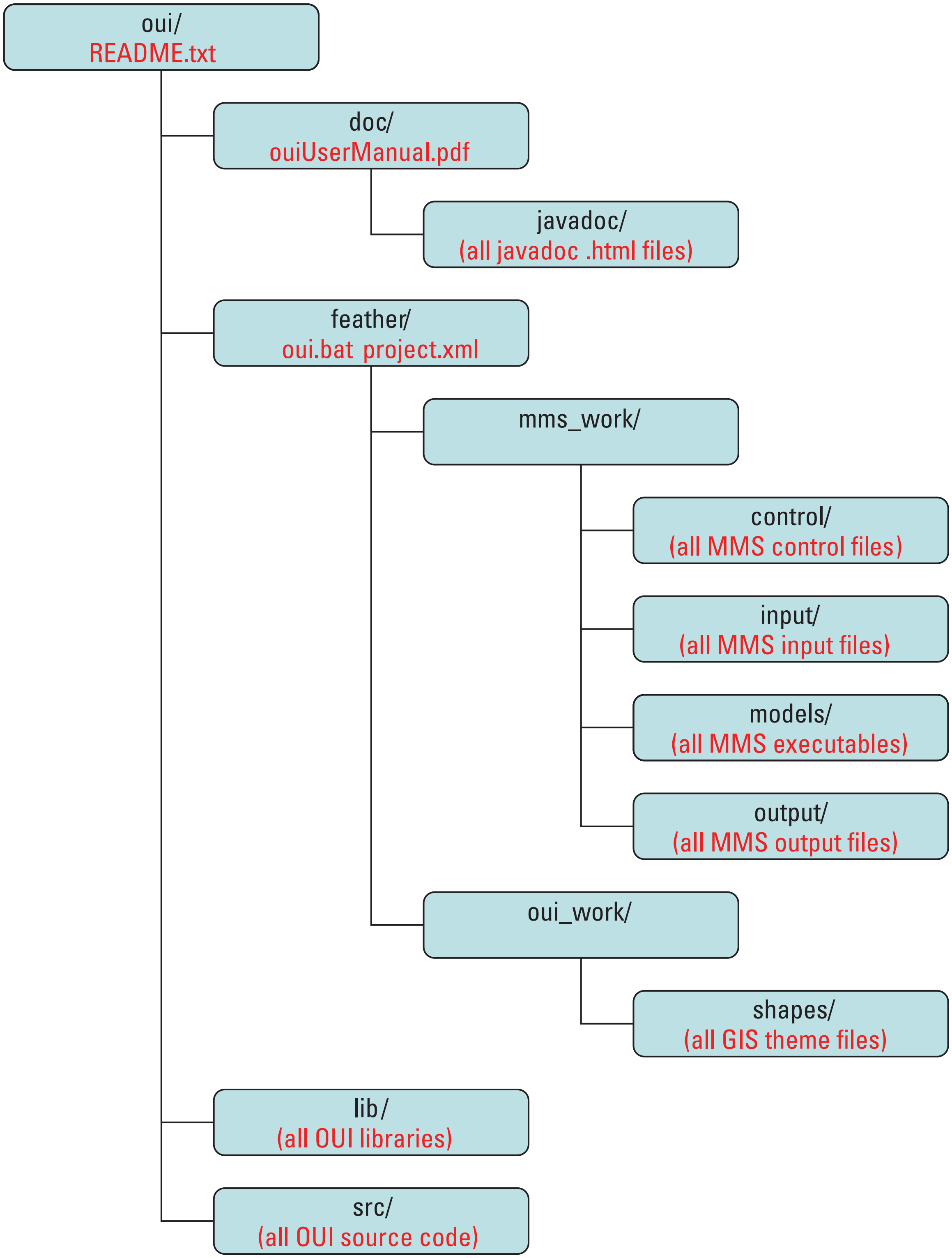


Figure 1. OUI project directory structure with directories shown in black and files shown in red.

Graphical User Interface Overview

Upon startup of OUI, the top-level window will appear (fig. 2). The *OUI* window includes minimize, maximize, and close buttons, which are located in the upper right-hand corner. This window has four main parts: (1) the menu bar, (2) the *Project Tree* window, (3) the *Map* window, and (4) the *Loaded Themes* window. These are described below.

Menu

The menu bar provides three top-level menus. These are *File*, *Path*, and *Help*. These menus are described below.

The *File* menu has two options, *Load Project* and *Quit*. *Load Project* activates a file selector window. The user can select a *project.xml* file to load a new project. The *Quit* option will quit OUI. This option also will terminate all processes initiated by this OUI session.

The *Path* menu has one option. *Set Path* activates the *OUI Set Paths* window (fig. 3). The user can modify, add, and delete the project directory paths that are specified in the *project.xml* file. The *Add* button is located at the bottom of the menu. Directory paths can be deleted by highlighting and erasing the contents of *Tag* and *Path*. The *project.xml* file is immediately updated with any new path information.

The *Help* menu has two options. The *OUI User Manual* option displays an online version of this report (fig. 4). The online manual provides an index and access to the report sections, through the table of contents or search by keywords. The *About* option describes the version of OUI, the version of Java used to compile OUI, and the uniform resource locater to the OUI webpage.

Project Tree Window

Below the menu bar in the upper left portion of the top-level window, the *Project Tree* window displays the OUI project tree nodes, as defined in the *project.xml* file, in a hierarchical structure (fig. 2). Tree nodes in the *Project Tree* window can be expanded or collapsed, to either display or hide their children, by double-clicking on the tree node (fig. 5). Each tree node in the *Project Tree* window has one parent and may or may not have children. This structure allows for navigating the *Project Tree* window at any level of detail.

Clicking on a tree node with the right mouse button will execute the associated Java class for this tree node. If the tree node contains a theme, it can be displayed by selecting *Load* from the pop-up menu. Tree nodes that contain MMIs or DMIs can be run by selecting *Run* from the pop-up menu.

Map Window

When a theme has been selected and loaded, the theme is displayed in the *Map* window on the right-hand side of the GUI. The *Map* window (figs. 2 and 6) has three themes selected and loaded from the *Project Tree* window. In this case, the *Model Boundaries*, *Natural Streams*, and *Shaded Elevation* themes for the Feather River project are displayed as overlays in the *Map* window.

The *Map Mode* buttons (figs. 2 and 6) determine the action associated with a left mouse button click on the *Map* window. The state of the *Select* and *Zoom In* buttons determines the *Map Mode*. If the *Map Mode* is set to *Select*, features of the active theme (see "*Loaded Themes* Window" section) displayed in the *Map* window are selectable. If the *Map Mode* is set to *Zoom In*, a left mouse click on the *Map* window will result in a 50 percent zoom in on the map, centered on the location of the mouse click. Only one of these buttons can be activated (clicked on) at a time. The *Zoom Out* button will zoom out the contents of the *Map* window 50 percent, regardless of whether the *Map* window is in *Select Mode* or *Zoom Mode*. The scroll bars on the right and bottom sides of the *Map* window will pan the viewable portion of the *Map* window.

The *Tracking* box on the bottom of the *Map* window (figs. 2 and 6) displays world coordinates of the projection of the GIS data at the cursor position when the cursor is in the *Map* window. Also, if there is an active theme (see "*Loaded Themes* Window" section), the *Tracking* box will identify the features of that theme by name.

To select features of the active theme (see "*Loaded Themes* Window" section), in the *Map* window, move the cursor over the feature until the name of that feature appears in the *Tracking* box. Select the feature with a click of the left mouse button. The selected feature will highlight in yellow (fig. 7). Further, if the active theme's *GIS Attribute Tool* window (see "*GIS Attribute Tool*" section) is open, the record corresponding to the feature selected in the *Map* window will be highlighted in this table in blue.

Loaded Themes Window

All themes, either raster or vector, that have been loaded to the *Map* window will be listed in the *Loaded Themes* window (fig. 2). When a theme is loaded, the map data, as well as any GIS attribute information associated with the mapped features, is available for display and selection. Once a theme is loaded, it cannot be loaded again.

As new themes are loaded, they appear at the top of the *Loaded Themes* list. This list also is the layer order in which the themes appear in the *Map* window. Raster themes are opaque and will obscure the themes below. It is a good idea to move raster themes to the bottom of the *Loaded Themes* list. To change theme order, click by using the right mouse button on a theme name. The *Theme Order* menu will appear (fig. 8). Options on this menu are as follows:

- The *Remove* menu item will unload the theme from the *Map* window and the *Loaded Themes* list.

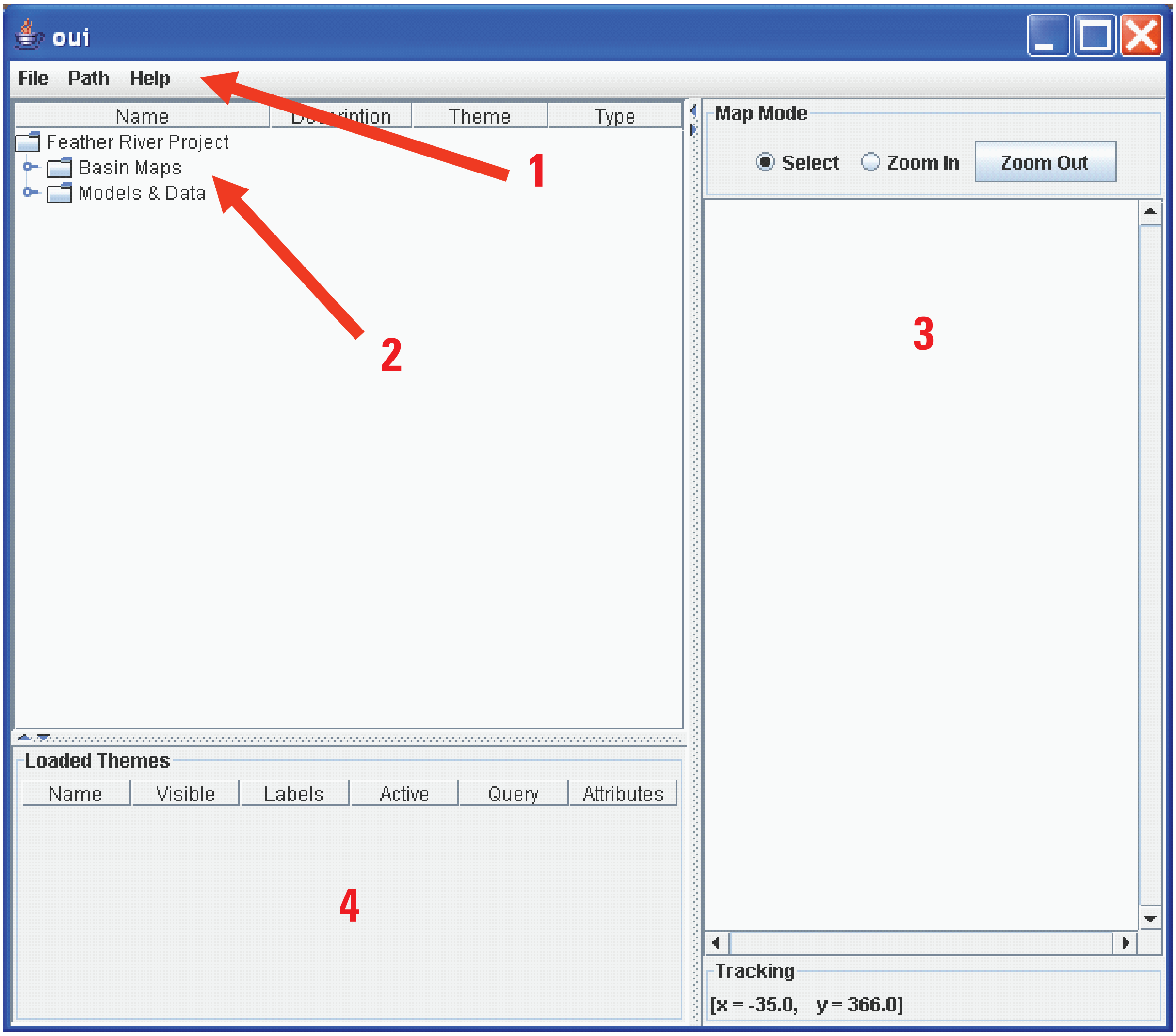
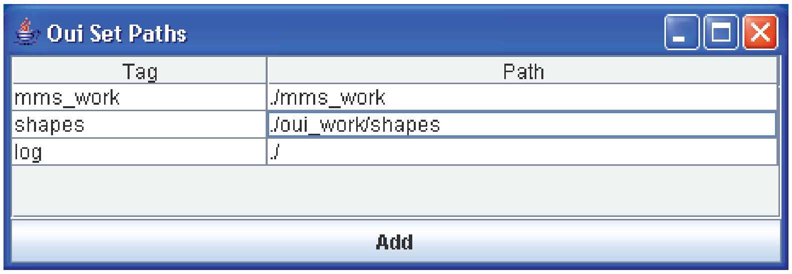
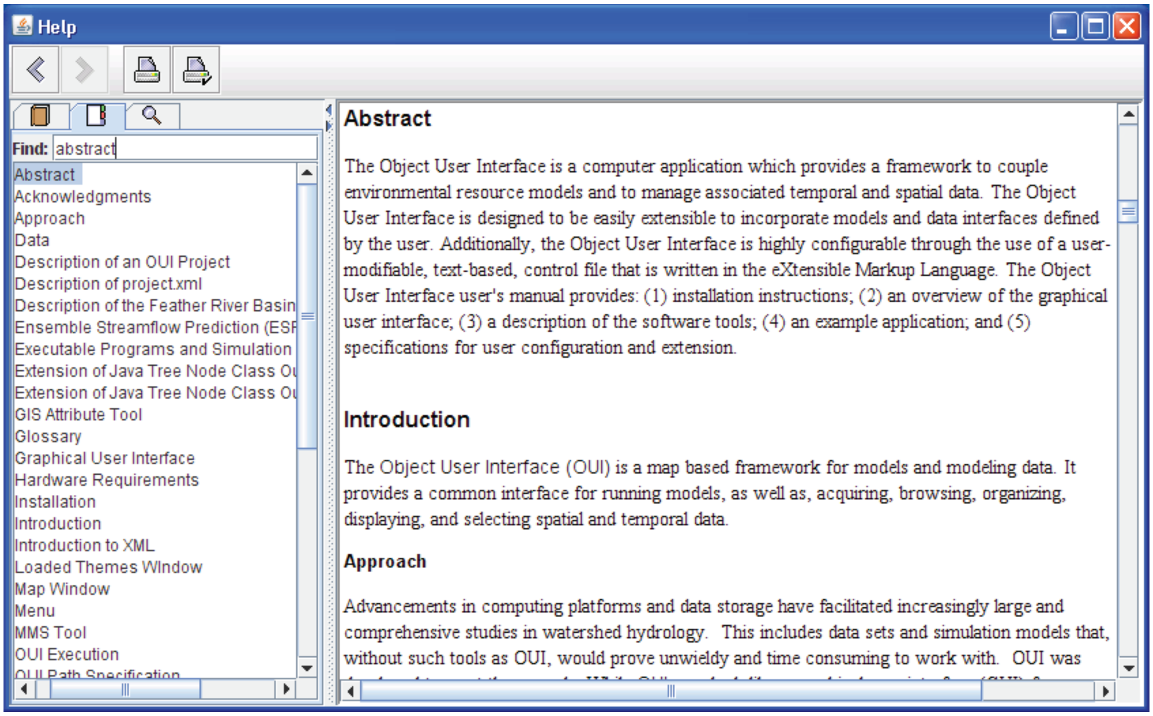


Figure 2. OUI graphical user interface window upon startup showing (1) the menu bar, (2) the *Project Tree* window, (3) the *Map window*, and (4) the *Loaded Themes* window.



Tag	Path
mms_work	./mms_work
shapes	./oui_work/shapes
log	./

Figure 3. *OUI Set Paths* window showing paths that are set for the current project.



Abstract
Acknowledgments
Approach
Data
Description of an OUI Project
Description of project.xml
Description of the Feather River Basin
Ensemble Streamflow Prediction (ESP
Executable Programs and Simulation
Extension of Java Tree Node Class Ou
Extension of Java Tree Node Class Ou
GIS Attribute Tool
Glossary
Graphical User Interface
Hardware Requirements
Installation
Introduction
Introduction to XML
Loaded Themes WIndow
Map Window
Menu
MMS Tool
OUI Execution
OUI Path Specification

Abstract

The Object User Interface is a computer application which provides a framework to couple environmental resource models and to manage associated temporal and spatial data. The Object User Interface is designed to be easily extensible to incorporate models and data interfaces defined by the user. Additionally, the Object User Interface is highly configurable through the use of a user-modifiable, text-based, control file that is written in the eXtensible Markup Language. The Object User Interface user's manual provides: (1) installation instructions; (2) an overview of the graphical user interface; (3) a description of the software tools; (4) an example application; and (5) specifications for user configuration and extension.

