Mission Research Corporation

EOSAEL *For Windows*™
FINAL REPORT

25 OCTOBER 95

PREPARED BY
MODELING, SIMULATION AND DATA MANAGEMENT DIVISION
MISSION RESEARCH CORPORATION
ONE TARA BOULEVARD
SUITE 302
NASHUA, NH 03062

PREPARED IN SUPPORT OF
SBIR A94-086
ENVIRONMENTAL INTERACTION FOR EOSAEL
(ELECTRO-OPTICAL ORIENTED SYSTEMS ATMOSPHERIC EFFECTS LIBRARY)

CONTRACT NUMBER DAAL01-95-C-2010
CDRL ITEM A003
MRC-NSH-TR-95-0007

EOSAEL Final Report

25 October 1995

Mission Research Corporation
One Tara Blvd. Suite 302
Nashua, NH 03062

Accesion For		
NTIS	CRA&I	☑
DTIC	TAB	☐
Unannounced		☐
Justification		
By		
Distribution /		
Availability Codes		
Dist	Avail and / or Special	
A-1		

Preface

This technical report was developed under contract to the U.S. Army Research Laboratory in support of SBIR A94-086, Environmental Support for EOSAEL (Environmentally Oriented Systems Atmospheric Effects Library).

The technical points of contact for this report are Dr. Paul Noah, or Ms. Meg Noah, MRC Nashua, NH (603) 891-0070. Questions and requests for further information on this report should be addressed to:

Director
U.S. Army Research Laboratory
AMSRL-BE-S (Dr. Alan Wetmore)
White Sands Missile Range, NM
88002-5501

Prepared by:

Mission Research Corporation
One Tara Boulevard
Suite 302
Nashua, NH 03062-2809

TABLE OF CONTENTS

LIST OF FIGURES

1. Executive Summary

1.1 EOSAEL for Windows – Improved EOSAEL Library Usability

The Electro-Optical Systems Atmospheric Effects Library (EOSAEL) is a state of the art computer library which describes atmospheric propagation in several wavelength regimes for the battlefield environment. The EOSAEL modules provide transmittance and radiance calculations through gases, natural aerosols, battlefield aerosols, smoke, haze, fog, and clouds for bandpass and laser propagation. The library consists of twenty-four FORTRAN codes which act fairly independently.

Since the library was developed over time by many different people, each code tends to show the developer's method of thinking. This means that each module has its own "feel" as to input stream, required input information, and, being FORTRAN, the formatting or method of data input varies from model to model.

A single interface system is provided in EOSAEL for Windows which can present the model input requirements in a format understandable to the user, not the code developer. A single or common GUI (Graphical User Interface) type of interface has the added advantage of presenting a single, logical stream avoiding duplication if the same information is needed in multiple modules. An additional GUI utility for manipulation and display of output data is also provided.

The operating system of choice is *Microsoft Windows 3.1*™, a graphical display operating system which gives a *common interface* to hardware. Therefore, a developer for Windows develops applications and does not need to delve into different hardware problems. EOSAEL for Windows has graphical capabilities and ease of use understandable by most PC users since Windows use is so widespread.

This document describes MRC's effort in developing an Expert System assisted Graphical User Interface (GUI) under a Small Business Innovation Research (SBIR) Phase I effort to develop EOSAEL for Windows The Phase I effort involves the integration of the EOSAEL library module COMBIC, the battlefield obscurant module, into EOSAEL for Windows. The design developed and implemented by MRC is scaleable to a full EOSAEL.

1.2 Design and Development Philosophy

MRC's approach was to design and develop an expert-assisted, Graphical User Interface (GUI) for problem definition and data entry. The developed system is a highly extensible software architecture, designed to provide the basis for managing the execution of the codes under the Windows environment, extensive data analysis tools, and management of simulation data inputs and outputs. Our system architecture allows inclusion of additional atmospheric models and will provide much greater accessibility and utility to the EOSAEL codes.

The following chapters detail the methods used to develop MRC's EOSAEL for Windows. To use EOSAEL for Windows, the reader should refer to the User's Guide provided with the software.

1.3 User Benefits from EOSAEL for Windows

This introductory section has briefly discussed the top level architecture of the MRC EOSAEL for Windows System. Key benefits of the MRC innovative design and software delivered under this SBIR effort are:

- The design implements an off-the-shelf expert system which monitors and guides the user through an EOSAEL application.
- The prototype delivers a GUI that streamlines application of COMBIC through the following features:
  - *Windows style toolbar*
  - *Data entry screen which presents the features of the COMBIC control cards familiar to COMBIC users*
  - *A set of descriptive tools which link to the Expert System*
  - *A set of functional tools (Exit, Save, Run, and output analysis) which perform stated functions.*
- The Phase I Prototype contains a Hypertext HELP System that "links" with the embedded expert system and provides detailed assistance to users as they are guided through an application.
- The design philosophy maintains the EOSAEL Scientific Codes in their native format which allows model changes to be transparent to the EOSAEL for Windows architecture.
- The modular design presented allows for ease of maintenance, adaptation of new features, growth of the EOSAEL Codes, and provides for growth and rule updates to the embedded expert system.
- The prototype design, software, and Hypertext Help are fully scaleable to the entire suite of models that comprise the EOSAEL code library.
- The success of Phase I has demonstrated the feasibility of Phase II and Phase III and has greatly reduced the technical risk associated with these phases.
- A market survey conducted as part of this effort has validated the need to upgrade the EOSAEL Library by the addition of a user friendly GUI.
- Update of the EOSAEL Library in Phase II and III will protect the long term investment in these models by ensuring their ease of application in future weapon systems design, operations research, and battlefield/war game simulations.

2. EOSAEL for Windows Architecture

2.1 System Overview

The EOSAEL for Windows system architecture follows that specified in the MRC Phase I proposal and is shown in Figure 1. The primary system structure is that of a Multiple Document Interface (MDI) and mirrors current industry trends for MDI as the standard for system design. The design provides the user with access to various EOSAEL tasks via a top level menu. The addition of an embedded expert system guides the user through processing tasks via child windows launched from the Main Interface.

Descriptions of the event driven features of the system are provided in the sections that follow. Components discussed are: the Main Interface; the Expert System; the GUI for Defining Code Input; the GUI for Analysis of Code Output; the Common Parameter Definition (CPD) Input File; the Common Output Data Object; and the On-line Help. Although shown in the figure, the Code Library is not discussed, since the EOSAEL Models are well documented in other manuals.

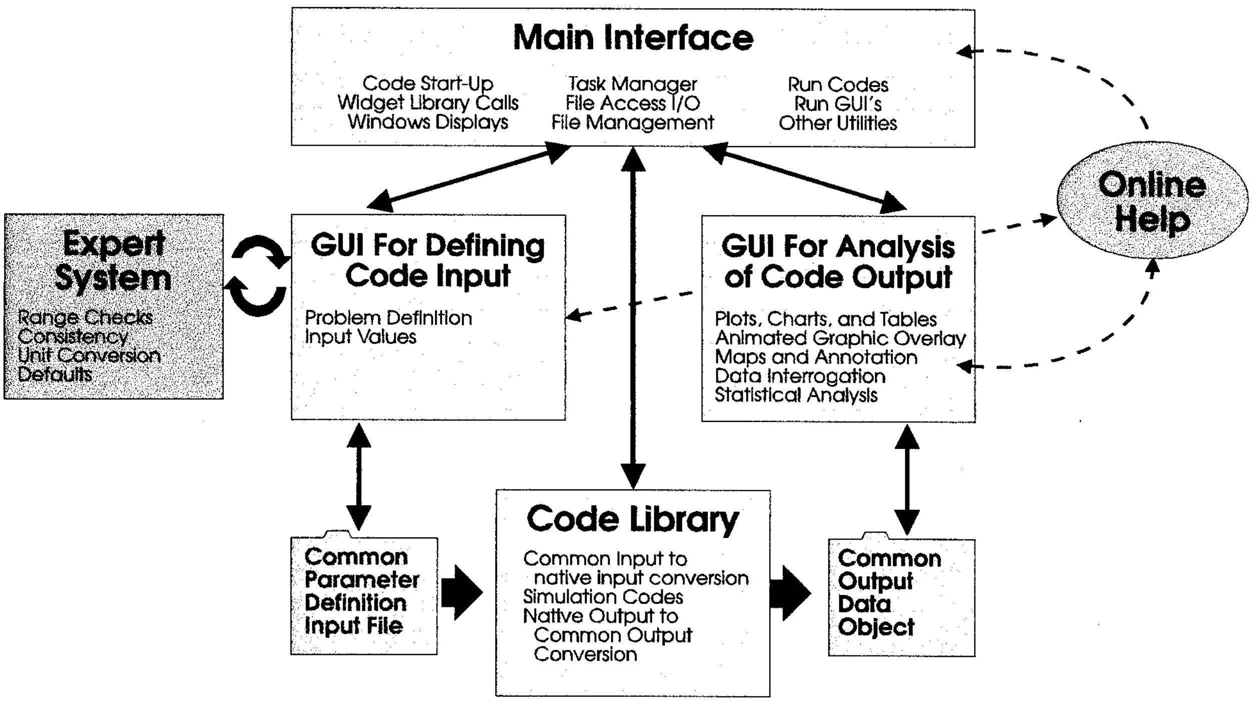

Figure 2-1: The MRC EOSAEL for Windows System Architecture

2.2 EOSAEL Architecture Components

2.2.1 Main Interface

The Main Interface provides all top level menus and serves as the container form for child windows and dialogs that lead the user through the particular EOSAEL application. This MDI architecture gives the EOSAEL system the feel of a typical commercial grade Windows program. This interface provides access to the following event driven activities: Code Start-Up; Widget Library Calls, Windows Displays, Task Manager, File Access I/O; File Management; various system utilities; Run GUIs; and Run Codes.

2.2.2 The C Language Integrated Production System (CLIPS) Expert System

The design feature that makes the MRC design unique and ensures proper EOSAEL model application is the CLIPS expert system shell tailored specifically for EOSAEL. CLIPS, developed by the NASA Johnson Space Center, is written in ANSI C. The EOSAEL software design has CLIPS as a standalone library in order to provide modularity. Our object-oriented integration of CLIPS is independent of other Windows functions, such as widget routines. Running in a "background mode", CLIPS monitors and guides the user through the EOSAEL application by providing external function calls which can be made to load rules, assert facts, read facts, and run application rules for correct use of the EOSAEL application. These calls are performed by the CLIPS Manager Class library. As an embedded Artificial Intelligence (AI) application, CLIPS is within the scope of the GUI for Defining Code Input and warns the user before launching an EOSAEL Code with inconsistent or incorrect inputs and thus prevents misusing an EOSAEL function. The CLIPS system does not prevent the user from "making a mistake". It only warns that the inputs are a violation of a known set of rules.

The architecture for the embedded expert system is provided in Figure 2. The components are:

- The GUI Interface Library;
- The CLIPS Manager;
- The CLIPS Library; and,
- The Scenario Definition Manager and the CPD Manager.