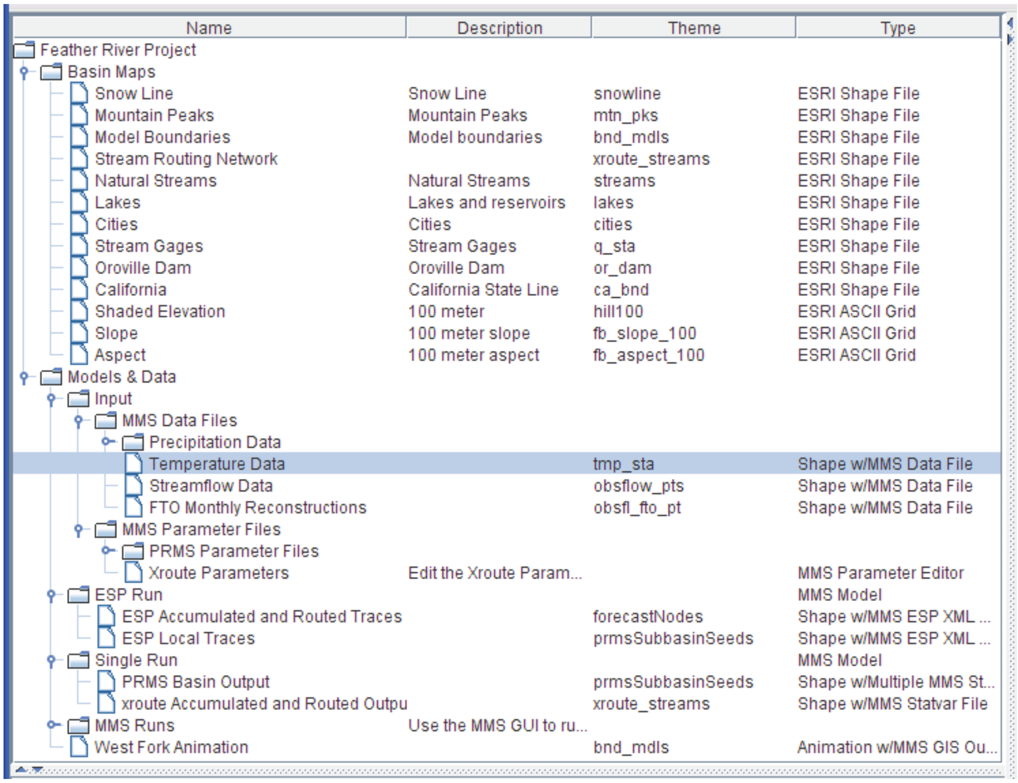
Introduction

The Object User Interface (OUI) is a map based framework for models and modeling data. It provides a common interface for running models, as well as, acquiring, browsing, organizing, displaying, and selecting spatial and temporal data.

Approach

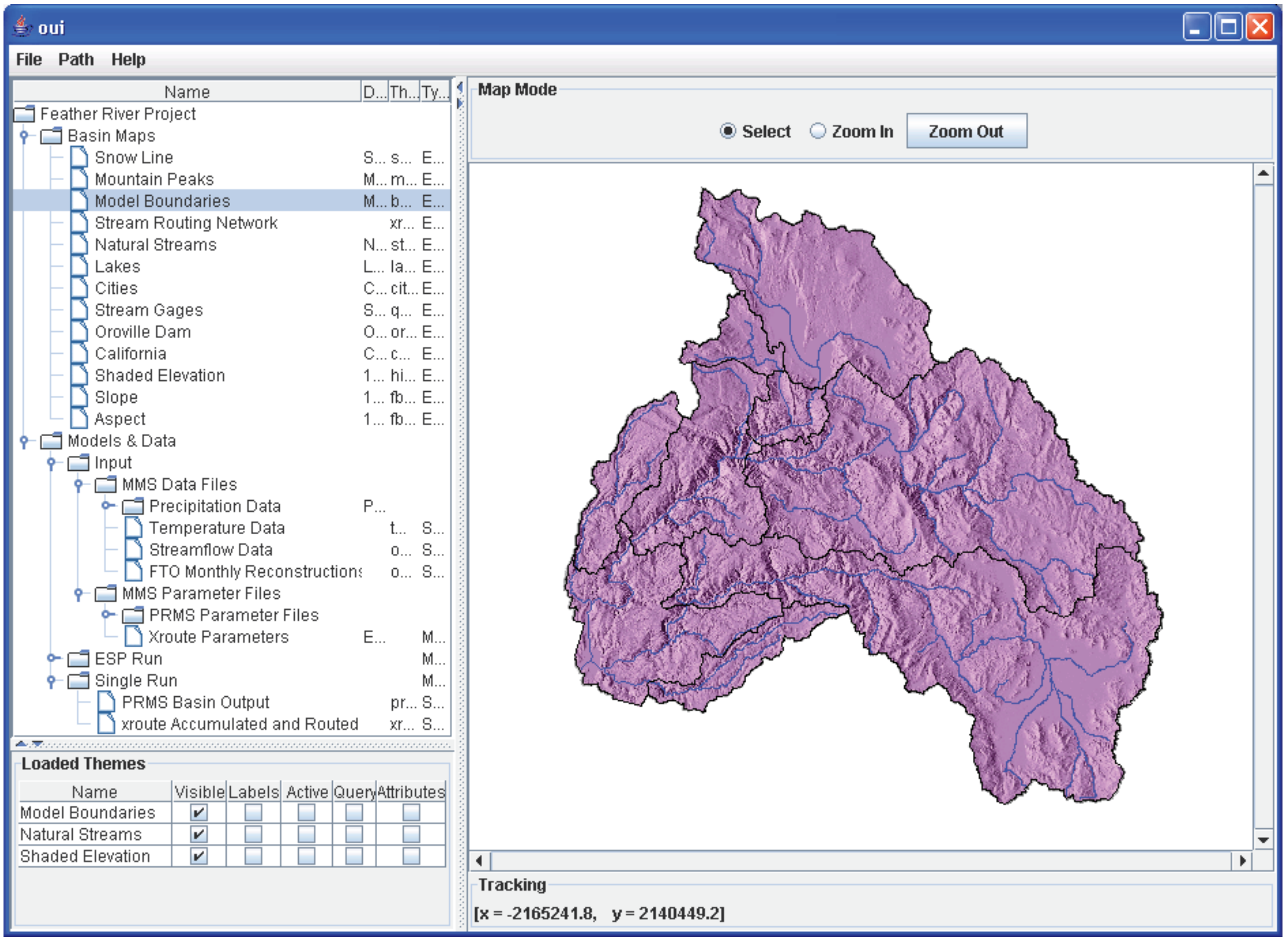
Advancements in computing platforms and data storage have facilitated increasingly large and comprehensive studies in watershed hydrology. This includes data sets and simulation models that, without such tools as OUI, would prove unwieldy and time consuming to work with. OUI was

Figure 4. *OUI Help* system window.



Name	Description	Theme	Type
Feather River Project			
Basin Maps			
Snow Line	Snow Line	snowline	ESRI Shape File
Mountain Peaks	Mountain Peaks	mtn_pks	ESRI Shape File
Model Boundaries	Model boundaries	bnd_mdls	ESRI Shape File
Stream Routing Network		xroute_streams	ESRI Shape File
Natural Streams	Natural Streams	streams	ESRI Shape File
Lakes	Lakes and reservoirs	lakes	ESRI Shape File
Cities	Cities	cities	ESRI Shape File
Stream Gages	Stream Gages	q_sta	ESRI Shape File
Oroville Dam	Oroville Dam	or_dam	ESRI Shape File
California	California State Line	ca_bnd	ESRI Shape File
Shaded Elevation	100 meter	hill100	ESRI ASCII Grid
Slope	100 meter slope	fb_slope_100	ESRI ASCII Grid
Aspect	100 meter aspect	fb_aspect_100	ESRI ASCII Grid
Models & Data			
Input			
MMS Data Files			
Precipitation Data			
Temperature Data		tmp_sta	Shape w/MMS Data File
Streamflow Data		obsflow_pts	Shape w/MMS Data File
FTO Monthly Reconstructions		obsfl_fto_pt	Shape w/MMS Data File
MMS Parameter Files			
PRMS Parameter Files			
Xroute Parameters	Edit the Xroute Param...		MMS Parameter Editor
ESP Run			MMS Model
ESP Accumulated and Routed Traces		forecastNodes	Shape w/MMS ESP XML ...
ESP Local Traces		prmsSubbasinSeeds	Shape w/MMS ESP XML ...
Single Run			MMS Model
PRMS Basin Output		prmsSubbasinSeeds	Shape w/Multiple MMS St...
xroute Accumulated and Routed Outpu		xroute_streams	Shape w/MMS Statvar File
MMS Runs	Use the MMS GUI to ru...		
West Fork Animation		bnd_mdls	Animation w/MMS GIS Ou...

Figure 5. *Project Tree* window expanded to show models and data for the project.



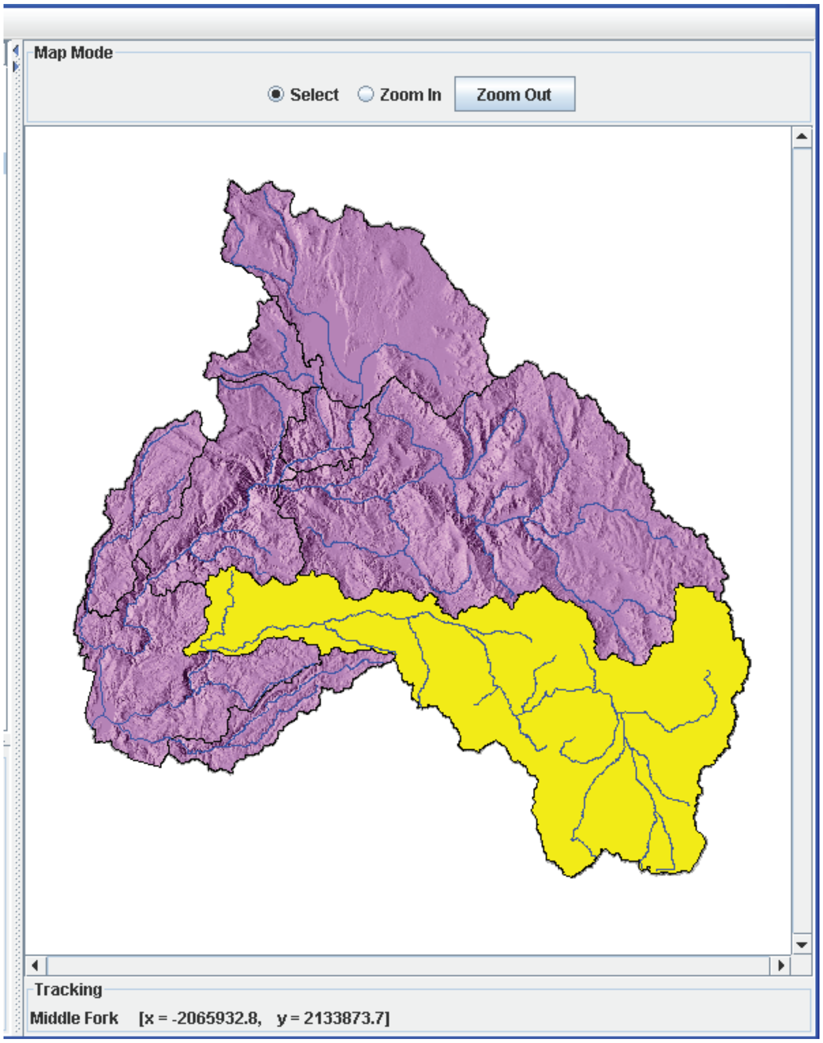
```
Name                                   D... Th... Ty...
Feather River Project
  Basin Maps
    Snow Line                          S... s...  E...
    Mountain Peaks                     M... m...  E...
    Model Boundaries                   M... b...  E...
    Stream Routing Network                  xr... E...
    Natural Streams                    N... st... E...
    Lakes                              L... la... E...
    Cities                             C... cit...E...
    Stream Gages                       S... q...  E...
    Oroville Dam                       O... or... E...
    California                         C... c...  E...
    Shaded Elevation                   1... hi... E...
    Slope                              1... fb... E...
    Aspect                             1... fb... E...
  Models & Data
    Input
      MMS Data Files
        Precipitation Data             P...
        Temperature Data                    t...  S...
        Streamflow Data                     o...  S...
        FTO Monthly Reconstructions         o...  S...
      MMS Parameter Files
        PRMS Parameter Files
        Xroute Parameters              E...       M...
    ESP Run                                       M...
    Single Run                                    M...
      PRMS Basin Output                     pr... S...
      xroute Accumulated and Routed         xr... S...
```

Loaded Themes

Name	Visible	Labels	Active	Query	Attributes
Model Boundaries	☑	□	□	□	□
Natural Streams	☑	□	□	□	□
Shaded Elevation	☑	□	□	□	□

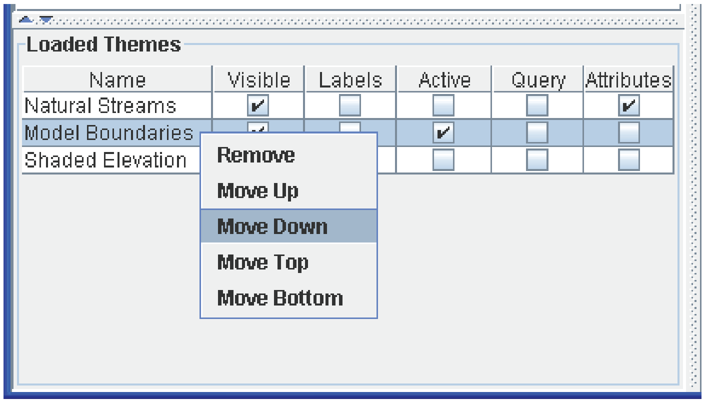
[x = -2165241.8, y = 2140449.2]

Figure 6. *OUI Map* window with three themes loaded.



Middle Fork [x = -2065932.8, y = 2133873.7]

Figure 7. *OUI Map* window with a feature selected (yellow) from the *Model Boundaries* theme.



Name	Visible	Labels	Active	Query	Attributes
Natural Streams	☑	□	□	□	☑
Model Boundaries	☑	□	☑	□	□
Shaded Elevation			□	□	□

Figure 8. *Theme Order* menu of the *Loaded Themes* window. The *Move Down* menu item is selected, which will move the selected theme down one position in theme order.

- The *Move Up* menu item will move the theme up one position on the *Map* window and *Loaded Themes* list.

- The *Move Down* menu item will move the theme down one position in the *Map* window and *Loaded Themes* list.

- The *Move Top* menu item will move the theme to the top position in the *Map* window and *Loaded Themes* list.

- The *Move Bottom* menu item will move the theme to the bottom position in the *Map* window the *Loaded Themes* list.

A series of check boxes are displayed next to the theme's name in the *Loaded Themes* window (fig. 8). These check boxes control the appearance and behavior of each theme in the *Map* window. A click on a box toggles an option; a check means "on" and blank means "off." If an option is not available for a particular theme, the check box will not turn on. These check boxes are as follows:

- The *Visible* check box indicates whether this theme is visible on the *Map* window. Click on this check box to toggle a theme's visibility without unloading it.

- The *Labels* check box indicates whether a theme's features are labeled on the *Map* window.

- The *Active* check box indicates the theme in the *Map* window that can be tracked by cursor movement. Information about the active theme is displayed in the tracking box. Only one theme can be active at a time. Upon activation of a theme, any previously active theme automatically becomes inactive and has its check box cleared.

- The *Query* check box indicates whether any data for a theme's features will be plotted when a feature is selected. This check box is available only if the theme has time-series data available for graphing.

- The *Attributes* check box will display a theme's GIS attribute table in the *GIS Attribute Tool*, as described in the "Software Tools" section.

Software Tools

There are six tools that provide custom data analysis and editing functionality in OUI. These are (1) the *Time Series Tool*, (2) the *Ensemble Streamflow Prediction (ESP) Tool*, (3) the *Parameter Tool*, (4) the *State Variable Animator*, (5) the *GIS Attribute Tool*, and (6) the *MMS Tool*. These tools are described herein.

Time Series Tool

The *Time Series Tool* is used for display and analysis of measured and simulated time-series data. Time-series data are accessed by using the *Query* option in the *Loaded Themes* window (fig. 6) and selecting a feature from the active theme in the *Map* window. A menu of time-series data will appear. Select the time-series data of interest by highlighting its name in the pop-up menu. The *Time Series Tool* will appear as a pop-up window. The *Time Series Tool* window includes a (1) *Trace List* window, (2) menu bar, and (3) plot window (fig. 9). As time-series data are selected by features in the *Map* window, entries are added to the *Trace List* window of the *Time Series Tool*. The multiple selections of the same entry will be ignored. These entries can be selected to generate plots and statistics.

There are three different methods for selecting entries from the *Trace List*:

- Select a single entry by clicking on the entry with the left mouse button.

- Select contiguous entries by pressing the `shift key` while clicking on entries with the left mouse button.

- Add or remove an entry from the selection by pressing the `control key` while clicking on an entry with the left mouse button.

The *Time Series Tool* menu bar has four options: *File, Statistics, Plots,* and *Help* (fig. 9).
The *File* menu has two options: *Get Trace From File* and *Quit*.

- *Get Trace From File* activates a file selector window. The user can select a time-series data set from either an MMS input or output file to load to the trace list.

- *Quit* option will close the *Time Series Tool.*

The *Statistics* menu provides the *Standard Statistic Report* (fig. 10). Any number of entries can be selected for this option. Information in this report includes count, missing value count, start time, end time, mean, minimum, maximum, standard deviation, and skewness. Click on the *Save* button in the *Standard Statistic Report* window to write this report to a file.
The *Plots* menu provides six time-series plot types: *Time Series, Time Series (Log Y-axis), XY, Difference, Difference Sum*, and *Flow Duration* plots (fig. 11).

- *Time Series* plot can be made from any number of entries that can be selected from the *Trace List*. Figure 11*A* shows the results of selecting both data sets from figure 9 for a *Time Series* plot.

- *Time Series (Log Y-axis)* shown in figure 11*B* displays the selection of two data sets from figure 10 and plotting them by using this option.