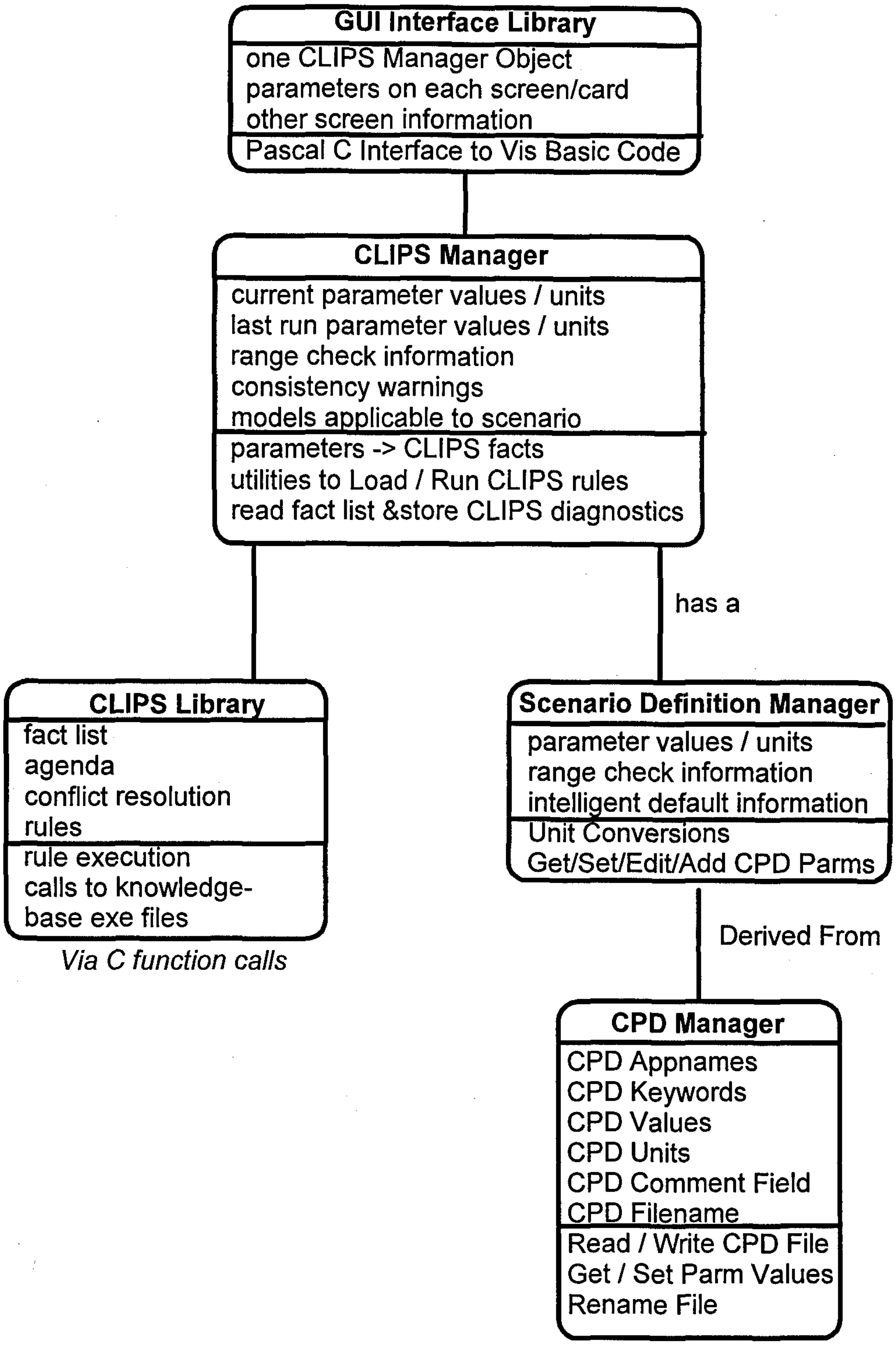

Figure 2-2: The Embedded CLIPS and GUI Interface Architecture

2.2.2.1 The GUI Interface Library

The primary functions of the GUI Interface Library are:

1. Provides interface between the GUI and the CLIPS Manager,
2. Maintains standard defaults,
3. Handles GUI related information (e.g., a map of screen prompts to CPD names, Help indices, etc.),
4. Writes the COMBIC input file, and
5. Adds "cards" to the CPD.

It maintains the state of the current system and provides a PASCAL C interface to the Visual Basic GUI code. It returns to the GUI the parameters required to develop EOSAEL model input (previously a set of card image, 80 byte records) with their values, units, and comment fields, and other system information required by CLIPS. When the user defines a parameter value from the GUI, the new information is passed to the CLIPS Manager through this interface.

This module also interacts with the Common Input Parameter File and the EOSAEL Code Library which provides access to the Common Input Data Interpreter (CIDI) and the Common Output Format Generator modules contained in the CODE Library. This module also takes the user generated inputs and translates them into code specific native-format input data sets. This additional format process is part of the design philosophy to maintain the EOSAEL Codes in their native state to the maximum possible extent and ensures modularity and ease of maintenance of the EOSAEL for Windows interfaces to the native model codes.

The advantage of this architecture is that it is easy to redesign the GUI and maintain all knowledge functionality.

2.2.2.2 The CLIPS Manager Class

The CLIPS Manager Class contains three Scenario Definition Manager Classes. The first describes the current values of the CPD that are being edited; the second houses intelligent defaults based on the user's scenario; and the third houses the last run values. The CLIPS Manager provides the interface to the CLIPS expert system and provides the function calls to assert facts, load rules, run rules, and read facts. The CLIPS manager also stores current expert system results such as consistency warnings and the name(s) of the EOSAEL Codes applicable to the input scenario.

2.2.2.3 The CLIPS Library

The CLIPS Library contains conflict resolution rules and implements rule execution by carrying out system calls to the knowledge base executable (*.exe) files.

2.2.2.4 The Scenario Definition Manager Class and CPD Manager

The Scenario Definition Manager Class is derived from the CPD Manager Class and additionally houses the expert system and GUI related information for each CPD parameter. For example, range check status and error messages, and intelligent defaults are maintained for each parameter.

2.2.3 GUI for Defining Code Input

This object provides the top level interface through which the user develops all of the required inputs to initiate an EOSAEL model run. The primary utility and power of this Task Manager child window are derived from its interaction with the CLIPS Expert System as it monitors user inputs to the system.

2.2.4 GUI for Analysis of Code Output

For Phase I, the GUI primarily defines the functionality for the Phase II analysis module. This initial functionality will minimally consist of text viewers such as Microsoft Windows Notepad™ to allow perusal of tabulated output. The full Phase II functionality will consist of a menu driven suite of analysis functions such as plots, charts, tables, animated graphics overlays, maps, statistical analysis, and data interrogation.

2.2.5 The CPD Input File

This module allows the user to define a Common Parameter Definition (CPD) input file which defines the superset of parameters required to run any of the EOSAEL models. It also allows the user to define any other additional parameters required to define an EOSAEL scenario and to select which EOSAEL model should be run and the format of the output data. Our innovative approach centers on the development and implementation of common data objects for storage of input values.

The CPD file parameters are user-specified, defaulted, expert-system derived, or obtained from other databases and resources. CPDs are reviewed and edited via the GUI for Defining Code Input which is a child window of the Main Interface form.

2.2.6 Common Output Data Object

Within the EOSAEL Code Library, each model has a code specific Common Output Format Generator module that converts the native-format model output to the formats required by the EOSAEL for Windows analysis routines. This additional format process is part of the design philosophy to maintain the EOSAEL Codes in their native state to the maximum possible extent and ensures modularity and ease of maintenance of the EOSAEL for Windows interfaces to the native model codes.

2.2.7 On Line Help

This function has the look and feel of the typical Windows hyper-text help and is accessible to the user through the scope of the primary GUIs (the Main Interface, GUI for Defining Code Input, the GUI for Analysis of Code Output, and through the Expert System. For Phase I, Help is tailored to the COMBIC Code and will be derived from the COMBIC User's Guide. The On Line Help also contains necessary system level guidance to assist the user with system level features and operations.

Help is available for:

- Each model parameter. Accessed by pressing the F1 function key when that parameter is in focus,
- Each card, and
- Specific sections from the Help option of the main menu.

3. EOSAEL for Windows User Interfaces

This section briefly discusses and provides an example of the types of screens encountered in the user interface. Additional details are available in the User's Guide and in the On-Line Help.

3.1 Overview

The MRC-developed EOSAEL for Windows interface has four main components

1) Showcard – EOSAEL User-friendly input for COMBIC that interacts with the embedded expert system.
2) EZ Entry – Card deck/input development tool that interacts with the embedded expert system. This works for COMBIC and will also apply to all of the other EOSAEL models under Phase II.
3) Graphical data reviewer, GUI-OUT and,
4) COMBIC Help System, EOHELP (described in section 4 of this report)

These are shown below in Figure 3-1.

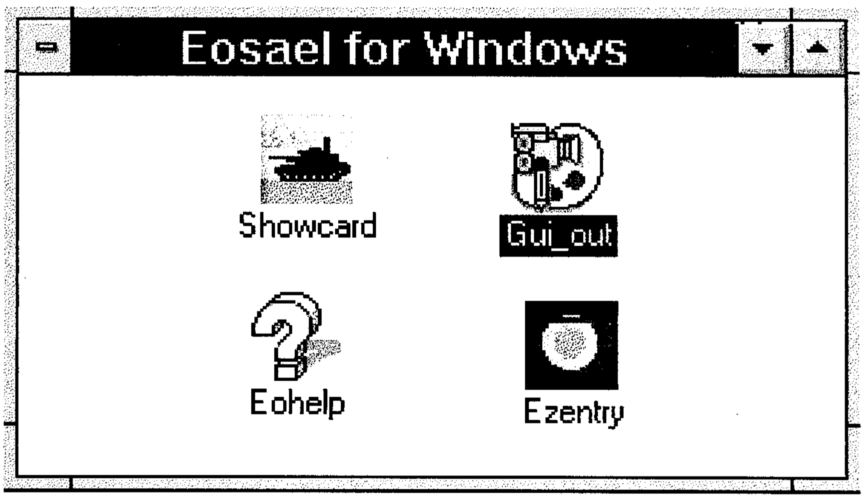

Figure 3-1: EOSAEL for Windows group

3.2 EOSAEL User Interface

The EOSAEL User Interface, EZ Entry, is a scenario level view of EOSAEL. Once the Phase II development is completed, it will be the link between all of the EOSAEL modules. The user builds a scenario with multiple EOSAEL components. Presently, four components can be specified:

1) Observer,
2) Intervening Atmosphere,
3) Target, and
4) Background.

The user uses drags and drops the selected components to be used. It is presented in Phase I as a prototype of the Phase II system. Further evaluation by ARL at the start of Phase II is encouraged.