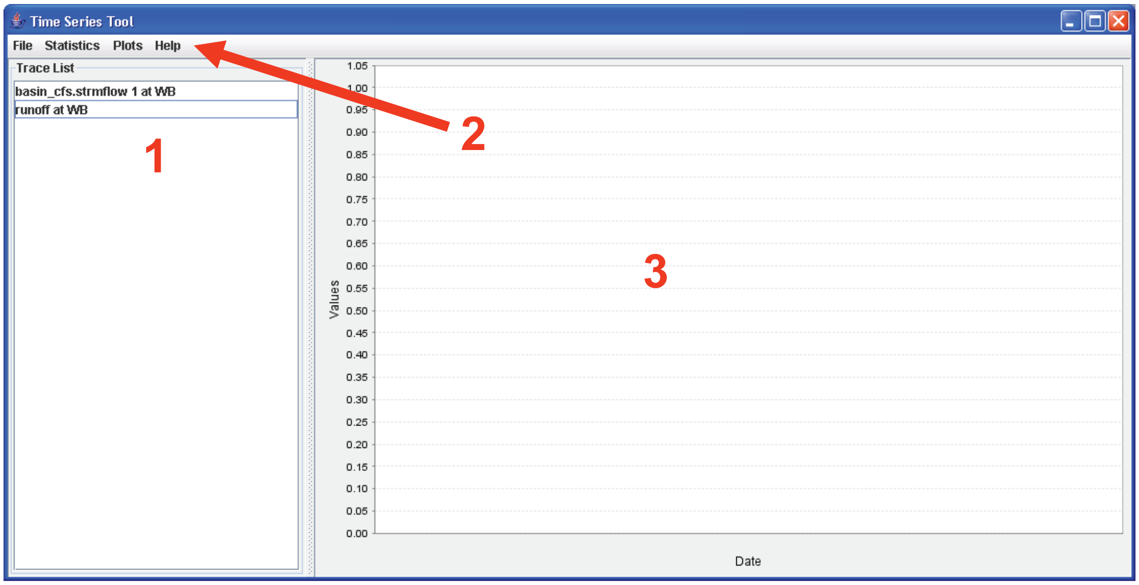
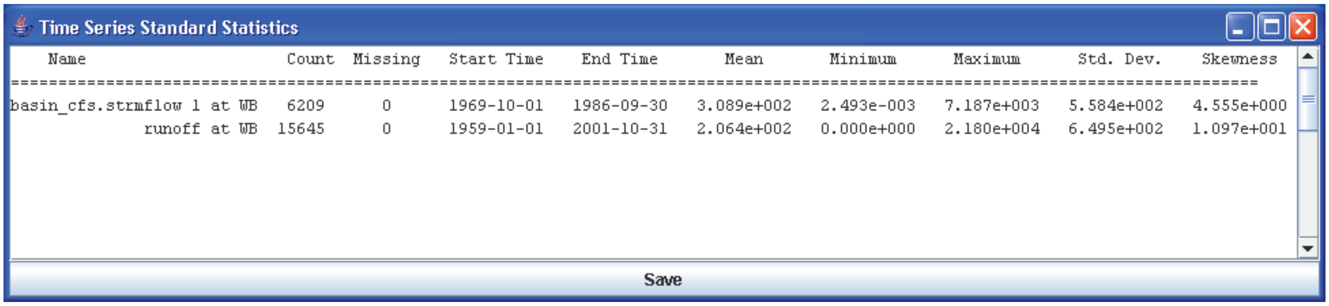


Figure 9. Time Series Tool with (1) two time series entries in the *Trace List* window, (2) the menu bar, and (3) the plot window.



```
    Name                     Count  Missing   Start Time   End Time     Mean        Minimum      Maximum      Std. Dev.    Skewness
=================================================================================================================================
basin_cfs.strmflow 1 at WB   6209      0      1969-10-01   1986-09-30   3.089e+002   2.493e-003   7.187e+003   5.584e+002   4.555e+000
              runoff at WB  15645      0      1959-01-01   2001-10-31   2.064e+002   0.000e+000   2.180e+004   6.495e+002   1.097e+001
```

Figure 10. Time Series Standard Statistics report for two time series.

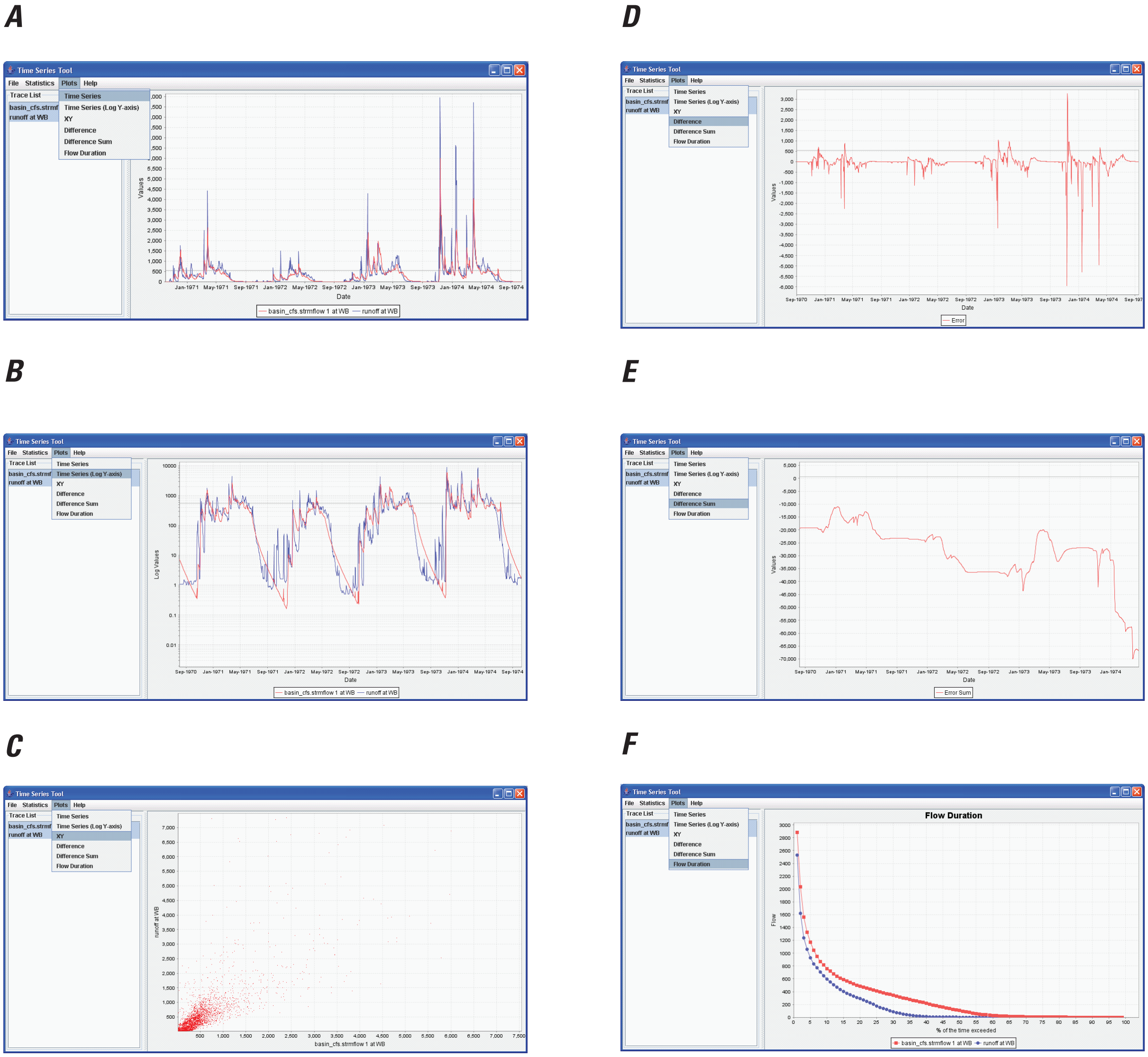


Figure 11. Windows showing *A*, the *Time Series Tool* with two time series plotted as a Time Series; *B*, the *Time Series Tool* with two time series plotted with *Log Y-axis*; *C*, the *Time Series Tool* with two time series as a *XY plot*; *D*, the *Time Series Tool* with two time series plotted as a *Difference plot*; *E*, the *Time Series Tool* with two time series plotted as a *Difference Sum plot*, and *F*, the *Time Series Tool* with two time series plotted as a *Flow Duration* plot.

- *XY* plot (fig. 11*C*) shows the same two time series displayed in figure 11*B*. In this plot, the first selected data entry is assigned to the x-axis and the second is assigned to the y-axis. If more than two entries are selected, the *Time Series Tool* uses the first two selected entries in the list. A point is plotted for each common time step in the two time series. If the model is simulating perfectly, the points will fall on the equal-value line. This plot is useful in determining certain kinds of model bias. In this case, measured streamflow for the West Branch watershed (runoff at WB) is plotted in relation to simulated streamflow (basin_cfs. streamflow 1 at WB). Most of the low flow values lie above the equal value line. This position indicates that the model is over-predicting in low conditions while simulating much better in high flow conditions.

- *Difference* plot (fig. 11*D*) shows the measured and simulated time-series data. The *Difference* time series is generated by subtracting the values of the selected time-series entries on a time-step-by-time-step basis. This plot also can reveal model bias in magnitudes and direction of cyclically occurring biases.

- *Difference Sum* plot (fig. 11*E*) shows the two time series displayed in figure 11*D*. This plot is similar to the *Difference* plot, but the plotted values include the summation of the difference on all previous time steps.

- *Flow Duration* plot (fig. 11*F*) shows the two time-series data. Any number of entries can be selected from the *Trace List* for a *Flow Duration* plot. These curves show the percentage of the number of time steps for each trace (x-axis) that exceed various flow values (y-axis). Substantial differences between the shapes of measured and simulated flow duration plots indicate model bias. If the measured flow duration curve is steeper than the simulated flow, the model may have a tendency to under predict individual storm events, particularly those caused by rainfall. If the measured flow duration curve is flatter that the simulated flow, the model may be showing insensitivity to streamflow originating from snowmelt, groundwater, or storage reservoir regulation.

The *Help* menu activates the OUI help system (fig. 4).

Ensemble Streamflow Prediction (ESP) Tool

The *OUI ESP Tool* (fig. 12) is used for display, analysis, and selection of ensembles of time series. The ESP methodology simulates possible future streamflow time series (ensembles) by combining current basin conditions and historical meteorological data. Details of the ESP methodology are discussed in detail by Day (1985). The “OUI Project Example: Feather River Basin” section describes how to generate and analyze these traces.

When a feature with ensemble data is selected in the *Map* window (fig. 7), the *ESP Tool* window is displayed. For example, the tree theme node *ESP Local Traces* contains features linked to ESP ensemble data (fig. 5). An entry for each trace of the ensemble selected from the ESP theme is added to the *Ensemble Traces* list of the *ESP Tool* (fig. 12). As entries are selected from this list, the corresponding trace is added to the plot. An *ESP Report* (fig. 13) can be generated for the selected entries.

Below the *Ensemble Traces* list, on the left-hand side of the GUI, is the *Rank By* box (fig. 12). The state of the *Volume, Peak,* and *Year* buttons determine the order of the entries in the *Ensemble Traces* list. The entries will be sorted from highest to lowest of the selected rank type. Only one of these buttons can be selected at a time.

The *Analysis Dates* window provides information about the *Ensemble Traces*. This window provides the Initialization and Forecast period of the ESP simulation. Users can specify an *Analysis Start* and *Analysis End* period. The *Analysis Dates* are used to set the extent of the data plot and in the *ESP Report*.

If the ESP Tool is connected to an external database with a DMI, users can specify a *DMI Start* and *DMI End* period in the *DMI Dates* window. These dates are used to specify the time period of the ensemble traces to export to the database. The Feather River Basin example does not use an external database, so this option is not available.

The *ESP Tool* menu bar has five options: *File*, *Select Years, Reports, DMIs,* and *Help.*
The *File* menu has two menu items: *Get Trace From File* and *Quit*.

- The *Get Trace From File* menu item allows the user to add additional time series to the ensemble plot.

- The *Quit* menu item closes the *ESP Tool.*

The *Select Years* menu will select groups of traces from the *Ensemble Traces* list on the basis of general climatic categories. It has five categories:

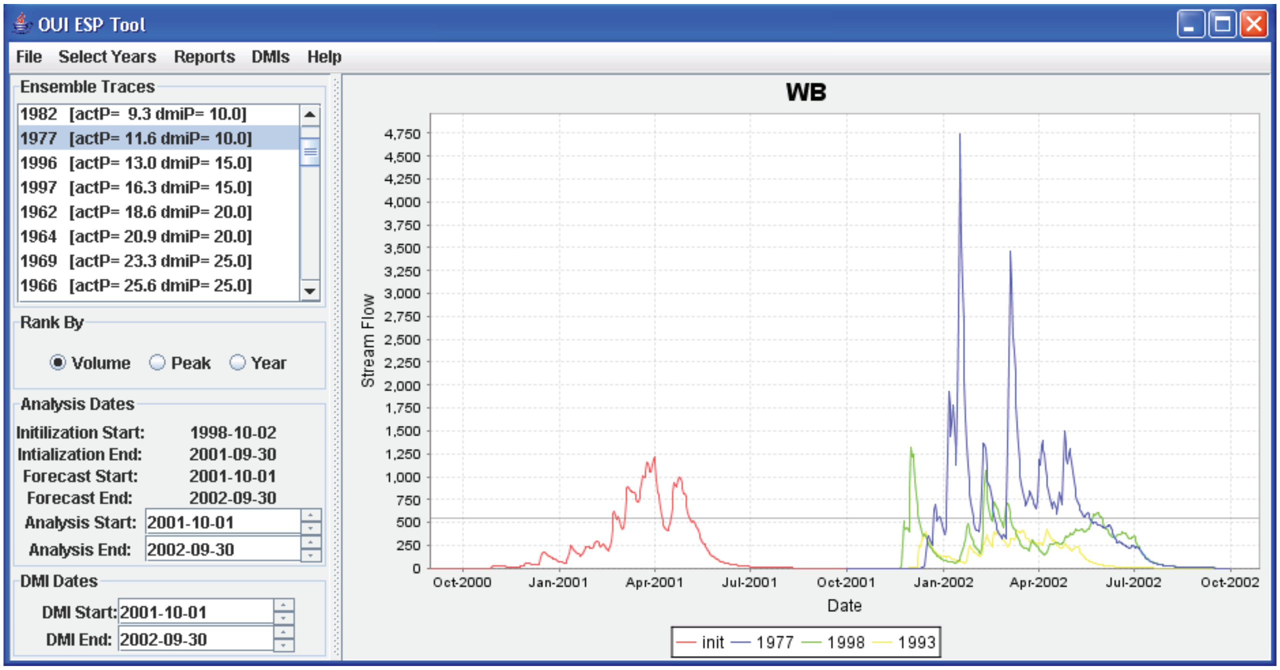
- *La Nina,* where the NINO3.4 sea surface temperatures are less than –0.5 degrees C below normal for the water year.

- *El Nino,* where the NINO3.4 sea surface temperatures are greater than 0.5 degrees C above normal for the water year.

- *ENSO Neutral,* where the NINO3.4 sea surface temperatures are within 0.5 degrees C of normal for the water year.

- *PDO < –0.5,* where the Pacific Decadal Oscillation index is considered negative for the water year.

- *PDO > 0.5,* where the Pacific Decadal Oscillation index is considered positive for the water year.



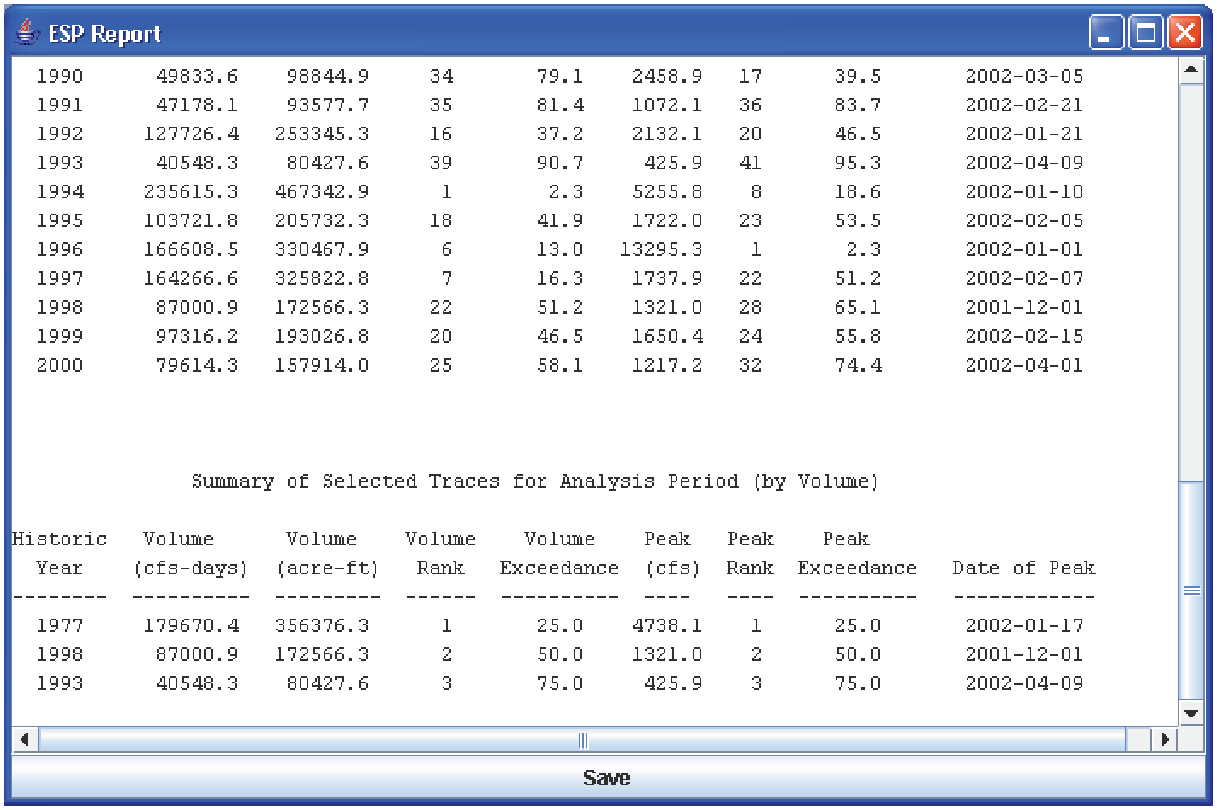
```
1982  [actP=  9.3 dmiP= 10.0]
1977  [actP= 11.6 dmiP= 10.0]
1996  [actP= 13.0 dmiP= 15.0]
1997  [actP= 16.3 dmiP= 15.0]
1962  [actP= 18.6 dmiP= 20.0]
1964  [actP= 20.9 dmiP= 20.0]
1969  [actP= 23.3 dmiP= 25.0]
1966  [actP= 25.6 dmiP= 25.0]
```

● Volume ○ Peak ○ Year

Initilization Start: 1998-10-02
Intialization End: 2001-09-30
Forecast Start: 2001-10-01
Forecast End: 2002-09-30
Analysis Start: 2001-10-01
Analysis End: 2002-09-30

DMI Start: 2001-10-01
DMI End: 2002-09-30

Figure 12. Ensemble Stream Prediction (*ESP) Tool* window with initialization period and an ensemble of three traces.



```
  1990      49833.6    98844.9     34        79.1     2458.9   17       39.5        2002-03-05
  1991      47178.1    93577.7     35        81.4     1072.1   36       83.7        2002-02-21
  1992     127726.4   253345.3     16        37.2     2132.1   20       46.5        2002-01-21
  1993      40548.3    80427.6     39        90.7      425.9   41       95.3        2002-04-09
  1994     235615.3   467342.9      1         2.3     5255.8    8       18.6        2002-01-10
  1995     103721.8   205732.3     18        41.9     1722.0   23       53.5        2002-02-05
  1996     166608.5   330467.9      6        13.0    13295.3    1        2.3        2002-01-01
  1997     164266.6   325822.8      7        16.3     1737.9   22       51.2        2002-02-07
  1998      87000.9   172566.3     22        51.2     1321.0   28       65.1        2001-12-01
  1999      97316.2   193026.8     20        46.5     1650.4   24       55.8        2002-02-15
  2000      79614.3   157914.0     25        58.1     1217.2   32       74.4        2002-04-01



               Summary of Selected Traces for Analysis Period (by Volume)

Historic    Volume      Volume    Volume    Volume      Peak   Peak    Peak
  Year    (cfs-days)  (acre-ft)    Rank   Exceedance   (cfs)   Rank  Exceedance    Date of Peak
--------  ----------  ---------  ------   ----------   ----    ----  ----------   ------------
  1977     179670.4   356376.3      1        25.0     4738.1    1       25.0        2002-01-17
  1998      87000.9   172566.3      2        50.0     1321.0    2       50.0        2001-12-01
  1993      40548.3    80427.6      3        75.0      425.9    3       75.0        2002-04-09
```

Figure 13. Ensemble Stream Prediction (*ESP)* Report window showing a report for a selected ensemble.

- *PDO Neutral*, where the Pacific Decadal Oscillation is considered neither positive nor negative for the water year.

The *Reports* menu provides the menu item, *Write Report*. Selecting this option invokes the *ESP Report* window. The report generated is a summary table of the ESP ensemble traces loaded to the *ESP Tool* (fig. 13). Information in this report includes volume, volume rank, and volume exceedance; and peak, peak rank, and date of peak for each trace of the ensemble for the analysis period. This report may be saved to a text file by using the *Save* button at the bottom of the window.

The *DMIs* menu provides the option to *Run ESP DMI for Selected Traces*. This option will run the DMI specified in *project.xml*. If no DMI is specified, no DMI will run.

The *Help* menu activates the OUI help system (fig. 4).

Parameter Tool

The *Parameter Tool* (fig. 14) is used to display and edit model parameters and dimensions. In the OUI project example provided with the current installation, this tool is invoked from the project tree by opening *Models & Data, Input, MMS Parameter Files, PRMS Parameter Files,* right-clicking on the file of interest and selecting the *Run* option. The *Parameter Tool* contains a (1) menu bar, (2) *tree* window, (3) *editor table* window, and (4) convenience buttons. These items are described below.

The *Parameter Tool* menu bar has three options: *File, Reset,* and *Help*.

The *File* menu of the *Parameter Tool* has four menu items: *Load, Save, Save As,* and *Exit*.

- The *Load* menu item loads a parameter file.
- The *Save* menu item saves the edited values to the current file.
- The *Save As* menu item saves the edited values to a new file.
- The *Exit* menu item exits the *Parameter Tool*.

The *Reset* menu of the *Parameter Tool* has two menu items: *Reset This Table* and *Reset All to Original Values*.

- The *Reset This Table to Original Values* menu item resets the values in the editor table to their original values.
- The *Reset All to Original Values* menu item resets all parameter and dimension values to their original values.

The *Help* menu activates the OUI help system (fig. 4).

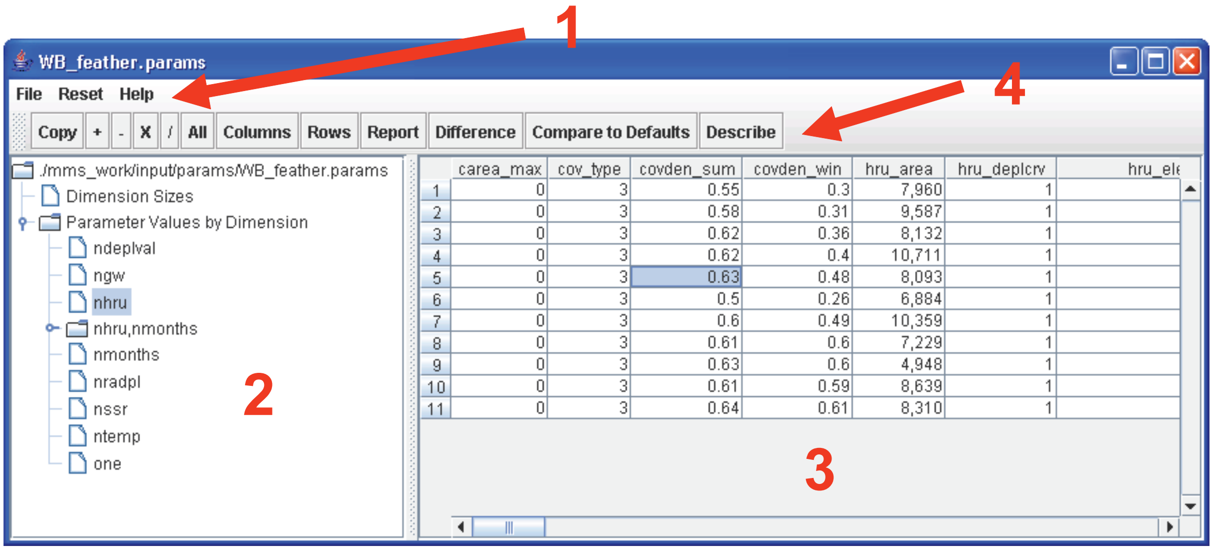
The *tree* window of the *Parameter Tool* (fig. 14) determines the content of the *editing table* window. The top level of the tree represents a parameter file with two items:

- Selecting *Dimension Sizes* loads the dimensions of the parameters into the editor table. The values of the dimensions represent the sizes of parameter arrays; thus, users can change the size of the parameter arrays by editing these values.
- Selecting *Parameter Values by Dimension* reveals the dimension structure of the parameter arrays. Selecting a dimension item will load all of the parameters of the dimension into the editor table.

Parameter values can be edited in the table by double-clicking with the left mouse button on a cell in the table and typing in a new value. Parameter values can be selected in the table by clicking and holding down the left mouse button on a cell in the table, moving the mouse to a new cell (defining the selection extent), and releasing the left mouse button. The selected cells will be a different color than the nonselected cells. The values in the editor table can be sorted on parameter values, in either ascending or descending order, by clicking on a column header with the left mouse button. Sorting by a parameter value changes only the order of the parameters in the editor table. It does not change the order in the parameter file.

The *Parameter Tool* has 12 convenience buttons located just below the menu bar:

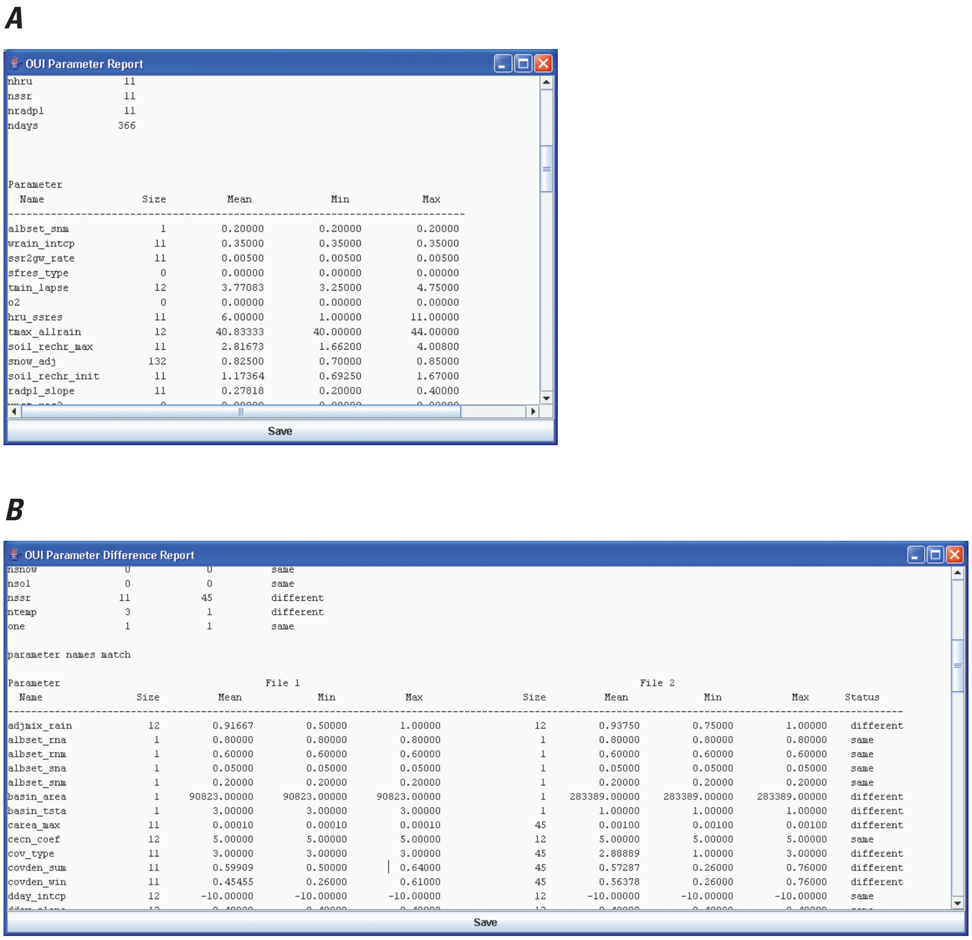
- The *Copy* button sets the selected table editor cells to the specified value.
- The + *(addition)* button adds the specified value to each selected table editor cell.
- The – *(subtraction)* button subtracts the specified value from each selected table editor cell.
- The *X (multiplication)* button multiplies the specified value with each selected table editor cell.
- The */ (division)* button divides each selected table editor cell by the specified value.
- The *All* button selects all cells in the table editor.
- The *Columns* button selects all cells in all columns with currently selected cells in the table editor.
- The *Rows* button selects all cells in all rows with currently selected cells in the table editor.
- The *Report* button generates a summary report of the dimension and parameter values for the selected parameter file (fig. 15*A*). Information in this report includes the parameter file name, dimension names and sizes, and parameter names, sizes, mean, minimum, and maximum values. This report can be written to a file by clicking on the *Save* button.


WB_feather.params
File Reset Help
Copy + - X / All Columns Rows Report Difference Compare to Defaults Describe
1
2
3
4

./mms_work/input/params/WB_feather.params
Dimension Sizes
Parameter Values by Dimension
ndeplval
ngw
nhru
nhru,nmonths
nmonths
nradpl
nssr
ntemp
one

	carea_max	cov_type	covden_sum	covden_win	hru_area	hru_deplcrv	hru_ele
1	0	3	0.55	0.3	7,960	1	
2	0	3	0.58	0.31	9,587	1	
3	0	3	0.62	0.36	8,132	1	
4	0	3	0.62	0.4	10,711	1	
5	0	3	0.63	0.48	8,093	1	
6	0	3	0.5	0.26	6,884	1	
7	0	3	0.6	0.49	10,359	1	
8	0	3	0.61	0.6	7,229	1	
9	0	3	0.63	0.6	4,948	1	
10	0	3	0.61	0.59	8,639	1	
11	0	3	0.64	0.61	8,310	1	

Figure 14. *Parameter Tool* window with parameter values from an MMS parameter file showing (1) menu bar, (2) tree window, (3) editor table window, and (4) convenience buttons.



```
nhru                11
nssr                11
nradpl              11
ndays              366



Parameter
  Name                 Size          Mean            Min             Max
--------------------------------------------------------------------------
albset_snm                1       0.20000        0.20000         0.20000
wrain_intcp              11       0.35000        0.35000         0.35000
ssr2gw_rate              11       0.00500        0.00500         0.00500
sfres_type                0       0.00000        0.00000         0.00000
tmin_lapse               12       3.77083        3.25000         4.75000
o2                        0       0.00000        0.00000         0.00000
hru_ssres                11       6.00000        1.00000        11.00000
tmax_allrain             12      40.83333       40.00000        44.00000
soil_rechr_max           11       2.81673        1.66200         4.00800
snow_adj                132       0.82500        0.70000         0.85000
soil_rechr_init          11       1.17364        0.69250         1.67000
radpl_slope              11       0.27818        0.20000         0.40000
```

```
nsnow              0             0          same
nsol               0             0          same
nssr              11            45          different
ntemp              3             1          different
one                1             1          same

parameter names match

Parameter                                    File 1                                                            File 2
  Name                 Size          Mean            Min             Max               Size          Mean            Min             Max       Status
---------------------------------------------------------------------------------------------------------------------------------------------------------
adjmix_rain              12       0.91667        0.50000         1.00000                 12       0.93750        0.75000         1.00000    different
albset_rna                1       0.80000        0.80000         0.80000                  1       0.80000        0.80000         0.80000    same
albset_rnm                1       0.60000        0.60000         0.60000                  1       0.60000        0.60000         0.60000    same
albset_sna                1       0.05000        0.05000         0.05000                  1       0.05000        0.05000         0.05000    same
albset_snm                1       0.20000        0.20000         0.20000                  1       0.20000        0.20000         0.20000    same
basin_area                1   90823.00000    90823.00000     90823.00000                  1  283389.00000   283389.00000    283389.00000    different
basin_tsta                1       3.00000        3.00000         3.00000                  1       1.00000        1.00000         1.00000    different
carea_max                11       0.00010        0.00010         0.00010                 45       0.00100        0.00100         0.00100    different
cecn_coef                12       5.00000        5.00000         5.00000                 12       5.00000        5.00000         5.00000    same
cov_type                 11       3.00000        3.00000         3.00000                 45       2.88889        1.00000         3.00000    different
covden_sum               11       0.59909        0.50000      |  0.64000                 45       0.57287        0.26000         0.76000    different
covden_win               11       0.45455        0.26000         0.61000                 45       0.56378        0.26000         0.76000    different
dday_intcp               12     -10.00000      -10.00000       -10.00000                 12     -10.00000      -10.00000       -10.00000    same
```

Figure 15. *A,* The *Parameter Report* window with parameter values from an MMS parameter file and *B,* the *Parameter Difference Report* window with parameter values from an MMS parameter file.

- The *Difference* button generates a summary report of the comparison of the dimension and parameter values in the selected parameter file and another (fig. 15*B*). Information in this report includes the two parameter file names, dimension names and sizes, and parameter names, sizes, mean, minimum, and maximum values. Any differences between the dimension and parameter values are identified. This report can be written to a file by clicking on the *Save* button.

- The *Compare to Defaults* button generates a summary of report of the difference between the default parameter value settings and the settings in the current selected parameter file. This report can be written to a file by clicking on the *Save* button.

- The *Describe* button will display a *Parameter Description* pop-up window for parameters in the *table* window that have selected cells.

State Variable Animator

The *State Variable Animator* creates animations of output variables generated by a model run, such as snow-covered area or soil-moisture content, that are draped over a digital elevation model map of the modeled region. The *State Variable Animator* is used for display and analysis of spatially and temporally distributed data. As features with time-series data are selected from the *Map* window (fig. 2), the *State Variable Animator* window will appear with a relief map showing elevation. The available state variables for a theme are displayed in the menu box under the map image. Figure 16 shows the *State Variable Animator* window after the state variable *pkwater_equiv* (liquid equivalent depth of the snowpack) has been selected.

The *Key* on the left side of the map shows the value range of the state variable broken into intervals. The color of the shading goes from cyan (lowest value) through gray (medium values) to red (highest values). The *Control* buttons directly under the map provide the following functions (described from left to right):

- *Double left arrow* runs the animation backwards quickly.

- *Left arrow bar* steps the animation backwards one time step.

- *Red stop* stops the animation.

- *Double bar* pauses the animation.

- *Right arrow* runs the animation forward.

- *Bar right arrow* steps the animation forward one time step.

- *Double right arrow* runs the animation forward quickly.

- *Movie* records the animation to a file.

The *Current Date* below the *Control* buttons shows the date of the data displayed on the *Map*. The *Map Date* slider bar shows the range of dates of the simulation and allows the user to set the animation date by clicking on the slider and dragging it directly to the desired date. The *Z Scale* slider allows the user to set the amount of elevation relief of the *Map*. Finally, the *Animation Speed* slider allows the user to set the speed of the animation.

The *State Variable Animator* menu bar has two options: *File,* and *Display.*
The *File* menu of the *State Variable Animator* has three menu items: *Load Files, Add Vector Map* and *Exit.*

- The *Load Files* menu item loads a new set of map and data files.

- The *Add Vector Map* menu item loads a shapefile that will be displayed on top of the animation.

- The *Exit* menu item exits the *State Variable Animator*.

- The *Display* menu of the *State Variable Animator* has three menu items: *Display, Recenter,* and *Define Key*.

- The *Display DEM* toggle menu item determines whether the underlying digital elevation model map is visible through the animation colors in the map.

- The *Recenter Map* menu item resets the visible animation map to the center of the map display window.