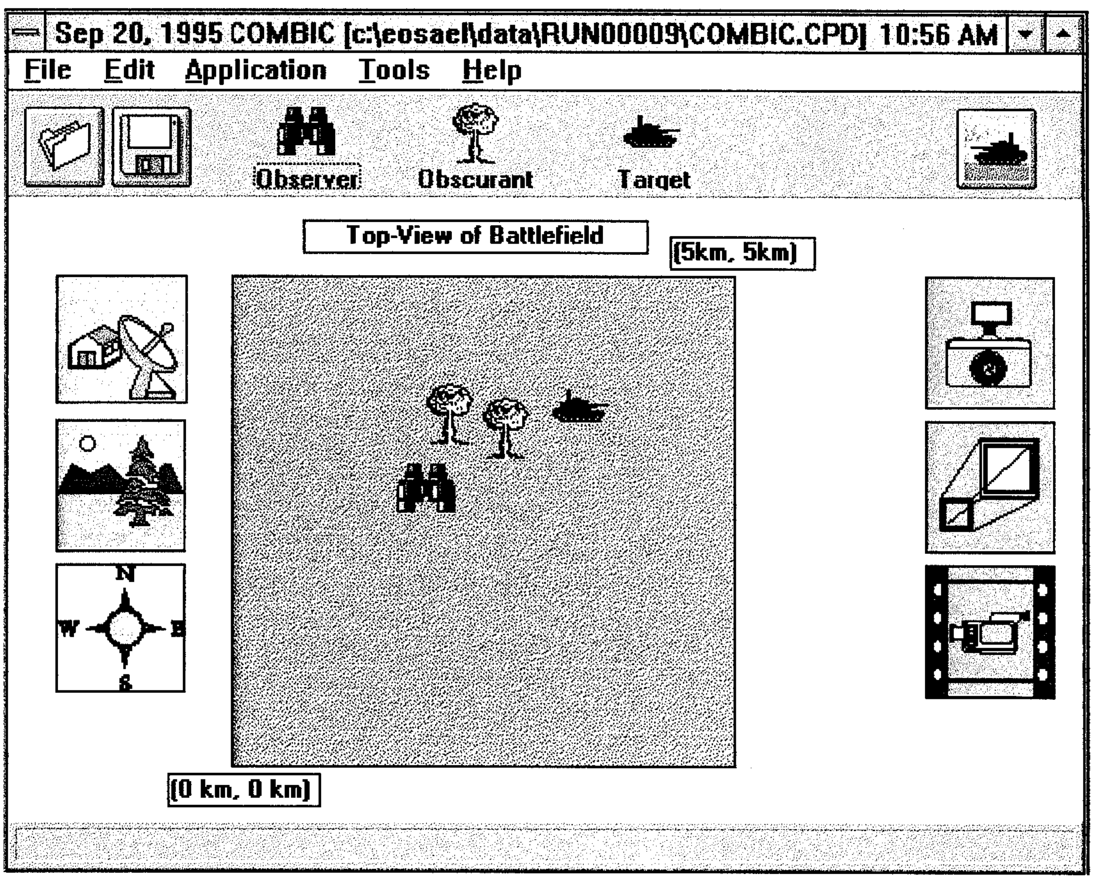

Figure 3-2: Screen dump of EZ Entry User Interface

The complete description of the EZ Entry User Interface is presented in the EOSAEL for Windows User's Guide, a separate document included with this report.

3.3 COMBIC Card Review Screens

The COMBIC input card review screens are intended for an intermediate or expert user. These screens are entered through the Showcard Icon discussed previously in Section 3.1. Showcard is intended for use by an experienced COMBIC user. A sample of one of the Showcard screens is shown in Figure 3-4.

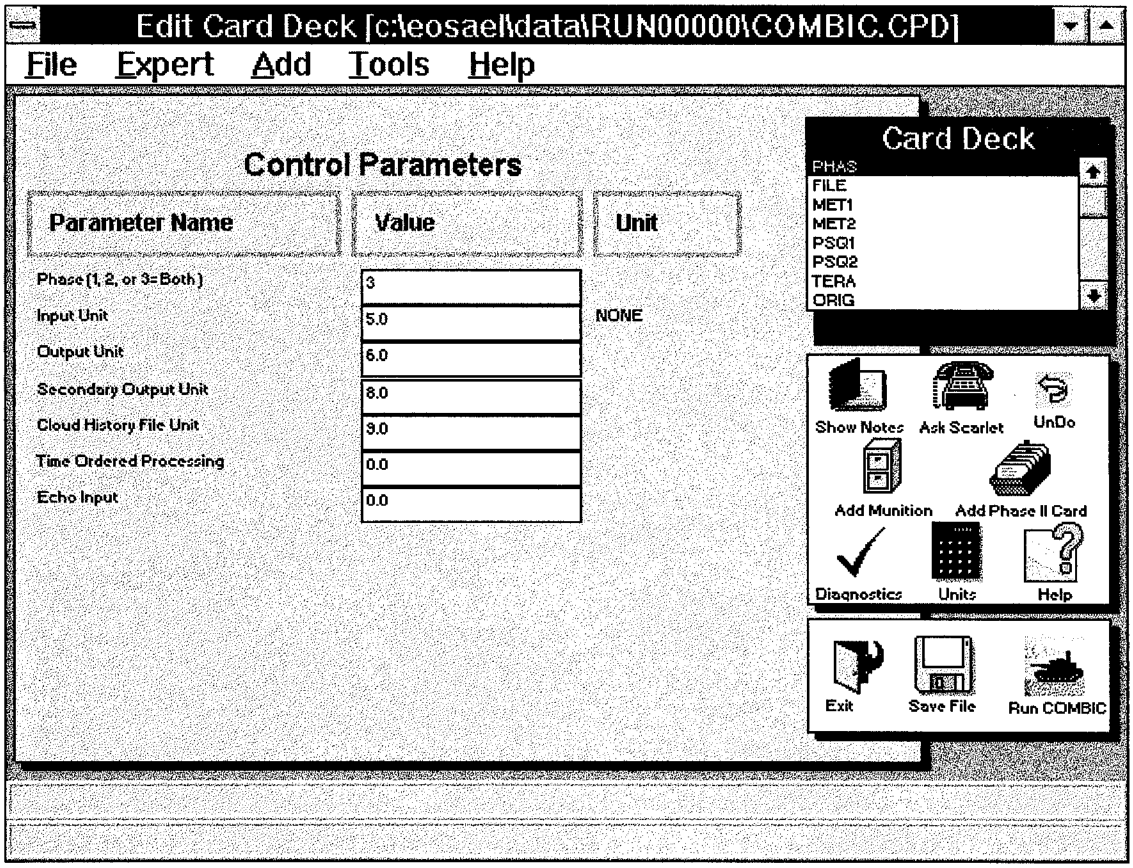

Figure 3-4: Sample COMBIC Showcard Screen.

The basic elements of the Showcard interface are:

1. The Windows style toolbar,
2. The data entry screen which presents the features of the COMBIC control cards familiar to COMBIC users,
3. Indicators of card selection,
4. A set of descriptive tools which link to the Expert System, and
5. A set of functional tools (Exit, Save, Run) which perform stated functions.

The complete description of the COMBIC Showcard Input GUI is presented in the EOSAEL for Windows User's Guide, a separate document included with this report.

4. EOSAEL for Windows Help

This section briefly describes the MRC Hypertext Help system that has been built into this prototype. This Help system is fully extensible to a full EOSAEL.

4.1 Introduction

Hypertext is a highly general, user oriented approach to textual information access employed in "Help" functions by many new commercial software products. It is the help system used by Microsoft Windows™ and it has also started migrating to the X/UNIX world. Hypertext relies on the concept of multiply-cross-linked key words to achieve user-chosen navigation through text. The underlying textual database can be arbitrarily large and is easily edited and expandable to include other EOSAEL models

Hypertext "Help" can be thought of as a super-set of the familiar context-sensitive "Help" facility. It allows the user to decide both how deeply to browse the available text and also the sequence in which to read specific sections. Hypertext type help systems are developed using Microsoft™ Visual Basic.

As part of the EOSAEL user interface, the MRC design implements a hypertext help facility which can be accessed in several familiar ways: through a "hot" key, selecting Help on the main menu, "clicking" on the description of a datum entry slot, and normal *Hypertext* browsing.

For Phase II, the fully developed EOSAEL's hypertext database design will incorporate multi-level descriptions of the embedded expert system application, all supported codes, the code input requirements, the code output products, and the technical terminology employed in descriptions of phenomenologies and their parameters.

4.2 On-Line Documentation

On-line documentation is commonly employed for several purposes:

- To provide Hypertext help in using the software, i.e. An on-line User Manual
- To provide warning and error messages when inconsistencies are found
- To educate novice users by providing tutorials

For the full scale EOSAEL, our on-line documentation will also provide additional information and explanation of error messages or requests generated by the expert system as it guides the user through an application.

4.3 Hypertext

4.3.1 Hypertext

As discussed before, hypertext is a highly general, usage-oriented approach to textual and graphical information access which is being employed for implementation of on-line documentation by most new commercial software productions. Hypertext is an information system that provides links (jumps or branches) between units of information (nodes or components) to achieve user-chosen navigation through text and images. Hypertext "Help" can be thought of as a super-set of the familiar context-sensitive "Help" facility. It allows the user to decide both how deeply to browse the available text and also the sequence in which to read specific sections. The underlying information database can be arbitrarily large and is easily expandable and editable. This database is usually organized hierarchically, allowing the user to easily travel from one topic to another remotely connected topic. The topic can include both text or graphics. The Hypertext design concept is illustrated in Figure 4-1.

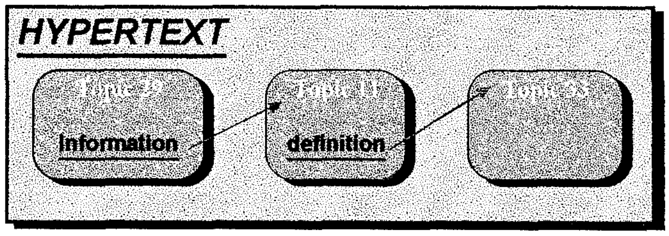

Figure 4-1: Concept of Hypertext.

4.3.2 Hypertext Navigation

One common problem faced by the Hypertext user is getting "lost" while navigating. To help alleviate this problem, the EOSAEL Hypertext system design:

- Provides a clear picture of the network structure, e.g. Table of contents;
- Provides a history of previously-viewed topics;
- Makes topics accessible via keyword search;
- Follows a guideline of maximum of four levels to reach a topic.

Our Hypertext Help design supports the "Search" and "History" functions. Other functions included are "Contents" (shows Table of Contents), "Back" (steps to previous topic), "Browse" (sequentially steps through related topics), and "Index" (shows index topics).

The basis for the information system in the Phase I EOSAEL Hypertext is the COMBIC User's Guide. That is, the EOSAEL end-user will have access to the COMBIC User's Guide while running EOSAEL for Windows The advantages of both types of help media are:

- Since the User's Guide design is well organized, the user is less likely to become "lost" while navigating.
- When the user gets tired of reading the on-line topic, he can easily find the section in the printed User's Guide.

- If the user has previously scanned the User's Guide, he has a general idea of the information network structure and will feel familiar and comfortable with the on-line help.

To assist the user, the topic that contains additional information is generally highlighted in some way. In Windows Help and in our design, the topic which provides the link is underlined and appears in green on a color monitor. When the mouse is moved on top on this object, the cursor changes from an arrow icon to a hand icon. An example from the EOSAEL on-line help is shown in Figure 4-2. This help is also accessible from the Program Manager by double-clicking on the EOHELP Icon.