- The *Define Key* menu item displays the *Key Editor* window.

- The *Key Editor* window allows users to modify the *Key* and colors used for the animation. The number of bins, as well as minimum and maximum value range of the *Key,* is set here. Also, an alternative color ramp can be specified. Finally, the slider allows the width of the *Key* bins to be set individually.

GIS Attribute Tool

The *GIS Attribute Tool* (fig. 17) is invoked in the *Loaded Themes* window by checking the *Attributes* check box next to a theme of interest (fig. 8). This window may be turned off either by unchecking the *Attributes* check box in the *Loaded Themes* window, by closing the window with the upper right-hand “x,” or by selecting *Quit* under the menu bar option *File*. GIS attributes may be added, deleted, or modified. Attribute values may be edited by double-clicking with the left mouse button on the appropriate cell in the table and typing the new value. Changes may be saved by using one of the methods provided under the *File* menu in order to complete the edits.

The *GIS Attribute Tool* window includes three menu bar options; *File, Attribute,* and *Help*.

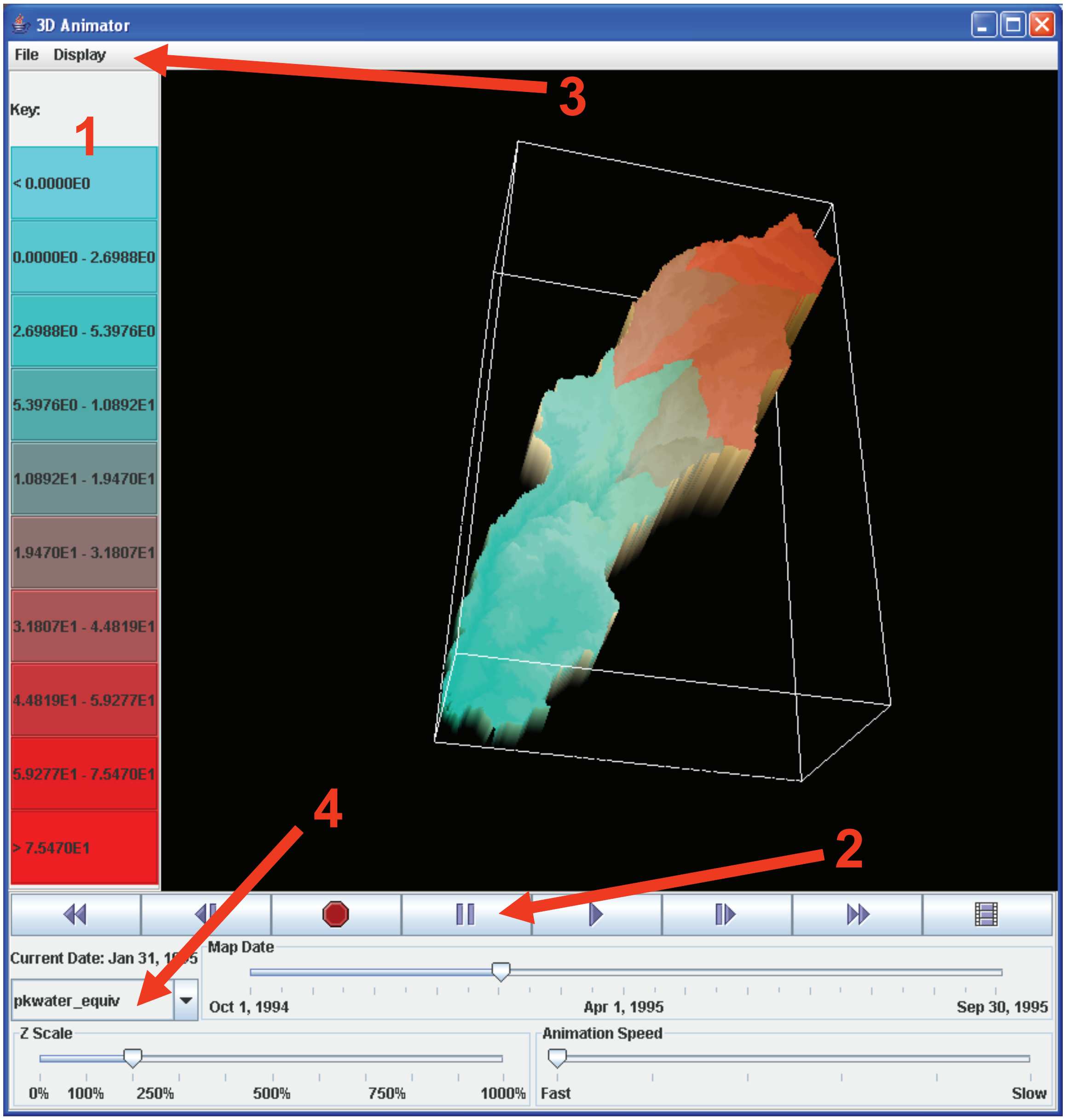
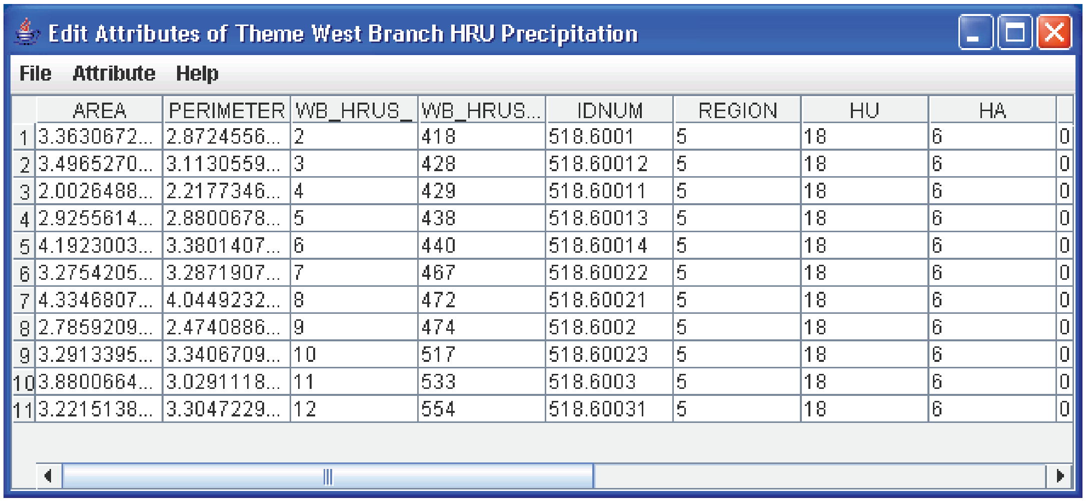


Figure 16. *State Variable Animator* window showing spatially distributed snowpack water equivalent on January 31, 1995 for the West Branch watershed. The image shows (1) key for state variable categories, (2) control buttons, (3) menu bar, and (4) state variable selector menu.



	AREA	PERIMETER	WB_HRUS_	WB_HRUS...	IDNUM	REGION	HU	HA	
1	3.3630672...	2.8724556...	2	418	518.6001	5	18	6	0
2	3.4965270...	3.1130559...	3	428	518.60012	5	18	6	0
3	2.0026488...	2.2177346...	4	429	518.60011	5	18	6	0
4	2.9255614...	2.8800678...	5	438	518.60013	5	18	6	0
5	4.1923003...	3.3801407...	6	440	518.60014	5	18	6	0
6	3.2754205...	3.2871907...	7	467	518.60022	5	18	6	0
7	4.3346807...	4.0449232...	8	472	518.60021	5	18	6	0
8	2.7859209...	2.4740886...	9	474	518.6002	5	18	6	0
9	3.2913395...	3.3406709...	10	517	518.60023	5	18	6	0
10	3.8800664...	3.0291118...	11	533	518.6003	5	18	6	0
11	3.2215138...	3.3047229...	12	554	518.60031	5	18	6	0

Figure 17. *GIS Attribute Tool* window for editing attributes of a theme.

The *File* menu has four options: *Save to dbf File, Write Report File*, *Write XML File,* and *Quit*.

- *Save to dbf File* will save the GIS attribute information to a file in dBASE file format (dbf). This option is used to update the attributes of the selected shapefile theme.
- *Write Report File* will save the GIS attribute information as a text file report.
- *Write XML File* will save the GIS attribute information to a file in XML format.
- *Quit* option will close the *GIS Attribute Tool* window.

The *Attribute* menu has three options: *Add attribute, Delete attribute,* and *Set label attribute*.

- *Add Attribute* item allows the user to add an attribute column to the table. A blank column will be added to the right end of the table.
- *Delete Attribute* allows the user to delete an attribute column from the table. Select this option and click on the column to delete.
- *Set Label Attribute* sets the selected attribute as the labels for the feature in the Map window.

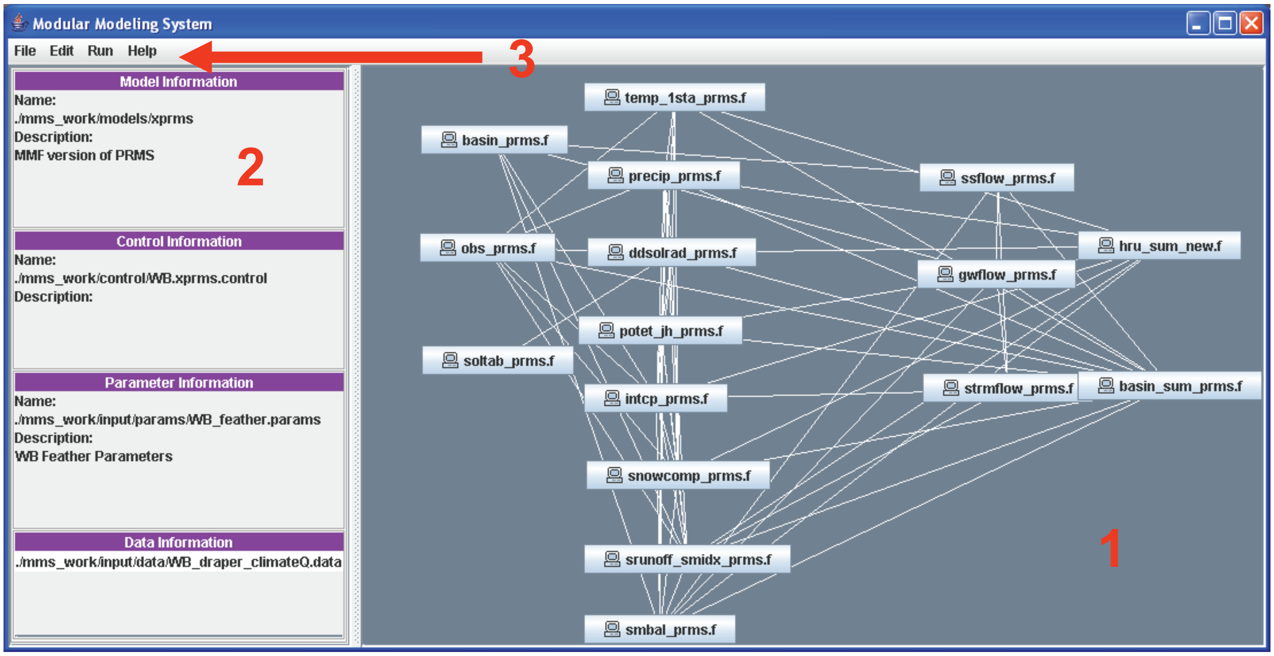
The *Help* menu activates the OUI help system (fig. 4) as described above.

MMS Tool

The *MMS Tool* (fig. 18) is used to run MMS models. MMS is a modeling framework that allows a user to selectively couple the most appropriate process algorithms from applicable models to create an "optimal"model for the desired application. MMS is described in detail by Leavesley and others (1996b). In the OUI project example provided with the current installation, this tool is invoked from the project tree by opening *Models & Data, MMS Runs,* right-clicking on the model of interest, and selecting the *Run* option. The *MMS Tool* contains a (1) *schematic* window that illustrates the physical processes simulated by the model, (2) *information* window that shows the input file used by the model, and (3) a menu bar.

The *MMS Tool* menu bar has four options: *File, Edit, Run,* and *Help*. These options are described in detail by Leavesley and others (1996b) and summarized below.
The *File* menu of the *MMS Tool* has eight menu items:

- The *Load Data Files* menu item loads a data file.
- The *Load Parameters* menu item loads a parameter file.
- The *Load Control File* menu item loads a control file.
- The *Set Model* menu item loads an executable model file.
- The *Save Parameters* menu item saves the edited parameters values to the current parameter file.
- The *Save Control* menu item saves the current GUI settings to the control file.
- The *Set File Names* menu item displays the file name editor, as described by Leavesley and others (1996b).
- The *Exit* menu item exits the *MMS Tool*.



Model Information
Name:
./mms_work/models/xprms
Description:
MMF version of PRMS

Control Information
Name:
./mms_work/control/WB.xprms.control
Description:

Parameter Information
Name:
./mms_work/input/params/WB_feather.params
Description:
WB Feather Parameters

Data Information
./mms_work/input/data/WB_draper_climateQ.data

Figure 18. *MMS Tool* window for running MMS models showing (1) schematic window, (2) information window, and (3) menu bar.

The *Edit* menu of the *MMS Tool* has four menu items:

- The *Parameters & Dimensions* menu item displays the *Parameter Tool*, as described above, for the current parameter file.
- The *Parameter Info* menu item allows the user to edit the description of the parameter file.
- The *Control Info* menu item allows the user to edit the description of the control file.
- The *Model Info* menu item allows the user to edit the description of the executable model.

The *Run* menu of the *MMS Tool* has four menu items:

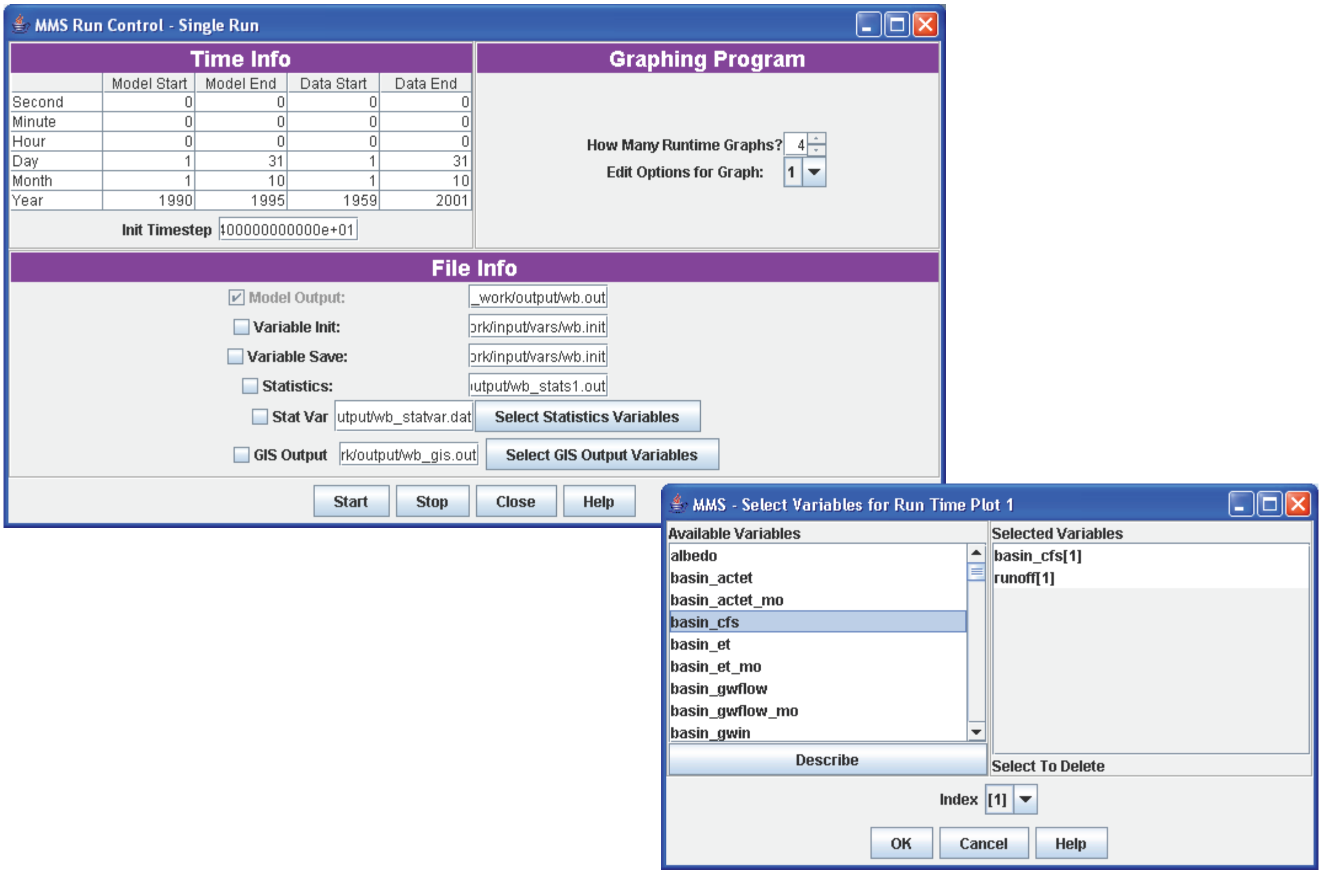
- The *Single Run* menu item displays the *Single Run* window, described in detail by Leavesley and others (1996b) and summarized below.
- The *Sensitivity* menu item displays the *Sensitivity* window, as described in detail by Leavesley and others (1996b).
- The *Optimization* menu item displays the *Optimization* window, as described in detail by Leavesley and others (1996b).
- The *ESP* menu item displays the *ESP* window, as described in detail by Leavesley and others (1996b).

The *Help* menu activates the *OUI* help system (fig. 4) as described above.

The *Single Run* window (fig. 19) allows users to control MMS model execution. This GUI is described in detail by Leavesley and others (1996b), and summarized here. This option enables the user to run the model for a selected period of time by using the parameter and data files shown in the *MMS Tool* window (fig. 18). The *MMS Single Run* window contains a number of features that can be used to define the period of the run, the analyses conducted on the output, and specify output file names (fig. 19).

The upper left quadrant of the window contains the *Time Info* entry box. Users can set the start and end times (year, month, day, hour, minute, and second) and the initial time step, in hours, for the run. In the lower half is the *File Info* entry box. Model output can be saved to a file by selecting the *Model Output* toggle button and entering a file name in the text entry window.

When a model is run, there are two options available for the initialization of declared model variables. One is that all declared variables are initialized according to the initialize sections of the program modules. The second is that all declared variables can be initialized from a set of variables saved at the end of a previous run. The option to save all the declared model variables at the end of a run is selected by pressing the *Variables Save* toggle button on and entering a file name in the text entry window. To use a previously saved variables file, press the *Variables Init* toggle button on and enter the name of the variables file in the text entry window.


MMS Run Control - Single Run

Time Info

	Model Start	Model End	Data Start	Data End
Second	0	0	0	0
Minute	0	0	0	0
Hour	0	0	0	0
Day	1	31	1	31
Month	1	10	1	10
Year	1990	1995	1959	2001

Init Timestep 400000000000e+01

Graphing Program

How Many Runtime Graphs? 4

Edit Options for Graph: 1

File Info

☑ Model Output: _work/output/wb.out

□ Variable Init: ork/input/vars/wb.init

□ Variable Save: ork/input/vars/wb.init

□ Statistics: utput/wb_stats1.out

□ Stat Var utput/wb_statvar.dat

□ GIS Output rk/output/wb_gis.out

MMS - Select Variables for Run Time Plot 1

Available Variables: albedo, basin_actet, basin_actet_mo, basin_cfs, basin_et, basin_et_mo, basin_gwflow, basin_gwflow_mo, basin_gwin

Selected Variables: basin_cfs[1], runoff[1]

Index [1]

Figure 19. *MMS Single Run* window with the *Select Variables for Run Time Plot* window inset.

In the upper right quadrant of the *MMS Single Run* window, the user can specify the *Graphing Program* options (fig. 19). Selecting the graph number from the *Edit Options for Graph* selector will display the *MMS Select Variables for Run Time Plot* window. In this window, the *Available Variables* list contains all of the variables that are available for plots. Selected variables in this list can be added to the *Selected Variables* list by choosing the desired array index from the *Index* list. Selecting the *Describe* button will display reference information about the variable. Variables in the *Selected Variables* list can be removed from the list by selecting with a right click of the mouse. All variables in the *Selected Variables* list will be plotted when the model runs.

The *Run Time Plot* window (fig. 20) option menu is available by right-clicking with the mouse on the plots. This menu provides a variety of display and printing options. The zoom function is enabled by left-clicking with the mouse on the plots.

Output of selected time-series variables is controlled by the *Selected Statistics Variables* button (fig. 20). Selection of these variables is similar to the process described for the *Graphing Program* above. Output files used by the *State Variable Animator* are controlled by the *Select GIS Output Variables* button. Selection of these variables is similar to the process described for the *Graphing Program* above. These options are described in detail by Leavesley and others (1996b).

At the bottom of the *Single Run* window are the execution control buttons: (1) the *Start* button runs the model, (2) the *Stop* button will stop the execution of a running model, (3) the *Close* button will close the *Single Run* window, and (4) the *Help* button will activate the OUI help system (fig. 4).

OUI Project Example: Feather River Basin

This section is intended as a self-guided demonstration of an OUI project and also may serve as a project documentation template. This example demonstrates the application of the Precipitation-Runoff Modeling System (PRMS) (Leavesley and others, 1983; Jeton, 1999, and Risley, 1994).

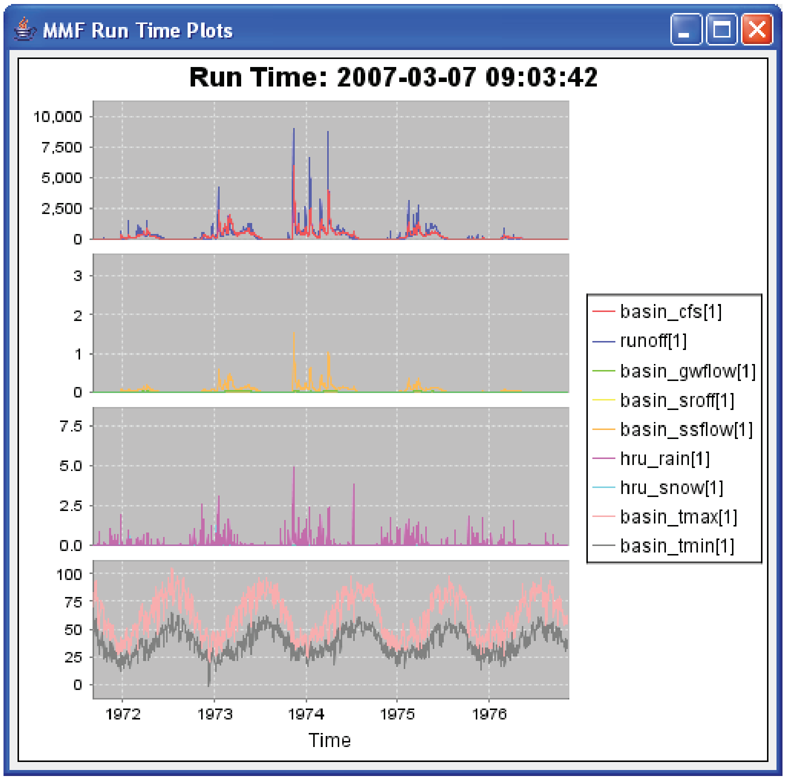


Figure 20. *Run Time Plots* window.

Description of the Feather River Basin

The following excerpt from Koczot and others (2005), describes the Feather River Basin modeling project:

> "Precipitation-runoff processes in the Feather River Basin of northern California determine short- and long-term streamflow variations that are of considerable local, State, and Federal concern. The river is an important source of water and power for the region. The basin forms the headwaters of the California State Water Project. Lake Oroville, at the outlet of the basin, plays an important role in flood management, water quality, and the health of fisheries as far downstream as the Sacramento–San Joaquin Delta. Existing models of the river simulate streamflow in hourly, daily, weekly, and seasonal time steps, but cannot adequately describe responses to climate and land-use variations in the basin. New spatially detailed precipitation-runoff models of the basin have been developed to simulate responses to climate and land-use variations at a higher spatial resolution than was available previously. This report characterizes daily rainfall, snow pack evolution, runoff, water and energy balances, and streamflow variations from, and within, the basin above Lake Oroville. The new model's ability to predict streamflow is assessed. The Feather River Basin sits astride geologic, topographic, and climatic divides that establish a hydrologic character that is relatively unusual among the basins of the Sierra Nevada. It straddles a north-south geologic transition in the Sierra Nevada between the granitic bedrock that underlies and forms most of the central and southern Sierra Nevada and volcanic bedrock that underlies the northernmost parts of the range (and basin). Because volcanic bedrock is more permeable than granitic, the northern, volcanic parts of the basin contribute larger fractions of ground-water flow to streams than do the southern, granitic parts of the basin. The Sierra Nevada topographic divide forms a high altitude ridge line running northwest to southeast through the middle of the basin. The topography east of this ridge line is more like the rain-shadowed basins of the northeastern Sierra Nevada than the uplands of most western Sierra Nevada river basins. The climate is Mediterranean, with most of the annual precipitation occurring in winter. Because the basin includes large areas that are near the average snowline, rainfall and rain-snow mixtures are common during winter storms. Consequently, the overall timing and rates of runoff from the basin are highly sensitive to winter temperature fluctuations.

A distributed-parameter, physically based, Precipitation-Runoff Modeling System (PRMS; Leavesley and others, 1983) was constructed for the Feather River Basin. The Feather River PRMS is composed of eight models representing eight drainages of the basin. Together, these models simulate streamflow from 98 percent of the basin above Lake Oroville. The models simulate daily water and heat balances, snow pack evolution and snowmelt, evaporation and transpiration, subsurface water storage and outflows, and streamflow at key streamflow gage sites. The drainages are modeled as 324 hydrologic-response units, each of which is assumed homogeneous in physical characteristics and response to precipitation and runoff. Calibration focused primarily on simulating flow during the April-July snowmelt season from 1971 to 1997, secondly on monthly simulations, and finally on daily flow characteristics. Model simulations were compared to monthly total natural inflows into Lake Oroville as reconstructed by the California Department of Water Resources. The models are most sensitive to input values and patterns of precipitation and soil characteristics. The input precipitation values were allowed to vary on a daily basis to reflect available observations by making daily transformations to an existing map of long-term mean monthly precipitation rates that account for altitude and rain-shadow effects."

Start OUI

OUI is now ready to run on the *Feather River* project. For windows systems, run the *feather.bat* file. For Linux systems, run the *feather.sh* file. The GUI should appear ready to load maps, data, and models. In the upper left window (fig. 2), click on the *Basin Maps* node in the *Feather River Project* tree. A list of available maps will display (fig. 5). At the top of the project tree window are four column titles (*Name, Description, Theme,* and *Type*). Move the mouse pointer to the divider line between the *Name* and *Description* column titles. Hover cursor over divider line; click left mouse button and drag the divider line to the right so that the names of the *Basin Maps* are fully displayed. Find the map titled *Shaded Elevation*. Right-click on this map and choose *Load* from the pop-up menu. The digital elevation model map for the *Feather River* project will appear in the *Map* window on the right. Having no processing or querying functionality, this map is for display purposes only.

In the *project tree* find the map entitled *Model Boundaries* in *Basin Maps*. Load this map. This map shows the delineation of eight separate drainages modeled in the Feather River Basin. With the mouse pointer, select the *Labels* check box for *Model Boundaries* in the *Loaded Themes* window at the bottom left. The model area names will appear on the *Map* window. Select the *Active* check box for *Model Boundaries* in the *Loaded Themes* window. Move the mouse pointer over the model areas in the *Map* window. Notice the *Tracking* box at the bottom of the *Map* window. Tracking provides the feature name of the currently active theme. Only one theme can be active at a time.

Use the mouse pointer to add other maps to the *Map* window. Maps can be removed from the *Map* window by a right mouse click on the map name in the *Loaded Themes* window.

Run Models

Left-click on the *Models & Data node* in the *project tree*. Three nodes (*Input, Single Run,* and *ESP Run*) will appear. These options are described below.

To view a model's input time-series data, click on the *Input* and then the *Update MMS Data Files* nodes in the project tree (fig. 5). Load the *Temperature Data* map. In the *Loaded Themes* window (fig. 8), put the *Temperature Data* map in *Query* mode (fig. 8). In the *Map* window (fig. 7), use the mouse pointer to select the *Canyon Dam* feature. Choose *tmax* from the pop-up menu. Repeat this procedure to choose *tmin* as well. The *Time Series Tool* (fig. 11) will appear with the data selected from the *Map* window. Choose the first item in the list in the *Time Series Tool*. Select the second data item by pressing the `shift key` while selecting with the mouse. Select *Plots* and then *Time Series* from the menu bar. Create other plots and statistic reports by selecting other time-series data sets from the input data maps and loading them into the *Time Series Tool*.

Single Run permits the user to make *PRMS* streamflow simulations for a user-specified period of record. Streamflow simulations from the eight drainage models are automatically routed and summed downstream. Inflow and outflow totals are computed for each stream segment within the modeled areas and also at the outlet point, at Lake Oroville.

To run the models in *Single Run* mode, right-click on the *Single Run* node (fig. 5) and choose *Run* from the pop-up menu. The *Run MMS Model* GUI will appear. The model has data from October 1, 1969, to October 31, 2001. Set the *Run Start Date* and *Run End Date* to the desired time period within the existing data. The default run period is the entire data record. Select the *Run* button to start the *Single Run* mode run of the model. A pop-up dialog will indicate when the model completes.

To view individual output from each drainage model, click on the *Single Run* node (fig. 5) and load the *PRMS Basin Output* map. Put the map in *Query* mode and select the desired model output sites in the *Map* window. The selected simulated streamflow time series will load into the *Time Series Tool*. Load observed streamflow time series from selections in the *Input Streamflow Data Map* to evaluate model performance.

To view accumulated and routed model output, load the *xroute Accumulated and Routed Output* (fig. 5). The map, in this case, is the stream-routing network. Put the map in *Query* mode (fig. 8) and select the desired model output reach. The selected simulated streamflow time series will load into the *Time Series Tool* (fig. 11). Load observed streamflow time series from the *Input Streamflow Data Map* (fig. 5) to evaluate model performance.

ESP Run (fig. 5) permits the user to make streamflow forecasts for a user-specified forecasting period by using historical temperature and precipitation data contained within the existing *PRMS* model data files. Results are automatically routed and summed downstream. As with the *Single Run* option, inflow and outflow totals are computed for stream segments within and at the Feather River Basin at Lake Oroville.

To run the models in *ESP Run* mode, right-click on the *ESP Run* node (fig. 5) and choose *Run* from the pop-up menu. The *Run ESP MMS Model* GUI will appear. The model has data from October 1, 1969 to October 31, 2001. The earliest *Forecast Start Date* can be 3 years after the start of the data record and the latest can be the day after the end of the data record. The *Forecast End Date* can be any date after the *Forecast Start Date*. Select the *Run* button to start the *ESP Run* mode run of the model. A pop-up dialog will indicate when the model completes.

To view individual model-area forecasts, click on the *ESP Run* node (fig. 5). From the *project tree*, load the *ESP Local Traces* map. Put the map in *Query* mode (fig. 8) and select the desired model output sites. The selected ensemble of streamflow forecast time series will load into the *OUI ESP Tool* (fig. 12).

To view accumulated and routed forecasts, load the *ESP Accumulated and Routed Output* (fig. 5). Put the map in *Query* mode (fig. 8) and select the desired model output sites. The selected ensemble of streamflow forecast time series will load into the *OUI ESP Tool* (fig. 12).

OUI Project Configuration

The *project.xml* file is a control file that provides OUI project configuration. It allows modelers to control the content, appearance, and functionality of OUI by defining paths, tree nodes, themes, and metadata for the OUI project. These concepts are described below. This section assumes the reader has a basic understanding of XML and object-oriented programming terminology. Unless otherwise noted, for the remainder of this document the term "attribute" will refer to XML attributes only.

Introduction to XML

The XML language can be used to capture the hierarchical information needed to describe complicated sets of models, data, and the interactions between them in a precise form. This makes XML an ideal descriptive language for configuring OUI to specific projects. XML provides validation of file syntax and can enforce a defined grammar and nomenclature. This means that XML files are both self-validating and self-describing, while allowing flexibility of the contents. The relation between themes, models, data, files, metadata, and tools can be described clearly. Finally, with a little practice, XML is easy to write, modify, and understand.

Examples of XML files are presented in the following sections with some explanatory information. A full description of XML is beyond the scope of this manual. Readers interested in exhaustive XML references should refer to Harold and Means (2004) or any of the many books and web sites dedicated to XML, such as *www.xml.com* (accessed March, 2008).

The basic unit of information in an XML file is an element. Elements are identified in the XML file by using opening and closing delimiters. Elements may contain other elements. Any content not contained in an element is a syntax error. Improperly closed or overlapping elements are the most common source of errors in writing XML files. Programmers of C, Fortran, or other procedural languages will be familiar with the behavior and rules of elements, as XML elements are syntactically similar to the block structure of these languages. Elements containing other elements give XML its hierarchical nature.

Tags are used to delimit elements and, thus, occur only as pairs. The characters < and > are used to identify tags. The first word (sequence of characters without a space) in a tag is referred to as the tag label or tag name. The first tag in the pair opens the element and is called the opening tag. The closing tag must have the same name, but has a / (slash) before the name. All of the content between the tag pair is in the element. It is possible to define an element in one line. If a / appears before the > in the line that closes the element, it is a one line element. XML parsers will check to make sure that every opening tag has a closing tag.

The opening tag of an element also may have XML attributes, which appear as assignment statements. XML attributes are used to set simple properties of the element. The assignment statement is defined by using an = (equals sign), where the left-hand side of the assignment statement is known as the attribute name and the right-hand side is the attribute value. XML attribute values are parsed as character data.

Description of project.xml

Every *project.xml* file must follow a prescribed structure (fig. 21) to work with OUI. To illustrate this, selected XML element content from the Feather River Basin *project.xml* file is annotated below. Note that the line numbers are not part of the actual XML file but are included here for reference purposes. Printed lines without line numbers are a continuation of the preceding line. The *project* element is opened at line 2 and closed at line 686:

```
2  <project name=”Feather River Project”
xmlAdapterClass=”oui.mms.MmsProjectXml”>

686 </project>
```

It contains all of the other elements and completely defines the project. It has two XML attributes:

- `name`—defines the name of the project.
- `xmlAdapterClass`—defines a Java class that is used to read and access the information in the *project.xml* file.

The *paths* element is opened at line 3 and closed at line 7:

```
3  <paths>

4   <path name=”mms_work” path=”./mms_work”/>

5   <path name=”shapes” path=”./shapes”/>

6   <path name=”log” path=”.”/>

7  </paths>
```

This element, used as a lookup table for the directory paths in the project, contains three one-line elements, each named *path*. Each *path* element has two XML attributes:

- `name`—defines the lookup name for *path*.
- `path`—defines the full directory path to substitute for the lookup name.

There are various types of tree node elements used in the OUI project tree. The *tree* element is opened at line 9 and closed at line 141:

```
9   <tree name=”Feather River Project”
logFile=”feather.log” path=”log”>

141  </tree>
```

This element contains all other tree-node elements and, thus, defines the entire content of the OUI project tree. It has three XML attributes:

- `name`—the name of the tree as displayed in the *Project Tree* window.
- `logFile`—the file name of the log file that records log messages generated by *OUI* and the Java tree node classes.
- `path`—is the lookup tag used to determine the path to the OUI log file.