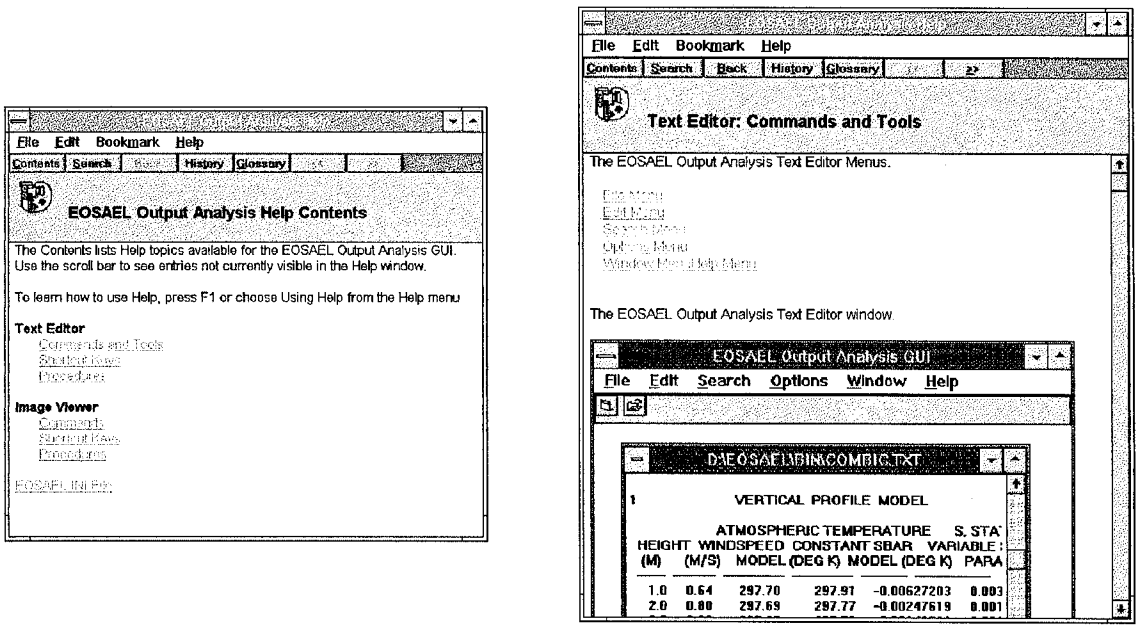

Figure 4-2: Illustration of the jump feature in Windows On-line Help.

The MRC Help design incorporates state-of-the-art features and elements of Hypertext. Development support tools are available in the Microsoft™ Windows Software Development Kit, the newest "C++" language packages, and in Microsoft™ Visual Basic.

4.3.3 Tutorials

For Phase II, our help system will include the use of tutorials. The objectives of tutorials are to introduce new users to the product, acquaint them to the main functionalities, and educate them about the use of the product. Tutorials supplement the User's Guide. Tutorials usually provide a set of friendly tasks for the new or novice users to learn the product. As an example, the COMBIC Showcard interface provides a simple walk through providing system defaults. This is shown in Figure 4-3.

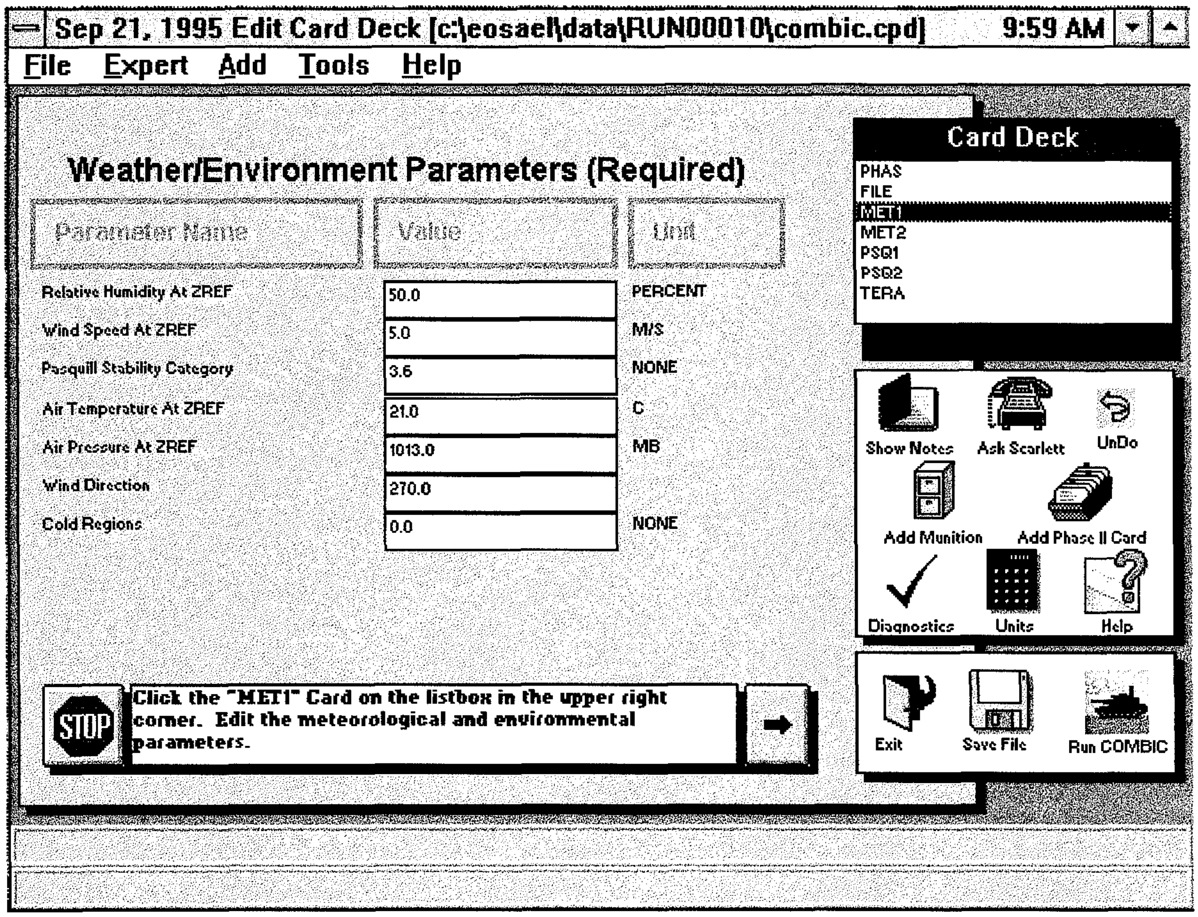

Figure 4-3: The COMBIC Interface Tutorial Sample Screens.

Well-designed tutorials will be of high value for first-time or novice EOSAEL users. Table 1 shows some useful tutorial topics directed at users with various backgrounds.

Table 1: Phase II EOSAEL Tutorial Topics.

Tutorial Topic	*Type of User*
Introduction to EOSAEL Platform Requirements	new EOSAEL users
Introduction to Graphical User Interface (GUI) Controls	new EOSAEL GUI users
Introduction to EOSAEL models	new EOSAEL users

4.3.4 Messages

Status Bar messages are also designed into our help/tutorial windows. In response to user action, on-line messages appear to give directions, warn of inappropriate actions or values, provide useful information, etc. Since this form of on-line documentation is highly interactive with the user and almost appears conversational, it is very important to design the message well so that the statement serves the intended function without offending the user. The figure below shows the result of entering an incorrect value to the Air Pressure parameter. The offending value is highlighted in red and an error message appears at the bottom of the screen. The user is not prevented from going on to additional screens, since this is just a warning that the value entered exceeds the normal input range. Since COMBIC and EOSAEL are dynamic codes, a user may be developing enhancements to the codes, and EOSAEL for Windows allows for this development as long as no new input parameters are defined.

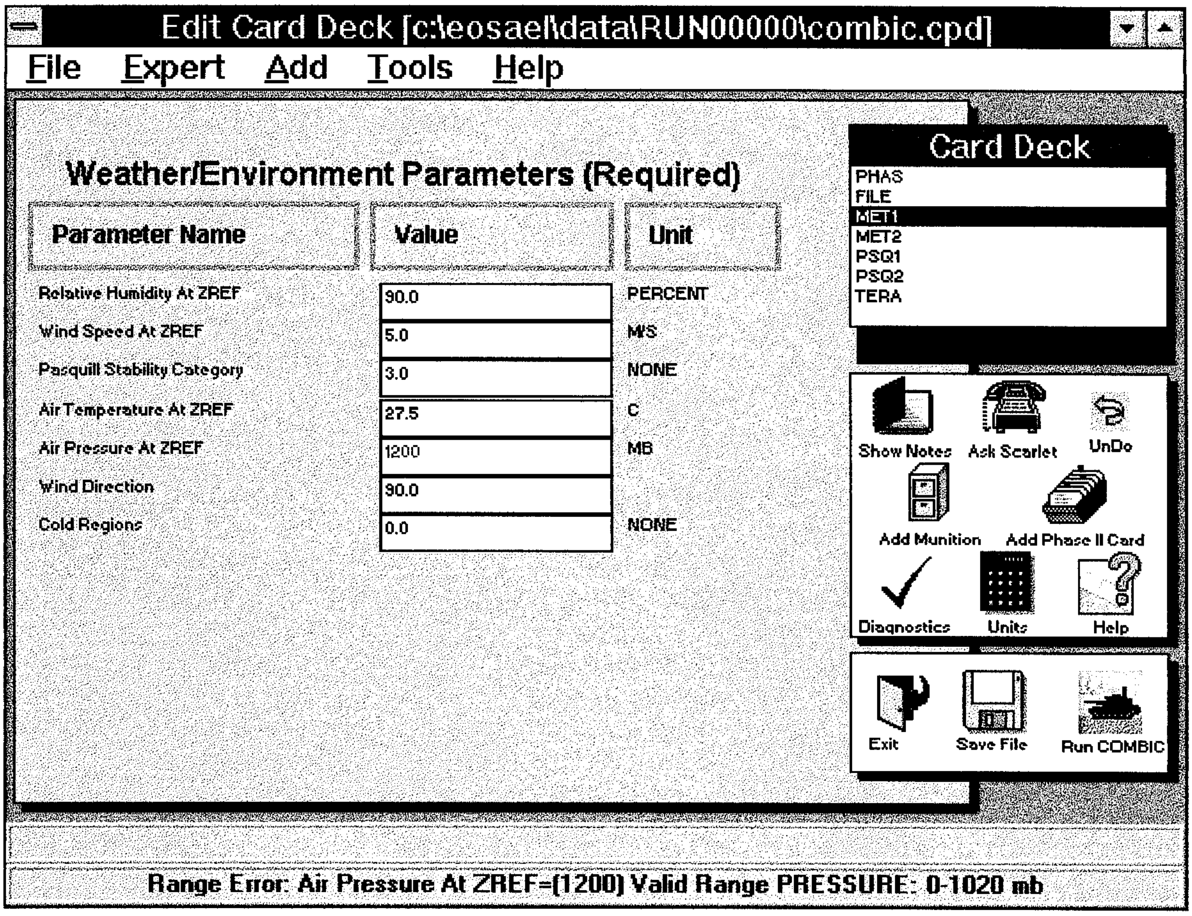

Figure 4-4: The COMBIC Status Bar Help Message for Toolbar.