The tree-node element, opened at line 10 and closed at line 24, illustrates an element that provides organizational structure to the *Project Tree* window:

```
10  <node name=”Basin Maps”>

24  </node>
```

The purpose is to hide its child nodes when the user is working in another part of the *Project Tree* window. It has one XML attribute, `name,` which is the name of the tree node as displayed in the *Project Tree* window.

The tree-node element at line 11 illustrates an element with a Java tree-node class that displays a shapefile theme in the *Map* window:

```
11   <node name=”Snow Line” desc=”Snow Line”
class=”oui.treetypes.OuiShapeTreeNode” color=”red”
theme=”snowline”/>
```

The last two XML attributes, `color` and `theme`, are input arguments to the Java tree-node class `OuiShapeTreeNode`. Tree-node elements with different Java tree-node classes may have different input XML attributes. Table 1 summarizes the available Java tree-node classes and the associated XML attributes. Table 2 describes the XML attributes and arguments that are valid for the different Java tree-node class elements.

The tree-node element, opened at line 30 and closed at line 32, illustrates an element with a Java tree-node class that displays a shapefile theme in the *Map* window and that plots time-series data from an MMS format file in the *Time Series* tool:

```
30  < name=”Measured Precipitation” desc=”Precip
Stations” class=”oui.mms.dmi.MMSDataTreeNode”
color=”orange” theme=”precip_sta” datafile=”feather_
ppt.data”/>

31   <variable variable=”precip”/>

32  </node>
```

The XML attribute `datafile` is an input argument to the Java tree-node class `MMSDataTreeNode` as described by tables 1 and 2. The other XML attributes, `name`, `desc`, `class`, `color`, and `theme` are input arguments defined by the parent Java tree-node classes `OuiTreeNode` and `Oui-DataTreeNode`, respectively. The `variable` element at line 31 also is input to `MMSDataTreeNode`, which is responsible for interpreting the content of any child elements. In this case, the `variable` element specifies the available time-series data content of MMS data file.

Lines 143 and 685 delimit the Metadata element of the *project.xml* file:

```
143  <MetaData>

685  </MetaData>
```

The content of the *MetaData* element provides lookup tables for the Java tree-node classes. The *project.xml* example has three internal elements that act as lookup tables: *themes, files,* and *mmis*. These elements are described in detail below.

The *themes* metadata element is delimited by lines 144 and 628:

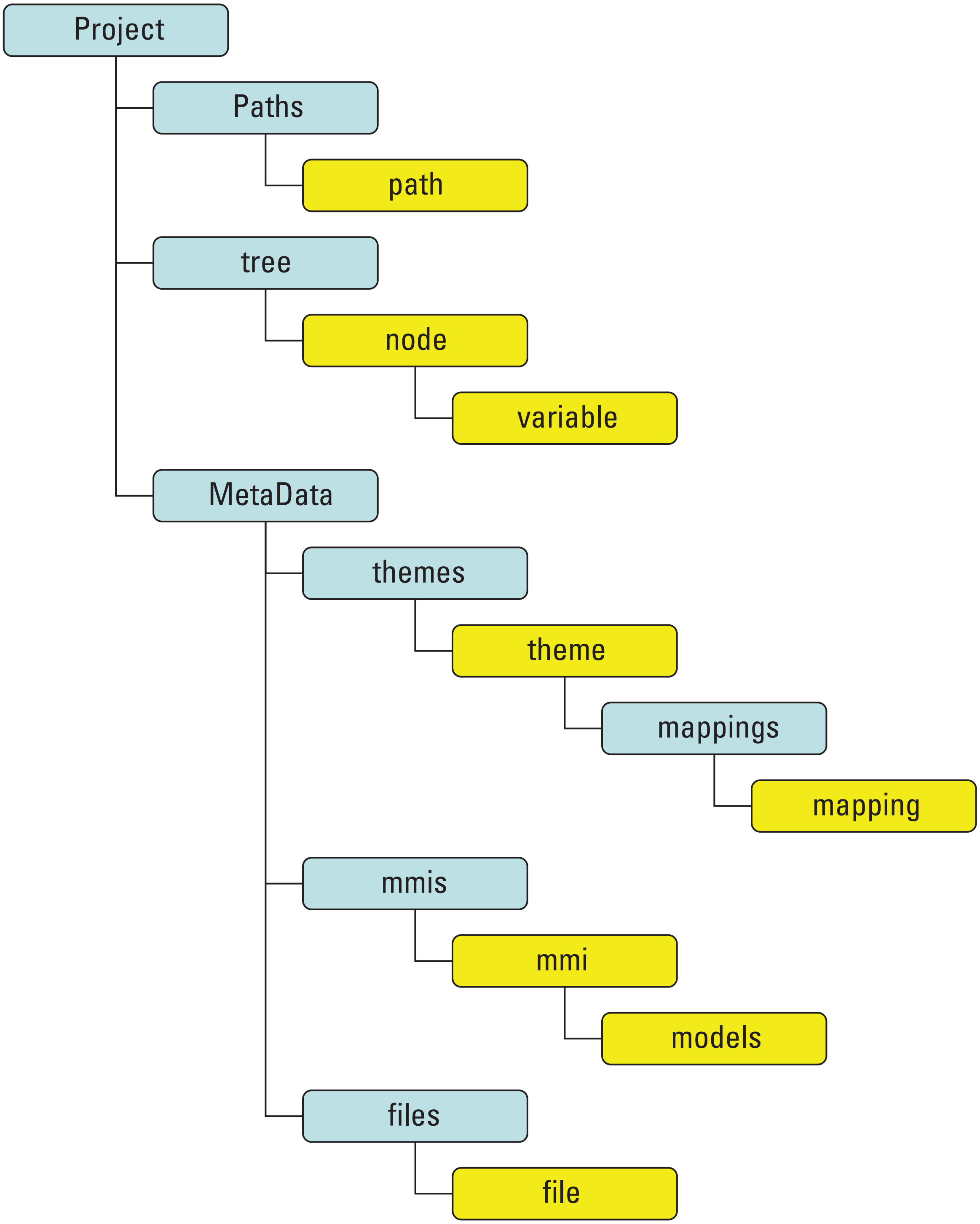


Figure 21. Structure of the Feather River OUI project tree (*project.xml*). Blue boxes represent XML elements that can occur only once, and yellow boxes represent XML elements that can occur zero or more times within the parent XML element.

Table 1. Description of Java tree-node classes implemented in Object User Interface.

[ESP, Ensemble Streamflow Prediction; MMS, Modular Modeling System; OUI, Object User Interface; XML, extensible markup language]

Class name	Description	Parent class name[1]	XML input	
			Attribute tag[2]	Metadata element[3]
	Classes from package oui.treetypes			
OuiTreeNode	Displays the tree node in the *OUI Project Tree* window		name desc class	
OuiModelTreeNode	Interface class that allows OUI to run models	OuiTreeNode		
OuiThemeTreeNode	Interface class that allows OUI to display themes	OuiTreeNode	theme	themes
OuiGridTreeNode	Displays a grid theme in the *OUI Map* window	OuiThemeTreeNode		
OuiShapeTreeNode	Displays a shapefile theme in the *OUI Map* window	OuiThemeTreeNode	color fillcolor bordercolor	
OuiDataTreeNode	Displays a shapefile theme in the *OUI Map* window and allows features to be selected from the theme and associates them with time-series or parameter data	OuiShapeTreeNode		
	Classes from package oui.mms.dmi			
MMSAnimationTreeNode	Displays the *State Variable Animator* window for the selected feature in the *OUI Map* window and displays associated time-series data as an animation	OuiDataTreeNode	demtheme featuretheme gisoutfile	files
MMSDataTreeNode	Displays the *Time Series Tool* window and loads time-series data from an MMS format input file for the selected feature in the *OUI Map* window	OuiDataTreeNode	datafile	files
MMSMultiDataTreeNode	Displays the *Time Series Tool* window and loads time-series data from multiple MMS format input files for the selected feature in the *OUI Map* window	OuiDataTreeNode	datafileExt	files
MMSMultiSeriesEspTreeNode	Displays the *ESP Tool* window and loads the ensemble and initialization period time-series output data from multiple MMS ESP simulations for the selected feature in the *OUI Map* window	OuiDataTreeNode	espSeries espInit espInitVar	files
MMSMultiStatvarTreeNode	Displays the *Time Series Tool* window and loads time-series data from multiple MMS format output files for the selected feature in the *OUI Map* window	OuiDataTreeNode	statvarExt	files
MMSOuiEspToolTreeNode	Displays the *ESP Tool* window and loads the ensemble and initialization period time-series output data from multiple OUI ESP simulations for the selected feature in the *OUI Map* window	OuiDataTreeNode	espDir espDmiClass espDmi	dmis
MMSParameterEditorTreeNode	Displays the *Parameter Editor* window for the selected feature in the *OUI Map* window	OuiModelTreeNode	parameterfile defaultparamfile	files

Table 1. Description of the Java tree-node classes implemented in Object User Interface.—Continued

[ESP, Ensemble Streamflow Prediction; MMS, Modular Modeling System; OUI, Object User Interface; XML, extensible markup language]

Class name	Description	Parent class name[1]	XML input	
			Attribute tag[2]	Metadata element[3]
MMSSingleStatvarTreeNode	Displays the *Time Series Tool* window and loads time-series data from a single MMS format output file for the selected feature in the *OUI Map* window	OuiDataTreeNode	statvarfile	files
	Selected Classes from package oui.mms.mmi			
CommandTreeNode	Runs a command as a system call submitted to the operating system	OuiModelTreeNode	command	
MmsModelTreeNode	Interface class that allows OUI to run MMS models	OuiModelTreeNode		
MmfOuiMultiBasinEspRunTreeNode	Runs ESP for an MMS format model with one input data file	MmsModelTreeNode	mmi dataFileName espVariableName espVariableIndex initLength espIODir executable controlFileExt paramFileExt	mmis files
MmfOuiMultiDataFileEspRun TreeNode	Runs ESP for an MMS format model with multiple input data files	MmsModelTreeNode	mmi dataFileName espVariableName espVariableIndex initLength espIODir executable controlFileExt paramFileExt	mmis files
MmfOuiMultiBasinRunTreeNode	Runs an MMS format model with a single input data file	MmsModelTreeNode	mmi dataFileName variableName variableIndex executable controlFileExt paramFileExt	mmis files

[1] Java tree-node classes retain all of the XML attributes and metadata of their parents.

[2] XML attribute tags and values are described in table 2.

[3] XML metadata elements are described in the OUI Project Configuration section.

Table 2. Description of XML attributes for Java tree-node classes presented in table 1.

[DEM, digital elevation model; DMI, data management interface; ESP, Ensemble Streamflow Prediction; GIS, geographic information system; MMI, model management interface; MMS, Modular Modeling System; OUI, Object User Interface]

Attribute tag	Description of attribute value	Default value[1]	Valid values
name	Name of tree node in the *OUI Project Tree* window	Blank string	
desc	Description of tree node in the *OUI Project Tree* window	Blank string	
class	Java tree-node class	OuiTreeNode	Any Java class that extends OuiTreeNode
theme	Lookup value for the theme in the themes metadata table		Valid theme tag
fillcolor	Color used to fill the body of a polygon feature	clear	Valid OUI color[2]
bordercolor	Color used to draw a point or the outline polygon feature	black	Valid OUI color[2]
color	Color used for both the fill color and border color	clear	Valid OUI color[2]
demtheme	Lookup value for the DEM theme in the themes metadata table		Valid DEM theme tag
featuretheme	Lookup value for the shapefile theme in the themes metadata table		Valid shapefile theme tag
gisoutfile	Lookup value for the MMS format GIS output file in the files metadata table		Valid gisoutfile file tag
mmi	Lookup value for the MMS model in the mmis metadata table		Valid mmi tag
datafile	Lookup value for the MMS format input file in the files metadata table		Valid datafile file tag
datafileExt	Extension used to determine the data file name(s)		
espSeries	Model output variable name used for ESP runs		Valid model variable name
espInit	Lookup value for the MMS format statvar file that contains the model output data corresponding to the ESP initialization period		Valid statvar file tag
espInitVar	Model output variable name used for ESP initialization period		Valid model variable name
statvarExt	Extension used to determine the statvar file name(s)		
espDir	Directory containing the ESP output files		Valid directory
espDmiClass	Runs a DMI that reformats model output into application specific data base		Java classfile
espDmi	Lookup value for the MMS model in the mmis metadata table		Valid MMI tag

[1] Default value is used if the XML attribute is not set in the project.xml file.

[2] Valid OUI colors are: black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white, yellow, and clear

```
144  <themes>

153   <theme theme=”hill100” path=”shapes” file_
name=”hill100.dat/>

628  </themes>
```

All *theme* elements, as illustrated by line 153, have three XML attributes. A more complicated theme element is illustrated by lines 245 through 254:

```
245  <theme theme=”bc_hrus” path=”shapes”
file_name=”bc_hrus” idAttribute=”HRU”
nameAttribute=”HRU”>

246   <mappings>

247     <mapping modelId=”1” modelName=””
themeId=”1”/>

248     <mapping modelId=”2” modelName=””
themeId=”2”/>

249     <mapping modelId=”3” modelName=””
themeId=”3”/>

250     <mapping modelId=”4” modelName=””
themeId=”4”/>

251     <mapping modelId=”5” modelName=””
themeId=”5”/>

252     <mapping modelId=”6” modelName=””
themeId=”6”/>

253   </mappings>

254  </theme>
```

A shapefile theme element may include a *mappings* element, as illustrated by lines 246 through 253. The *mapping* elements provide a lookup table for the Java tree-node classes to relate unique features in the theme to model data files or specific time series contained in a model data file. The XML attributes in a *mapping* element are described in table 3. Although a particular theme may be declared only once, a specific shapefile may be referenced by multiple themes. This feature allows the modeler to map a specific shapefile differently for each use.

The *mmis* metadata element is delimited by lines 630 and 640:

```
630  <mmis>

631   <mmi mmi=”single”>

632     <models controlFileExt=”.xprms.
control” dataFileExt=”_draper_climateQ.
data” executable=”models/xprms” name=”prms”
paramFileExt=”_feather.params” path=”mms_work”
theme=”prmsSubbasinSeeds” variableIndex=”1”
variableName=”basin_cfs.strmflow”/>

634   </mmi>

640  </mmis>
```

All *mmi* elements, as illustrated by line 631 through line 634, have a lookup attribute tag used to find this element. The contained *models* element (line 632) defines the information necessary for a Java tree-node class to reference, execute, and process the models. Valid attributes for *models* elements are described in table 4.

The *files metadata element* is delimited by lines 82 and 87:

Table 3. Description of XML attributes for themes element metadata.

[GIS, geographic information system; ID, identifier]

Attribute tag	Description of attribute value	Valid values
theme	Lookup tag used to find this *theme* element	
path	Lookup tag used to determine the path to the directory containing the file with the theme data	Valid path tag
file_name	Name of the theme file relative to the directory path	
nameAttribute	Column name in the GIS attribute table where feature names are stored	
idAttribute	Column name in the GIS attribute table where the feature IDs are stored	
	Mappings element metadata	
themeId	Lookup tag used to find this *mapping element* where the value is matched from the ID column of the theme GIS attribute table for a feature in the theme	
modelName	Model name to associate with the feature where the attribute relates a feature to a specific parameter file, data file, control file, or output file	
modelId	Position of a time-series variable to associate with the feature where the attribute relates a feature to a specific variable in a file	
modelOrder	Order to run the model associated with the feature	

```
        <files>
                <file type=”parameterfile” name=”AC_feather.params” path=”mms_work” file=”input/params/AC_feather.params”/>
                <file type=”statvarfile” name=”AC.statvar” path=”mms_work” file=”output/AC.statvar”/>
                <file type=”controlFile” name=”AC.xprms.control” path=”mms_work” file=”control/AC.xprms.control”/> 86 <file type=”datafile” name=”ac_draper_climateQ.data” path=”mms_work” file=”input/data/AC_draper_climateQ.data”>
        </files>
```

The *files element* contains *file elements*. Each *file element* is referenced by *type* and *name*. It contains the information necessary for a *TreeType* class to reference and process a file. All *file elements* have four fundamental attributes described in table 5. *File elements* may optionally include additional elements that describe file contents (not shown in the *project.xml* example). Adding elements is typically done for DMIs and is described in the *Extension* section of this manual.

OUI Project Extension

An OUI project can accommodate project-specific needs through extension. This adaptability is demonstrated below with two examples that extend standard OUI Java tree-node classes. These examples assume a good working knowledge of the Java programming language.

Extension of Java Tree Node Class OuiDataTreeNode

An example of a new time-series data reader is `MMS-DataTreeNode`, which reads a time-series data set from an MMS format data file for a feature selected from the *Map* window and displays the values in the *Times Series* tool.

```
1   public class MMSDataTreeNode extends OuiDataTreeNode {
```

At line 1, the new class extends `OuiDataTreeNode`, which provides much of the functionality of this class through inheritance.

```
10  public MMSDataTreeNode(Node xml_node) {
11    super(xml_node);
12    type = “Shape w/MMS Data File
13    pxml = MmsProjectXml.getMmsProjectXml();
14    dataFileTag = pxml.getElementContent(xml_node, “@datafile”, “set your datafile!”);
15    variableNames = pxml.getDmiVariableNames(_xml_node);
16  }
```

The constructor at lines 10–16 reads XML attribute information into the class variables `dataFileTag` and `variableNames` by using the `MmsProjectXml` convenience class.

```
23  public void queryData(Feature feature) {
24    String stationName = getStationNameForFeature(feature);
26    int variableIndex = -1;
27    try {
28      variableIndex = Integer.decode(pxml.getModelIdForThemeId(_theme_name, stationName)).intValue();
29    } catch (NumberFormatException ex) {
30      System.out.println(“Bad value for modelId tag. Theme name = “ + _theme_name + “ station name = “ + stationName);
31    }
32    if(variableIndex == -1) return; // No data for selected item
```

The method `queryData`, defined at line 23, implements the abstract method defined in `OuiDataTreeNode`. This method is called by OUI when a feature is selected in the *Map* window. The input argument `feature` is an object that represents this selected feature. Lines 23 through 31 determine the lookup identifier (`variableIndex`) of the feature (in this case the column corresponding to the observation station) in the time-series data file. If the file does not contain any data for this feature, the method returns at line 32.

```
38    JPopupMenu analysis_popup = new JPopupMenu();
39    analysis_popup.add(new JLabel(“Which variable for “ + stationName + “?”));
```

Table 4. Description of XML attributes for Model Management Interface element metadata.