Status bars and dialog boxes are two common controls used for sending messages. As shown in the example from COMBIC, the system provides a short statement of the function of the Toolbar button on a status bar when the mouse is over that button.

As another example, the expert system coupled to the COMBIC interface provides warnings when a user has input an out-of-range value or when an input value causes inconsistencies with another parameter. A short informative statement first appears in a dialog box. If the user needs further information clicking on the Help button causes a file containing more explanation to appear. This is demonstrated in Figure 4-5.

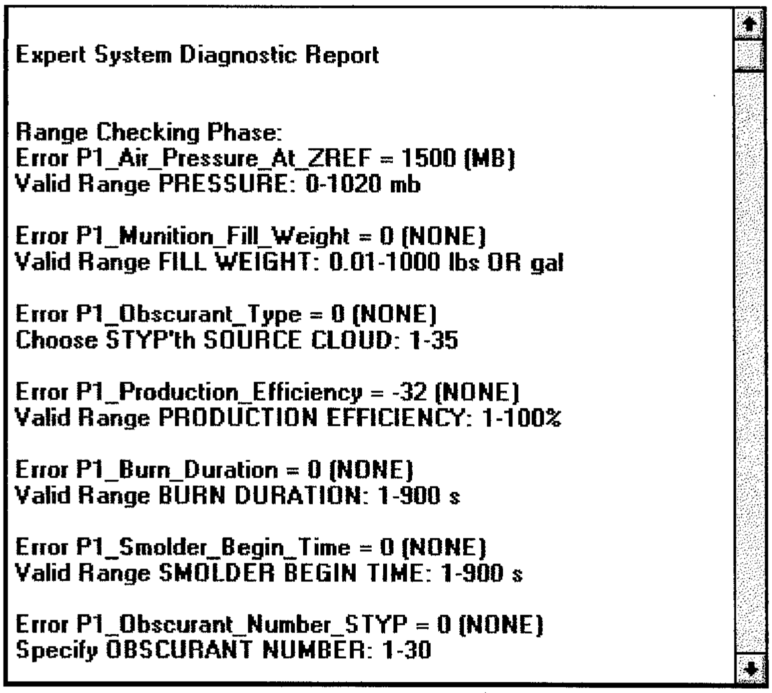

Figure 4-5: Interaction of the Embedded Expert System and On-Line Help.

4.4 Summary

The design of the EOSAEL HELP System for Phase I and Phase II has the following key features that enhance user friendliness and user application of EOSAEL:

- The HELP System "links" with the embedded expert system and provides more detailed assistance to users as they are guided through an application.
- The design is based on well known hypertext systems which give the HELP system the feel of a commercial grade software package.
- The HELP System design supports both hardcopy and softcopy access to HELP information as would be found in a commercial package.
- The HELP System design allows ease of maintenance and provides for scalability to a full EOSAEL package.

5. EOSAEL for Windows Expert System

5.1 Overview

5.1.1 Scope of the EOSAEL for Windows Expert System

In its broadest definition, an expert system is a computer system that emulates the decision-making ability of a human expert. Like human experts, expert systems are designed to provide solutions for specific problem domains, as opposed to general problem-solving techniques. Since the 1980's, expert systems applications have been successfully developed for many fields: scientific, medical, business, among others. The goals in developing an expert system are to provide complete, accurate, and consistent solutions for a well-defined application.

A knowledge-based expert system is a software/data product that emulates the problem solving expertise of a human expert within a particular domain. As an example, conventional, algorithmic style of coding is error-prone and cumbersome for solving decision graphs. Computer scientists consider expert system technology a pragmatic attempt for using innovative pattern matching and conflict resolution routines to perform such logic.

More specifically, an expert system consists of a domain specific knowledge base and the expert system shell, complete with inference engine and utilities for constructing the knowledge base. Figure 5-1 illustrates the conceptual modules of an expert system architecture.

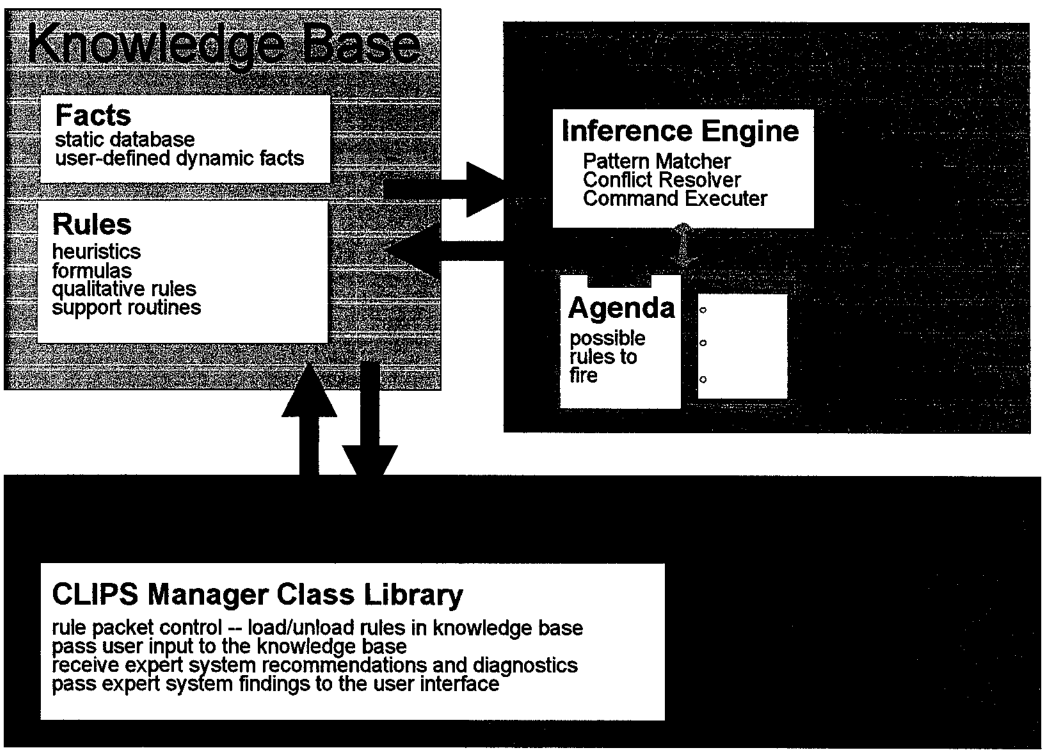

Figure 5-1: Expert System Conceptual Modules.

EOSAEL for Windows imposes additional requirements on the expert system architecture including modularity, portability, maintainability, and extensibility. In addition, the ability to easily interface to the expert system is essential. The architecture of the knowledge base should be properly designed for transfer to the expert system shell and support libraries. Finally, the system must be inexpensive to distribute.

For EOSAEL for Windows MRC has successfully implemented an expert system with objectives of assisting various levels of users in solving atmospheric problems. The users may be novices, narrow-topic experts, intermediate, or experts in the fields. The knowledge base is based on first principles, encompassing fundamental and advanced knowledge in the essential domains. The expert system is coupled to the problem definition interface acting as an advisor making suggestions, reporting inconsistencies, deriving model input parameters, and offering an optimal solution for a user's problem. Final model input is checked by the user who has options for overriding the expert system's advice.

The requirements for developing the EOSAEL for Windows expert system are:

- Cost-effective expansion and enhancement of EOSAEL models,
- Compensation for user error or inexperience,
- Provide reasonable advice/input defaults.

The EOSAEL for Windows expert system has the following functions:

- **Focus on user's problem**

  Coupling an expert system to the problem definition interface can provide the user an environment whereby he can concentrate on his problem and its solution rather than having to figure out which code is appropriate for his problem and going through manuals to determine the input parameters and how to run a code.

- **Minimize queries**

  Based on a user's initial inputs, the expert system can be used to determine the additional input parameters to be queried. Using the expert system provides an efficient mechanism to eliminate inconsequential questions. For example, if the user's problem is for a night time case, then we would not ask any questions related to the Sun. For novices, we will create default values for parameters unfamiliar to them.

- **Perform range and consistency checks**

  An expert system can dynamically check whether the user input values are within the proper ranges of the code and whether the parameters are consistent with other inputs.

- **Offer instant solution**

  In Phase II, the expert system will determine from user inputs whether a user problem matches a case in the library of pre-calculated results. The user can instantly view an approximate answer. After viewing the pre-calculated results, the user can still choose to continue running the phenomenology models. This feature is a real time-saver since some model calculations are extremely time-consuming.

- **Choose best model**

An important function of the expert system in Phase II is to determine the best code(s) to solve the user's problem. For the novice user, this is performed transparently. For advanced users, a list of possible codes and commentary information is provided.

5.1.2 Expert System Architecture

Expert system shells range from small public domain versions to large copyrighted software packages costing many thousands of dollars. These offer diverse methods of inferencing, knowledge base organization, and procedures for embedding the knowledge base within a larger software product. Several features must be critically evaluated for EOSAEL for Windows. These are:

5.1.2.1 Knowledge Base

The heart of an expert system is a knowledge base for a domain, i.e., a special purpose database. Since EOSAEL for Windows requires volume-intensive database information (e.g., climatology, radiosonde, etc.), it is important that the knowledge base be organized flexibly. Knowledge bases can be written using a variety of designs. The design of the knowledge base is specific to the features of the shell being used. However, the underlying content of any knowledge base can be represented by facts and rules.

5.1.2.2 Facts

Expert systems run by evaluating *facts*. Facts are used, destroyed, and created by the user and the expert system shell. Facts consist of a key name with one or more attributes. Facts can be static (e.g., "The ground altitude of Denver Colorado is 5,280 ft above Sea Level.") or dynamic (e.g., "The current air pressure is 750 Torr."). Static facts are normally incorporated into the knowledge base to aid in range and consistency checking. Dynamic facts not only represent the current user's session but also hold internal bookkeeping data to run the expert system.

5.1.2.3 Rules

The most common form of expert system is *rule*-based. Rules are expressions that take the form of "if-then" statements. If the expression in the left hand side (antecedent or condition) is true, then the right hand side (consequent or action) executes. The antecedents have patterns that attempt to match the fact sets.

5.1.2.4 Packets

If one considers individual rules to be simple and small pieces of knowledge, then many rules may need to be written to perform a specific task. Groups of rules can be created to obtain a specific goal. These groupings are termed rule modules or *packets*. Rule packets offer many advantages. They allow the knowledge programmer to debug the knowledge base to a greater degree by isolating the inputs and outputs to the packet. Packets increase efficiency by allowing the inference engine to focus on a small subset of rules. Finally, packets are themselves a form of knowledge representation since the individual rules share a common goal.

5.1.2.5 Inference Engine

The *inference engine* performs the primary processing in the expert system shell. In the inference engine, pattern matching occurs between the current facts and the antecedents of the rules. The Rete Pattern Matching algorithm [Forgy, 1982] is commonly used since it takes advantage of the temporal redundancy exhibited by rule-based systems.

When a match occurs, it is placed on the *agenda.* The agenda is the list of rules that may fire. The inference engine manages the agenda. The expert system community has developed many different agenda management schemes including breadth and depth searches depending on a thoroughness/performance tradeoff. *Conflict resolution* is the process by which a rule is selected from the agenda to actually execute or *fire*. Firing can involve fact destruction(s), fact creation(s), function call(s) among many other actions. After the rule has fired, the agenda is updated and process repeats itself until either the agenda is empty or the system is halted.

Facts can be combined with some organization into *frames.* The implementation of frames varies from groups analogous to records in common imperative languages, to mechanisms that incorporate inheritance. For example, the manager frame can inherit characteristics from the employee frame. A frame system is useful when the knowledge to be incorporated has a large database of static facts. Many rules can be implicitly written as a single rule that references a row of data from frames in its antecedent.

5.1.2.6 Methodology

Object-oriented methodologies have recently been introduced into expert system technology. Analogous to the conventional concept of objects in C++, expert system objects tightly couple data and rules. Within large knowledge bases, objects allow a very high degree of modularity and efficiency. During construction, objects are normally not used for the prototyping stage because of their intensive up-front design requirements.

5.1.2.7 Process

The inferencing scheme defines how the rules will be evaluated. *Chaining* is the process where rule firings result in new rules being placed on the agenda. By definition, all expert systems must perform chaining to reach a conclusion from given inputs. Chaining can be either forward or backward. Forward chaining simulates inductive logic. That is, based on existing premises, forward chaining makes the prognosis. Forward chaining is the most commonly used form since a user typically has a set of variables or conditions and would like to know what is the suggested action. Backward chaining simulates deductive logic, and thus, given a result determines a set of probable causes. An example of deductive logic is a mechanic diagnosing an engine that will not start when the air is damp. What are the possible causes? Since both inferencing methods are useful, several shells offer an option to allow the same rule to be run either forward or backward. However, one inferencing method can be emulated in the other.

5.1.2.8 Confidence level

Confidence level is another feature that has various flavors in the expert system community. When a result is given to the end user, the user often wants to know the certainty to which the conclusion was made. For example, if the result returns with a rating 88% certainty the recommendation is probably worth following while a 34% rating is not very useful.

5.1.2.9 Truth Maintenance

Truth maintenance is a key feature of robust expert shells. Many shells offer the ability to place a *daemon* onto a specific fact so when the fact's attributes are accessed, the fact's values are updated before being used even if they are already instantiated. Therefore, an expert system can be used within a real-time system without having to write special intermediate level access functions.

5.2 Expert System Development

5.2.1 Creating the Knowledge Base

We developed the following procedures in building a rule-based expert system for the EOSAEL for Windows Phase I Prototype:

- Identify resources and transfer knowledge from model experts to software engineers
- Find key concepts in and inter-relationships among acquired model application knowledge
- Organize knowledge base and create prototype
- Verify and validate

Because of the iterative nature of building the expert system, the research, implementation, and testing procedures are recursive. The knowledge development process is expected to be continually evolving with frequent interaction and feedback between the model experts and knowledge engineers.

Transfer of knowledge from expert to expert system is best represented in Figure 5-2.

A tremendous amount of an expert's knowledge in a particular field or domain can be found in journal articles, reports, or books. In the modeling community, this knowledge is also encapsulated in computer models. In addition, there is heuristic knowledge that the experts have acquired over the years, usually not formally documented. The best way to access the heuristic knowledge is to interview the experts.

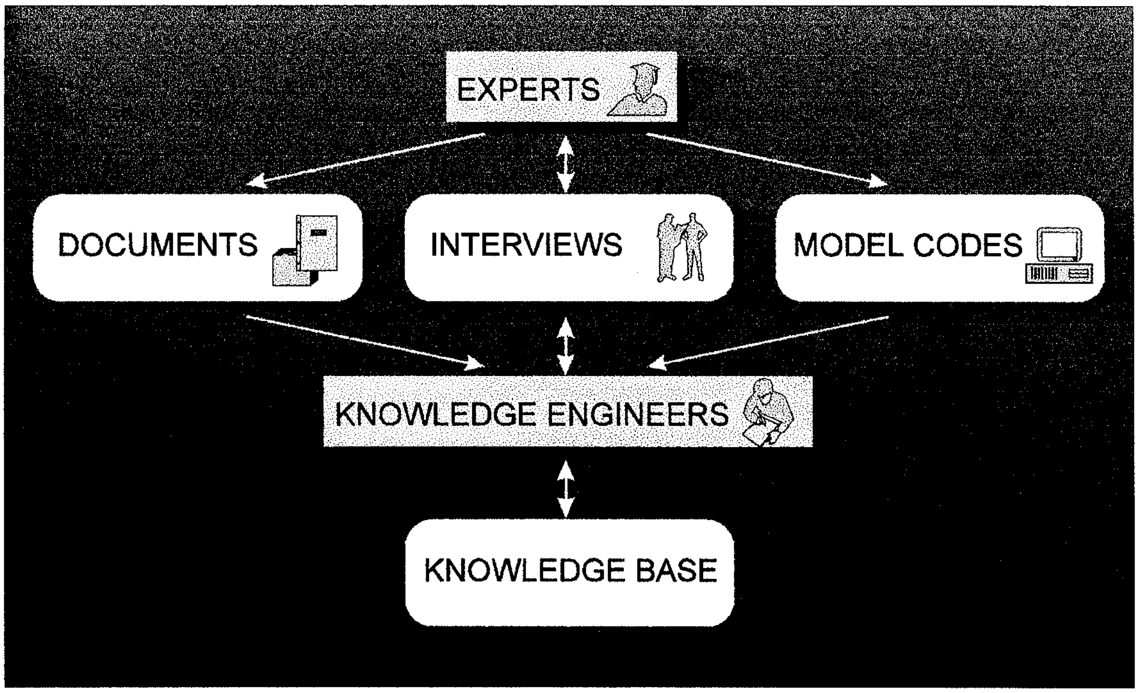

Figure 5-2: Knowledge Acquisition Process.

To build the knowledge base, we first extract the scientific knowledge from printed sources. In addition, we adopt reverse engineering approaches and extract the scientific knowledge embedded in the codes. This means examining the user manuals and source codes to extract as much information as possible about proper use of the codes.

After organizing the preliminary information, we conduct a series of interviews with domain experts to obtain heuristic information. Our experience in acquiring a knowledge base for a model indicates that, in at least a few instances, simple generalizations cannot be made and experts recommend that results be obtained from running the codes. Our experience has proven it useful to perform systematic parametric studies with existing models for additional knowledge acquisition.

Interviewing the experts can be an arduous experience. The model expert may not know at what level to speak to the interviewer in the domain and the interviewer may not be sure how to proceed with the model expert. The productive transfer of data from the expert's mind to the knowledge base takes time and iterations. To this end, the interviewer can use a CASE (Computer Aided Software Engineering) tool to aid in structuring the knowledge base and the interviewing process.

The knowledge obtained from the acquisition phase is separated into facts and rules. Appropriate groups of facts, rules, formulas and algorithms are formed. The knowledge base acquired must be organized in a logical, coherent, and modular fashion. Proper organization of knowledge ensures easy access and expansion in the future. During this effort, the facts/frames and rules/phases are translated into the appropriate expert shell language and routines are coded. At the end of this effort, a prototype of the expert system is created.

5.2.2 Integration of Expert System to Application

One of the greatest difficulties in using expert system techniques is the integration into conventional software paradigms. Many of the earlier knowledge-based systems were stand-alone packages that presented the user with direct access to the shell. When incorporating the technology into another software product, there are a number of ways to do the design.

5.2.2.1 Independent Programs

Some vendors require running their systems as separate applications or executables. This inhibits the degree of incorporation into the final product and presents the user with a stilted application that must temporarily pause to access the knowledge base. This is not as much a problem within environments allowing efficient interprocess communication; however, it is still not the most efficient approach.

5.2.2.2 Part of a GUI Display Object

Another method is to allow the user to construct a GUI object and, instead of the language used being conventional (e.g., C in Microsoft Windows™), the language is the one used by the inference engine. This approach offers a quick method to address a problem that would be difficult to implement within a conventional imperative language. When the expertise of the knowledge base is required in only a limited section of the application, this is a viable approach. However, when the knowledge base might be useful in several areas a more uniform approach is required.

5.2.2.3 A Separate Expert System Module

The main criterion is that the entire expert system be built into one or more callable libraries. By using dynamically linked shareable libraries, the expert shell code can be shared among instances of applications, executable sizes are smaller (good for performance reasons) and updates can easily be made by sending libraries to the end users.

In this approach, the user interface and expert system share control to guide the user through the application. The interface obtains and displays data and the expert system uses the data to reach conclusions and aid in guiding the interface.

5.3 *Expert System Shells*

5.3.1 Selection and Evaluation Criteria

Expert system shells can be rather complicated software packages that require a lengthy period of use to become sufficiently proficient. Also, the methods used by one package may differ greatly from those of another. Therefore, one must take great care in the selection process. A list of the important criteria follows.

Cost and Licensing Considerations	Many packages require the end-user to purchase a run-time license. Royalties are sometimes required on commercially distributed products.
Portability	A platform that requires a special configuration or hardware support beyond what is commonly used would limit the number of potential users and should be avoided.
Programming Interfaces	An expert system that is difficult or restrictive to use is of little value to the overall project. If the shell is simple to use for the applications programmer, then it will be used more often.
Maintenance Considerations	The ability to quickly, robustly, and easily update and maintain the knowledge base is critical to using the expert system in a wide variety of application areas. Some expert systems offer graphical utilities for debugging and mapping the logical flow.
Source Code Availability	Access to the source code allows us to customize and reverse-engineer the methods used.
Company Stability and Support	Many AI companies disappeared in the 1980's; they left customers looking for support.
Expert Shell Features and Performance	Some features such as bi-directional chaining, frames, and objects are very useful but not strictly required. These useful features may be attractive; however, they must be weighed against the expense of making the shell large and slow.

Our overall goal was to select a system that encompassed the desirable traits of the above requirements. Our overall assessment of the products led us to select CLIPS as the primary expert system shell for EOSAEL for Windows.

5.3.2 The C Language Integrated Production System (CLIPS)

The CLIPS expert system shell is the choice of the EOSAEL project and has been extensively extended and utilized within the project. With CLIPS, we have met our goals of creating a portable, modular, and extensible expert system embedded in a GUI. This expert system/GUI approach has many benefits for EOSAEL for Windows . The EOSAEL for Windows end-user will get advice and "the someone looking over my shoulder" approach with no noticeable lag in response time. The following discussion describes CLIPS and how it is implemented in EOSAEL for Windows.

5.3.2.1 Background

Expert system technology was being used at NASA/Johnson Space Center for a number of years but was exceedingly difficult to embed into onboard computers for space missions. At the time, many expert system shells were written in Lisp, which often required special hardware to obtain reasonable performance. Another problem was that the Lisp source code proved tedious to integrate with conventional languages.

CLIPS was developed at NASA/Johnson Space Center to provide a low-cost and portable alternative to Lisp based shells. The shell itself is written in ANSI C to facilitate efficiency and portability. The CLIPS language is Lisp-like but is written much more in a declarative style than the recursive list processing method characteristic of Lisp.