Attribute tag	Description of attribute value	Valid values
name	Lookup tag used to find this *models element*	
path	Lookup tag used to determine the path to the directory containing the model	Valid path tag
executable	Name of the executable relative to the directory path	
theme	Theme and associated mappings to use for the model execution	Valid theme tag
dataFile	File tag for input data file	Valid file tag
dataFileExt	Data file name extension that is combined with the `modelName` attribute specified in the theme mappings element to assign the data file path	
dataFileHeader	Name of the data file header file to use for creation of intermediate data files	
paramFile	Input parameter file name	Valid file tag
paramFileExt	Model parameter file name extension that is combined with the `modelName` attribute specified in the theme mappings element to assign the parameter file path	
controlFile	Input control file name	Valid file tag
controlFileExt	Control file name extension that is combined with the `modelName` attribute specified in the theme mappings element to assign the control file path	
envFile	Environment file name	Valid file tag
variableName	Output time series variable name	
variableIndex	Output time series variable index	

Table 5. Description of XML attributes for files element metadata.

Attribute tag	Description of attribute value	Valid values
type	Category of the file	statvarfile parameterfile defaultparamfile controlfile datafile gisoutfile
name	Lookup tag used to find this *file element*	
path	Lookup tag used to determine the path to the directory containing the file	Valid path tag
file	Name of the file located in the *path* directory	

```
40   analysis_popup.addSeparator();

42   if (variableNames != null) {

43     for (int i = 0; i < variableNames.length;
i++) {

44      analysis_popup.add(new
JMenuItem(variableNames[i])).addActionListener(new A
nalysisActionListener(stationName, variableIndex));

45     }

46     analysis_popup.show(OuiGui.getOuiGisPanel(),
100, 100);

47   }

48  }
```

Lines 38 through 48 display a pop-up menu, which contains the names of the data that are available from the data file.

```
50  private class AnalysisActionListener implements
ActionListener {

51   private String stationName;

52   private int variableIndex;

54   public AnalysisActionListener(String
stationName, int variableIndex) {

55     this.stationName = stationName;

56     this.variableIndex = variableIndex;

57   }

59   public void actionPerformed(ActionEvent e) {

60     String variableName = e.getActionCommand();

62     MmsDataFileReader dataFileReader = new
MmsDataFileReader(pxml.getFileFullPath(“datafile”,
dataFileTag));

63     TimeSeriesCookie tsc = dataFileReader.
getTimeSeries(variableName + “ “ + variableIndex);

65     TimeSeriesTool tst = TimeSeriesTool.
createTimeSereiesTool();

66     tsc.setName(variableName + “ at “ +
stationName);

67     tst.addTrace(tsc);

68   }

69  }
```

The method actionPerformed in the inner class AnalysisActionListener at line 59 is called when the user selects a variable name from the pop-up menu. The assignment at line 60 gets the selected variable name. Lines 62–67 read the time-series data by using the variable name and index as the lookup key and by passing it into to the *Time Series* Tool.

Extension of Java Tree Node Class OuiModelTreeNode

An example of an MMI is MmsSingleBasinRun, which runs a single MMS model.

```
1  public class MmsSingleBasinRun extends
OuiModelTreeNode implements
MmsSingleModelRunner {
```

At line 1, the new class extends OuiModelTreeNode, which provides much of the functionality of this class through inheritance. The interface MmsSingleModelRunner allows this class to create a custom GUI and provide a well-defined method to call when the model is ready to run. This feature is described in more detail below.

```
12  public void run() {

13   mmi = pxml.getElementContent(_xml_node, “@mmi”,
“none”);

15   String dataFile = pxml.getMmiFile(mmi, “@
data”);

16   dataFilePath = pxml.getFileFullPath(“datafile”,
dataFile);

18   MmsDataFileReader mdfr = new MmsDataFileReader(
dataFilePath);
19   ModelDateTime data_file_start = mdfr.getStart();

20   ModelDateTime data_file_end = mdfr.getEnd();

22   new MmsSingleBasinRunGui(dataFilePath, data_file_start, data_file_end, this).show();

23  }
```

The `run` method definition at line 12 overrides the abstract definition in `OuiModelTreeNode`. This method is called by OUI when the user selects the *Run* option on the node pop-up menu in the *Project Tree* window. The assignment at line 13 calls an element of *XML* from the *project.xml* file, which describes the model input and output in detail. Lines 15 and 16 get the path to the data file. The assignment at line 18 creates the class that reads the time-series data from the specified data file. Lines 18 and 19 get the start and end time of the time-series data in the data file. This information is passed into the object, which creates a simple GUI (`MmsSingleBasin-RunGui`) for running the model at line 22.

```
25  public void cleanup() {}

26  public void declare() {}

27  public void initialize() {}
```

The empty method definitions `cleanup`, `declare`, and `initialize` (lines 25 through 27) also override abstract definitions in `OuiModelTreeNode`. These methods are not used in `MmsSingleBasinRun` but are called by OUI.

```
29  public void runModel(ModelDateTime queryStart, ModelDateTime queryEnd) {

30   String envFile = pxml.getMmiFile(mmi, “@env”);

31   String env = “ -E” + pxml.getFileFullPath(“envfile”, envFile);

32   String control = “ -C” + pxml.getMmiFile(mmi, “@control”);

33   String statvar = “ -set stat_var_file “ + pxml.getMmiFile(mmi, “@statvar”);

34   String paramFile = pxml.getMmiFile(mmi, “@param”);

35   String param = “ -set param_file “ + pxml.getFileFullPath(“parameterfile”, paramFile);

36   String data = “ -set data_file “ + dataFilePath;

37   String executable = pxml.getMmiPath (mmi) + “/” + pxml.getMmiFile(mmi, “@executable”);

38   String runoption = “ “ + pxml.getElementContent(_xml_node, “@options”, “”);

39   String start_time = “ -set start_time “ + queryStart.getControlFileDateTime();

40   String end_time = “ -set end_time “ + queryEnd.getControlFileDateTime();

41   String arg = executable + “ -batch” + env + control + data + param + start_time + end_time + statvar + “ -set statsON_OFF 1 “ + runoption;

42   System.out.println(“MmsRun: executing = “ + arg);

44   try {

45     new CommandRunner(arg).evaluate();

46     JOptionPane.showMessageDialog(OuiGui.getTreeScrollPane(), “Run Completed”, “Run Status”, JOptionPane.INFORMATION_MESSAGE);

47   } catch (IOException e) {

48     JOptionPane.showMessageDialog(OuiGui.getTreeScrollPane(), “Run Unsuccessful\n” + e.getMessage(), “Run Status”, JOptionPane.ERROR_MESSAGE);

49   }

50  }
```

The runModel method definition at line 29 implements the interface MmsSingleModelRunner. This method is used so that the GUI defined by MmsSingleBasinRunGui can be reused by other classes that extend OuiModelTreeNode for similar purposes. The assignments at lines 30 through 41 are preparing the command line argument that will run the model. Much of this information comes from the MMI XML element that was determined at line 13. The try and catch elements (line 44 through 49) run the model as a system call. The OUI utility class CommandRunner (line 45) submits the command line to the operating system. It manages the execution thread, which can either wait for the thread to finish or run concurrently. It also manages the standard input, output, and error streams. Lines 46 and 48 will pop up a dialog window indicating the final run status.

Summary

This report introduces OUI as a map-based framework that provides a common interface for running models, as well as acquiring, browsing, organizing, displaying, and selecting spatial and temporal data. Instructions are provided for system installation and manipulation of the various windows of the graphical user interface. Several modeling and data visualization tools associated with OUI also are described. A successful OUI application in the Feather River Basin, California, is included as an example of how to set up a complex modeling project. Finally, examples of how users can programmatically configure and extend OUI are presented.

References Cited

Day, G.N., 1985, Extended streamflow forecasting using NWSRFS: American Society of Civil Engineers, Journal of Water Resources Planning and Management, v. 111, no. 2, p. 157–170.

Harold, E.R., and Means, W.S., 2004, XML in a nutshell, a desktop quick reference (3d ed.): Sebastopol, Calif., O’Reilly Media, Inc., 689 p.

Jeton, A.E., 1999, Precipitation-runoff simulations for the upper part of the Truckee River Basin, California and Nevada: U.S. Geological Survey Water-Resources Investigations Report 99–4282, 41 p.

Koczot, K.M., Jeton, A.E., McGurk, B., and Dettinger, M.D., 2005, Precipitation-runoff processes in the Feather River Basin, northeastern California, with prospects for streamflow predictability, water years 1971–97: U.S. Geological Survey Scientific Investigations Report 2004–5202, 82 p.

Leavesley, G.H., Lichty, R.W., Troutman, B.M., and Saindon, L.G., 1983, Precipitation-runoff modeling system—User’s manual: U.S. Geological Survey Water-Resources Investigation Report 83–4238, 207 p.

Leavesley, G.H., Markstrom, S.L., Brewer, M.S., and Viger, R.J., 1996a, The Modular Modeling System (MMS)—The physical process modeling component of a database-centered decision support system for water and power management: Water, Air, and Soil Pollution, v. 90, p. 303–311.

Leavesley, G.H., Markstrom, S.L., Viger, R.J., and Hay, L.E., 2005, USGS Modular Modeling System (MMS)—Precipitation-runoff modeling system (PRMS) MMS–PRMS, *in* Singh, V., and Frevert, D., eds., Watershed models: Boca Raton, Fla., CRC Press, p. 159–177.

Leavesley, G.H., Restrepo, P.J., Markstrom, S.L., Dixon, M., and Stannard, L.G., 1996b, The Modular Modeling System (MMS) user’s manual: U.S. Geological Survey Open-File Report 96–151, 175.

Risley, J.C., 1994, Use of a precipitation-runoff model for simulating effects of forest management on stream flow in 11 small drainage basins, Oregon Coast Range: U.S. Geological Survey Water-Resources Investigations Report 93–4181, 61 p.

U.S. Geological Survey, Object user interface (OUI), *http://water.usgs.gov/lookup/get?crresearch/oui*. Accessed March 2008.

Glossary

A

abstract class An abstract class is a Java class that defines the necessary methods and data of any of its subclasses. The subclasses are responsible for implementing the contents. OUI uses abstract classes to enforce a minimum set of functionality in all DMIs and MMIs.

absolute paths An absolute path is a path specified from the root directory of the file system that points to a specific location regardless of the working directory.

abstract method An abstract method is a method definition in an abstract class. This method is implemented only in a subclass.

active theme Themes are made active by turning on the *Active* check box in the *Loaded Themes* window. This check box indicates the theme in the *Map* window that can be tracked by cursor movement. Using the cursor, the user can display information about features of the active theme in the tracking box of the *Map* window. Only one theme can be active at a time.

attribute See GIS attribute or XML attribute.

B

branch Branches are one of two node types that appear in the *Project Tree* window. Branches are nodes that have subnodes. They can be opened by clicking on them with the mouse. A branch also may have a theme associated with it.

class A class is a set of methods and data that operate together to define an object. Specifically, any Java code within curly braces and declared with the class keyword is a class.

D

data management interface (DMI) DMIs are Java classes that interface the OUI to site-specific data sources. Some standard DMIs are provided in the OUI system library. Users can write custom DMIs by extending the abstract OUI management interface classes.

dbf file dbf stands for dBASE file format. This is one of the files that collectively make up a shapefile. The dbf contains the GIS attribute information for the shapefile.

digital elevation model (DEM) A DEM file is a raster spatial data format containing elevation data.

E

element An element is the basic unit of information in an XML file. Elements are identified in a XML file by delimiter "tags" that open and close the element. Elements may contain other elements.

ensemble streamflow prediction (ESP) ESP is a modeling methodology developed by the National Weather Service (Day, 1985), which generates an ensemble of possible outcomes based on current conditions and historic driving variables.

extensible markup language (XML) XML is a meta-markup language that provides grammar and validation for development of application-specific markup languages. The *project.xml* file implements a markup language specifically for OUI.

feature A feature is the smallest selectable unit of a theme that is stored in OUI as a vector shapefile. Features of shapefile themes and the related GIS attribute data may be displayed and queried in OUI as points, lines, or polygons. For example, each stream segment in the stream shapefile theme is a line feature.

G

GIS attribute A GIS attribute is a field assigned to the attribute table of a shapefile. The attribute table is stored in dBASE file format (dbf). Data about the shapefile features are stored in the attribute field. For example, for each stream segment in a stream shapefile, a length measurement is stored. Stream length is an attribute of the stream shapefile, and the measurement is the attribute data associated with a particular stream.

J

Java Java is an object-oriented, interpreted, and portable programming language developed by Sun Microsystems.

Javadoc Javadoc is the tool for generating documentation in HTML format from structure and comments in source code developed by Sun Microsystems.

K

key A key is an attribute in a theme that is used to look up data associated with a feature.

I

label A label is an attribute in a theme that is used for displaying additional information about a feature. Usually, the label is the name that humans use to describe the feature.

M

method A method is a function or subroutine in a Java class.

MMS input file An MMS input file is an ASCII file containing time-series data that conforms to the MMS standard.

MMS statvar file A MMS statvar file is an ASCII output file containing time-series data that has been produced by MMS.

MMS timeseries file see MMS input file

MMS work space (mms_work) An MMS work space is a set of directories created specifically for running MMS models. A work space contains input and output data, parameters, executable models, source code, and other MMS support files.

model A model is an executable piece of computer software, along with all associated data used to simulate physical processes.

model management interface (MMI) MMIs are Java classes that interface OUI to specific models. Some standard MMIs are provided in the OUI system library for MMS. Users can write custom MMIs by extending the abstract OUI management interface classes.

modular modeling system (MMS) MMS is a modeling framework used for multidisciplinary research and operational applications developed by the U.S. Geological Survey. (*http://water.usgs.gov/lookup/get?crresearch/mms*, accessed March, 2008)

N

native programming language or native code A native programming language is a language, like C or Fortran, which is compiled into an executable that will run only on the targeted computer platform.

O

object An object is an instance of a class.

object user interface (OUI) OUI is a map-based interface for models and modeling data developed by the U.S. Geological Survey. (*http://water.usgs.gov/lookup/get?crresearch/oui*, accessed March, 2008)

P

package A Java package is a way to group associated classes together. Also, the CLASSPATH environment variable and the package are used to uniquely identify classes.

precipitation-runoff modeling system (PRMS) PRMS is a model that has been developed to evaluate the effects of various combinations of precipitation, climate, and land use on surface-water runoff, sediment yields, and general basin hydrology. PRMS was developed by the U.S. Geological Survey. (*http://water.usgs.gov/lookup/get?crresearch/prms*, accessed March, 2008)

project Unless otherwise, noted the term "project" refers to an OUI project. An OUI project contains a unique *project.xml* file, associated models, data, and DMIs/MMIs.

project tree The project tree defines how the data files, executable programs, and simulation models are organized within the OUI project. It is defined by the *project.xml* file and visualized by the *Project Tree* window in the GUI.

R

relative path A relative path is a path specified by the relation to the current working directory.

S

shapefile A shapefile is a vector data format used to digitally store the location, shape, and attributes of geographic features. Geographic features may be stored as points, lines, or polygons. A shapefile comprises a set of related files, contains one type of feature class, and includes a dBASE file format (dbf) file that holds descriptive information about individual features.

shp file One of the files that collectively make up a shapefile. The shp file contains the coordinate information for the shapefile.

statvar file see MMS statvar file

T

tag Tags are used to delimit elements in XML files and occur only as pairs. The characters < and > are used to delimit tags. The first word (sequence of characters without a space) in a tag is referred to as the tag label or tag name. The first tag in the pair opens the element and is called the opening tag. The closing tag must have the same name, but has a / (slash) before the name. All of the content between the tag pair is in the element. It is possible to define an element in one line. If a / appears before the > in the line that opens the element, it is a one-line element.

theme A theme is a spatial data set of unique geographic features stored in OUI. Themes may be shapefiles containing vector feature data stored as points, lines, or polygons. Themes also contain the related feature GIS attribute information. Themes also may be stored in ASCII raster formats. Further, OUI reads in MMS-configured data tables. These themes may be linked to the shapefile feature data and made part of a unique theme. For example, a map of stream gages and the related simulated streamflow data is considered a theme.

tree node Tree nodes are used to build the OUI project tree. A top-level project tree node must be defined for the project. All other tree nodes are contained within this project tree node. Also, any tree node can contain other tree nodes. The most basic tree node in OUI is defined by the OuiTreeNode Java class. This class is used to add branches to a project tree or it can be extended to add additional functionality.

X

XML see Extensible Markup Language

XML attributes XML attributes are used to set simple properties of the *element*. These are designated in *opening tags* and will have *XML attribute descriptions known as "assignment statements."* The assignment statement is defined by using an = (equals sign), where the left-hand side of the assignment statement is known as the *attribute name* and the right-hand side is the *attribute value.* The *XML Attribute values* may be character or numeric data or may call other elements.

Publishing support provided by:
Denver Publishing Service Center

For more information concerning this publication contact:
Chief, Office of Surface Water
U.S. Geological Survey
411 National Center
Reston, VA 20192
(703) 648-5629

The latest version of the program and this report can be obtained using the Internet address: http://water.usgs.gov/software

Published in the Central Region, Denver, Colo.
Manuscript approved for publication, March 24, 2008
Graphics by author
Edited by Mary-Margaret Coates (Contractor, ATA Services)
Layout by Sharon M. Powers

Markstrom and Koczot—**User's Manual for Object User Interface (OUI)**—Open-File Report 2008–1120

Printed on recycled paper