The system has been installed on a variety of computer hardware ranging from x86s to Cray supercomputers and various researchers are working on parallel versions of CLIPS. Because it is a NASA funded effort, it is free to NASA and DoD organizations and their contractors. For a nominal fee, others may purchase CLIPS from the COSMIC center at the University of Georgia, Athens GA.

CLIPS is currently distributed for DOS, Windows, Macintosh, and UNIX/X platforms. Source code is distributed so one may examine the actual algorithms in detail, quickly add patches, and when needed, put application-specific low-level hooks into the shell.

5.3.2.2 Methods of Usage

CLIPS can be constructed into a stand-alone executable without difficulty since that is the default. However, a major strength is that libraries can be constructed that have either domain specific rules actually embedded in them, or a library can read the knowledge base(s) during processing. EOSAEL for Windows has made the CLIPS library into a DLL (dynamic link library) on the Windows system which can be distributed to end users.

The expert system libraries read the compiled knowledge bases. In that fashion, we can control, through the expert system itself, the actual knowledge bases that are used! This will eventually give the end users capabilities (if so desired) to load the knowledge base of their choice without having to download separate dynamic libraries. For additional details on CLIPS refer to the CLIPS User Manual.

5.4 EOSAEL For Windows Knowledge Base

5.4.1 Overall Approach

The EOSAEL for Windows expert system functionality includes diagnosis of user input, advice to non-experts for model selection and set-up, and assistance in establishing parameter values. The EOSAEL for Windows requirements are that the system transparently provide "over-the-shoulder" diagnosis of input parameters and model selections, and the user may always overwrite a recommended value.

We currently have rule packets and knowledge for:

- Parameter Checking (range and inter-consistency),
- Recommending Parameter Values Based on Scenario,
- Checking Scenario Consistency with Nature, and
- An Explanation Facility.

In Phase II we will extend these rule packets and associated knowledge bases to include rule packets to include:

- Additional ARL Codes (ACOUSTIC, CLIMAT, COPTER, etc.),
- Provide Model Synthesis for the Additional ARL Codes,
- Expert-assisted Atmospheric Profiles,
- Selecting Match to Database of Pre-Calculated Results,
- Selecting Appropriate Codes for a User-Defined Scenario,
- Providing Intelligent Defaults for Sensitivity Analysis of a User-Defined Scenario (i.e., recommend bounding extremes for parameter values), and
- Performing Code Run-Time Analysis.

Our implementation provides a GUI that both minimizes and simplifies user input. Input values are immediately checked for their range in the code domain. Out of bounds values are highlighted (red) and expert error messages are posted in status bars on the GUI. Direct access from the GUI to the EOSAEL for Windows support routines and databases (described in the next section) greatly simplifies user input.

5.5 Considerations of Target Models Knowledge Base

5.5.1 Overall Approach and Goals

For Phase II MRC plans to create, expand and enhance the EOSAEL knowledge base to include facts and rules for inclusion of target model information. With respect to target modeling, the EOSAEL expert system requirements include the diagnosis of user input and the advice to non-experts for target model selection and set-up. User input diagnosis involves the recognition of omissions and out-of-range parameter values as well as inconsistencies with nature or the overall scenario being defined.

MRC proposes to design and to develop for Phase II the target related rules and associated database of facts for the additional EOSAEL codes. These target knowledge modules will provide narrow-topic expertise for very specific aspects of target modeling. Our goal is to define rule packets which will:

- Provide recommendations for model selection for a scenario,
- Establish consistent model inputs,
- Diagnosis of model inputs for overall run,
- Provide an explanation facility with detailed reasoning, graphics and references,
- Assist user in building pre-calculated target object libraries,
- Assist user in using pre-calculated target object libraries,
- Recommend rapid calculation estimates, and
- Assist in verification and validation for model development.

Many support routines (e.g., unit conversions) and general environmental knowledge bases developed for the background knowledge base will supplement these rule packets. We anticipate developing other support routines including data reformatting tools. We will address the following categories of knowledge:

- Specific target code knowledge,
- General aircraft knowledge,
- General ground vehicle knowledge,
- Typical engagement scenario information, and
- Typical target analyses performed.

6. EOSAEL for Windows Documentation

6.1 Introduction

An integral part of any system or software development effort is the documentation. In this section we define the EOSAEL deliverable items of software documents for the Phase II development and release phases. Those to be developed in Phase II are shown in the tables below.

Table 2: Phase II Document Development

Development Documents	*Functions*
Software Development Plan	Identifies problems to be solved; outlines procedures to develop software
Software Design Document	Describes how problems will be solved; designs software architecture
Test Plans/Procedures	Outlines plans and procedures to test products
Software Test Report	States results of testing and solutions to problems

Table 3: Release Documents for a Full Scale EOSAEL

Release Documents	*Functions*
Software Users' Manual (*)	Describes how to use EOSAEL
Software Programmers' Manual (ARL and Contractor Responsibility)	Guide to maintaining, modifying, and extending the software by third-party; describes overall code structure; explains all code variables related to model input parameters
Analysts' Manual (Model Developer/ARL Responsibility)	Guide to exploring and understanding the models; describes science, equations, model methodology; defines all input and output parameters; reference to literature
System Managers' Guide (ARL Responsibility)	Guide to code distribution; master index of all documents; troubleshooting; hardware maintenance; code listings
Version Description Document (ARL Responsibility)	EOSAEL version control.

(*) = Partial development and design under Phase I

The above should adhere to the best commercial documentation standards to improve user acceptance.

6.1.1 User Manual

The purpose of a User's Guide is to show the end user how to use the product. Listed below are general features that will be included in the Phase I Prototype and Phase II User's Guides:

- Usage of modern techniques and methods,
- Adherence to standards,
- Logical organization,
- Creative and liberal use of graphics, and
- Reference to test cases and examples.

The contents of the User's Guide are dependent on the application. For Phase II, the full software User's Guide will include the following topics:

- Table Of Contents
- Requirements - Hardware, Software, User Assumed Expertise
- Conventions - Common Abbreviations And Terminology
- Product Description - Brief Summary Of What Software Does
- Installation - How To Install The Product
- Verification - How To Verify Installation Is Successful
- Getting Started - How To Start/Exit Software
- Using The Product - How To Do Basic And Advanced Tasks
- Reference - Describe All Software Features
- Error Messages - What They Mean And What To Do
- Troubleshooting - What To Do If Hardware and/or Software Do Not Work
- Support - How To Get Help
- Index

For the full scale EOSAEL application, the following topics will also be designed into the Phase II User's Guide:

- Brief summary of theory/or reference to source code manuals
- General description of model GUI structure with reference to code manual
- Basic hardware operation and routine maintenance with reference to system managers' guide
- Examples or references to test data for input and output
- Glossary
- Bibliography

The User's Guide will be divided into volumes such as:

- Getting Started - for quick introduction to using the product
- Installation Guide - hardware and software consideration
- Reference Manual
- Theory/description of each major code

The EOSAEL User's Guide design provides for all levels of users in atmospheric models: novice, intermediate, or advanced. The manual will be organized following commercial standards, and takes advantage of modern desktop publishing capabilities. Graphics and examples will be extensively employed to make it easier for users to learn by visualizing and following examples.

6.2 HELP System Related Documents

As part of the entire help package for Phase II, the EOSAEL on-line help will also be supplemented, or synergized by the development of hard-copy system documentation and manuals. The on-line system help will be designed and implemented to refer the user to the additional level of highly technical description that may be required fully understand the correct application of a particular model within the correct context.

6.3 Style

Contemporary commercial software packages contain a tremendous number of features to attract a wide spectrum of users. To support users in exploiting these features, detailed instructions are necessary. For users who only occasionally use the software or who rarely access the more advanced functions, instructions read during prior sessions are quickly forgotten. Therefore, it is crucial that proper documentation, both printed and on-line, is available. There is little doubt that user valuation and acceptance of software packages depends highly on documentation. This will also be the case for EOSAEL and acceptance of the HELP System by the User is crucial.

The current emphasis on and popularity of graphical user interfaces has also permeated the documents-producing environment. The reason is simple: graphics convey information easily absorbed and retained by users. User's Guide documentation typically comes in both hardcopy and softcopy formats. The hardcopy is the traditional media form of a User's Guide and the softcopy format appears digitally as on-line help. The two supplement each other and enhance the effective application of the software.

7. EOSAEL for Windows User Survey

7.1 Introduction

One of the contractual requirements of the SBIR is to conduct a market survey to determine if there are any other Government or commercial organizations that may have an interest in or would be willing to support the follow-on efforts under Phase II. To meet this requirement, MRC developed a formal survey form and distributed it to over 60 Government organizations and private companies. This form was reviewed and approved by ARL prior to being distributed.

7.2 Purpose of Survey

The goals and purpose of the survey are:

- To canvas present and potential EOSAEL users on their requirements for EOSAEL model interface upgrades.
- To determine interest in Government and commercial organizations on the upgrade of other complex geophysical model interfaces to include Windows GUIs with an embedded knowledge based system.
- To determine to what extent other Government or commercial organizations would be willing to participate or to support Phase II and Phase III of the SBIR effort.
- To identify potential new users and applications for EOSAEL.

7.3 Results

Out of the 13 respondents, 3 indicated they are current users of EOSAEL and use EOSAEL for atmospheric effects research. Two use EOSAEL on VAX systems running VMS and one used some of the models on a PC under DOS. The response to ease of use and use under Microsoft™ Windows indicate that two users think EOSAEL is difficult to learn and set up a model run and that a Windows version would enhance model applications and staff efficiency.

One EOSAEL user also indicated that GNU Plot would be a useful, non-commercial, addition to EOSAEL.

The overall impressions of EOSAEL are:

- The programs are scientifically very good (1 response),
- There are too many versions (1 response), and
- They are difficult to compile and get user support (1 response).

Two current users stated they are willing to state a formal requirement for EOSAEL upgrades that support their needs. Finally, one respondent would be willing to attend project and design reviews and/or comment on project documentation related to EOSAEL upgrades.

The second section of the Survey dealt with the software upgrade techniques used to develop EOSAEL for Windows

Table 7-1: Response To Software Questions

Question	Yes	No
Learn more about EOSAEL	9	2
Attend EOSAEL Meetings	9	2
Interest in SBIR results	12	1
Receive copy of COMBIC in EOSAEL for Windows	13	0

The breakdown of job descriptions of the respondents is shown below. One response did not include the job description.

Staff Meteorologist
Plumes Program Software Manager
Atmospheric Effects Researcher
Program Analyst, Sec Def Level
Weapons Effects Researcher
ARL Contractor
Astronomy Data Analyst
Manager, Optical Technology Branch
Physicist, Coastal and Desert Regions
IR System Designer
Atmospheric Effects Modeler
Senior Systems Analyst, Simulations

7.4 Analysis and Recommendations

MRC has the following observations on the survey results received to date:

7.4.1 Improvements to EOSAEL

- Although only a few respondents had used EOSAEL, replies indicated a certain level of user dissatisfaction with the present level of difficulty of applying the model and in getting technical support.
- One user indicated some dissatisfaction with EOSAEL model version control.
- One user noted that integration of GNUPLOT into the EOSAEL system would be helpful.

7.4.2 RECOMMENDATIONS:

- **Publish Phase I Results**: Based on the survey, there is a great deal of potential interest in the results of the Phase I SBIR. As a result of this interest, ARL may want to announce availability of copies of the final report and prototype software.
- **Advertise Availability of EOSAEL**: There seems to be additional untapped interest in EOSAEL. ARL may want to make its availability better known through scientific publications, or through poster board sessions at technical conferences.
- **Complete Phase II and Phase III:** Users of EOSAEL and users of other environmental codes have recognized the need for interface upgrades to a GUI in order to make these types of models easier to use and to protect the long term investment made in these models. Thus Phase II and Phase III of the SBIR need to be completed and the results made available to the Government and commercial sectors.

8. Summary

8.1 Conclusions

As shown in this report and by hands-on demonstration of the prototype software delivered to ARL, MRC has successfully developed a fully functional Phase I Prototype EOSAEL System with COMBIC as the first test module.

Key features of the MRC innovative design and software are:

- The design implements an off-the-shelf expert system (CLIPS) which monitors and guides the user through an EOSAEL application.
- The prototype delivers a GUI that streamlines application of COMBIC through the following features:
  - *Windows style toolbar*
  - *Data entry screen which presents the features of the COMBIC control cards familiar to COMBIC users*
  - *A set of descriptive tools which link to the Expert System*
  - *A set of functional tools (Exit, Save, Run, and output analysis) which perform stated functions.*
- The Phase I Prototype contains a Hypertext HELP System that "links" with the embedded expert system and provides detailed assistance to users as they are guided through an application.
- The design philosophy maintains the EOSAEL Scientific Codes in their native format which allows model changes to be transparent to the EOSAEL for Windows architecture.
- The modular design presented allows for ease of maintenance, adaptation of new features, growth of the EOSAEL Codes, and provides for growth and rule updates to the embedded expert system.
- The prototype design, software, and Hypertext Help are fully scaleable to the entire suite of models that comprise the EOSAEL code library.

8.1.1 RECOMMENDATIONS

8.1.1.1 Publish Phase I Results

Based on the survey and success of the Phase I effort, ARL may want to announce availability of copies of the final report and prototype software.

8.1.1.2 Advertise Availability of EOSAEL

ARL may want to make EOSAEL's availability better known through scientific publications, or through poster board sessions at technical conferences.

8.1.1.3 Complete Phase II and Phase III

Users of EOSAEL and other environmental codes have recognized the need for interface upgrades to a GUI in order to make these types of models easier to use and to protect the long term investment made in these models by the Government and private industry. Thus Phase II and Phase III of the SBIR need to be completed and the results made available to the Government and commercial sectors.

8.2 Considerations for Phase II

The success of Phase I has demonstrated the feasibility of Phase II and Phase III and has greatly reduced the technical risk associated with these phases.

9. References

Booch, G., "Object-Oriented Analysis and Design with Applications," Benjamin/Cummings Inc., Redwood City, CA, 1994.

Forgy, C., "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem," Artificial Intelligence, 19 pp. 17-37, 1982.

Giarratano, J. C., "CLIPS Software Documentation", Software Technology Branch, Information Systems Directorate, Johnson Space Center, NASA, 1993.

Hughes, P. M., Shirah G. W., Luczak, E. C., "Advancing Satellite Operations with Intelligent Graphical Monitoring Systems," AIAA Computing in Aerospace IX Conference, San Diego, CA, Oct. 1993.

Lanich, W., Kreiss, W., Niple, E., "Electro-Optical Aerial Targeting Workstation," NAECON, IEEE, 1989.

Wirfs-Brock, R., Wilkerson, B., Wiener, L., "Designing Object-Oriented Software," Prentice-Hall Inc., Englewood Cliffs, NJ, 1990.

CLIPS User's Guide

APPENDIX A
EOSAEL MARKET SURVEY

EOSAEL For Windows™

***E*NVIRONMENTALLY *O*RIENTED *S*YSTEMS *A*TMOSPHERIC *E*FFECTS *L*IBRARY**

USER SURVEY

The goals of this survey are:

- To canvas present and potential EOSAEL users on their requirements for *EOSAEL* model interface upgrades.
- To determine interest in government and commercial organizations on the upgrade of other complex geophysical model interfaces to include Windows™ GUIs with an embedded knowledge based system.
- To determine to what extent other government or commercial organizations would be willing to participate or to support Phase II and Phase III of the SBIR effort.
- To identify potential new users and applications for *EOSAEL*

To have an impact on the results of this effort, and to help us to provide you with enhanced model interfaces and advanced techniques, **please return this survey NLT 8 September 1995**.

- **EMAIL**: If you prefer to respond by email, request a text copy of the survey from **pnoah@pldac.plh.af.mil**.
- **FAX**: Faxes may be sent to: Dr. Paul Noah, MRC, **(603) 891-0088**, or **(603) 891-1447**.
- **Regular Mail**: An addressed, stamped envelope is included for your convenience.
- **Non-Attribution**: Your individual inputs will be ***held in confidence and are non-binding***; however, they will be tabulated into an analysis of survey results.

PREPARED BY THE
MODELING, SIMULATION AND DATA MANAGEMENT DIVISION
MISSION RESEARCH CORPORATION
ONE TARA BOULEVARD
SUITE 302
NASHUA, NH 03062

EOSAEL User Survey

SECTION 1 - *EOSAEL* Software Applications

Are you presently using the EOSAEL library?

- ☐ Yes.
- ☐ No -- Skip to Section 2.

Which describes your application of the EOSAEL software? You may choose more than one.

- ☐ Atmospheric effects research.
- ☐ Tactical decision aid modeling.
- ☐ Battlefield war gaming simulations.
- ☐ Sensor System design and performance prediction.
- ☐ Other____________________.

How do you access the EOSAEL Library?

- ☐ Models available on local system.
- ☐ Operate Models via remote login at ARL.
- ☐ Download Models via INTERNET or modem.

On what Platform do you run the models?

Hardware:	Operating System:	Version #

Which specific EOSAEL codes do you use most frequently?

☐ CLIMAT	☐ CLTRAN	☐ COMBIC	☐ COPTER	☐ FASCAT
☐ FCLOUD	☐ GRNADE	☐ ILUMA	☐ KWIK	☐ LASS
☐ LOWTRN	☐ LZTRAN	☐ MPLUME	☐ NMMW	☐ NOVAE
☐ OVRCST	☐ RADAR	☐ TARGAC	☐ XSCALE	

What is your opinion of the ease of use of the model(s) listed in the response above?

- ☐ Difficult to learn and set up a model run.
- ☐ Easy to use and iterate model runs.
- ☐ Present model interfaces make model iterations difficult and not worth time to apply to research.
- ☐ Model unfriendliness has caused us to use or develop other models for in house applications.

EOSAEL User Survey

Would you use the EOSAEL codes on a PC platform under Microsoft® Windows™?

- ☐ A definite advantage and would be interested in supporting model conversion efforts to WINDOWS environment.
- ☐ Yes, would enhance model application and staff efficiency.
- ☐ Nice to have, but model applications are infrequent, and not worth time to change to new system.
- ☐ Present model applications and configuration are adequate and no change needed.
- ☐ Too much in house auxiliary software has been developed to support present EOSAEL analysis and conversion to new system would be cost prohibitive.

What analysis packages do you use to support EOSAEL analysis? Please list.

____

Of those GFE or Shareware packages listed, which would you like added to the EOSAEL Library?

____

What are your impressions of the EOSAEL models? Please list strengths and/or weaknesses or needs for improvement.

____

____

____

____

My organization/company would be willing to support requirements definition:

- ☐ Willing to state a formal requirement for EOSAEL upgrades that support my needs.
- ☐ Present system meets my requirements, not interested in supporting or defining new requirements.

Would your agency/company be willing to contribute resource support to an EOSAEL Library upgrade?

- ☐ Yes, willing to provide share of funding for a reasonable effort if upgrades support my future needs.
- ☐ Yes, willing to provide software engineering support for upgrades that meet my requirements.
- ☐ Yes, willing to contribute software modules to integrate into upgrades.
- ☐ Yes, willing to attend project and design reviews and/or comment on project documentation related to EOSAEL upgrades.

EOSAEL User Survey

☐ No, not able to contribute nor willing to contribute. Present models are satisfactory.

EOSAEL User Survey

SECTION 2 - Software Upgrade Techniques

This part of the survey deals with the software upgrade techniques developed under the ARL SBIR *EOSAEL* effort.

Would your organization desire to learn more about the EOSAEL Model Library and gain access to it?

☐ Yes

☐ No, no interest at this time

Would you like to attend an ARL sponsored EOSAEL Modeling and Model Upgrade Meeting?

☐ Yes

☐ No, no interest at this time

Would your organization be interested in receiving a copy of the Phase I EOSAEL SBIR report when made available by the ARL?

☐ Yes

☐ No, no interest at this time.

When made available, would your organization or company be interested in receiving a demo copy of the EOSAEL COMBIC model prototype knowledge-based WINDOWS GUI developed under the Phase I SBIR?

☐ Yes

☐ No, no interest at this time.

Do you now have atmospheric or other geophysical models that could be enhanced or commercialized by "wrapping" them in a knowledge based GUI?

☐ Yes (please list models and software language).

____

____

☐ No. Present models have sufficient interfaces.

Is your organization presently involved in upgrading existing models to run on PC or work station platforms with WINDOWS GUIs?

☐ Yes/Please describe.

____

____

____

☐ No, not interested, present models adequate and upgrade not cost effective.

☐ Not presently upgrading models, but planning for future requirements to do so.

Please state your present position with a brief job description.

____

____

____

____

Do you know of any one else that may have an interest in this effort and survey?

☐ Yes

☐ No

If so, please provide their name, organization, and address.

____

____

EOSAEL User Survey

____

____

May we use your name in contacting them?

☐ Yes ☐ No

Appendix B

Acronym List

AI	Artificial Intelligence
ANSI	American National Standards Institute
ARL	Army Research Laboratory
CASE	Computer Aided Software Engineering
CIDI	Common Input Data Interpreter
CLIPS	C Language Integrated Production System
COMBIC	Combined Obscuration Model for Battlefield Induced Contaminants
COPTER	EOSAEL Helicopter Downwash Model
COSMIC	Computer Software Management and Information Center
CPD	Common Parameter Definition
CPD_Parm	Common Parameter Definition (Parameter)
DLL	Dynamic Link Library
DOD	Department of Defense
E-O	Electro-Optical
EOHELP	EOSAEL Help Documents
EOSAEL	Electro-Optical System Atmospheric Effects Library
FORTRAN	Formula Translation
GNU	GNU's Not Unix - Free Software Foundation's unix implementation
GNUPLOT	Free Software Foundation's Plot Model for GNU
GRNADE	Hand Grenade Explosives Model
GUI	Graphical User Interface
IR	InfraRed
LINUX	A shareware version of Unix
MDI	Multiple Document Interface
MRC	Mission Research Corporation
NASA	National Aeronautic and Space Administration
SBIR	Small Business Innovative Research
TARGAC	Target Acquisition Model
VMS	Virtual Memory System