

DEVELOPMENT OF A SOFTWARE PACKAGE FOR
CHEMICAL ENGINEERING THERMODYNAMIC
RESEARCH AND CALCULATIONS

By

WUZI GAO

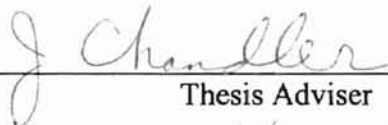
Bachelor of Science
Xi'an Jiaotong University
Beijing, P. R. China
1990

Master of Science
Beijing University of Chemical Technology
Beijing, P. R. China
1993

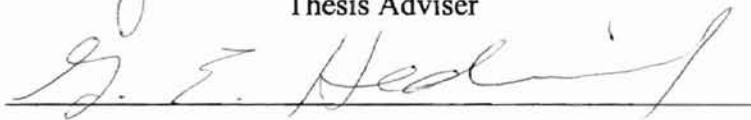
Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 1999

DEVELOPMENT OF A SOFTWARE PACKAGE FOR
CHEMICAL ENGINEERING THERMODYNAMIC
RESEARCH AND CALCULATIONS

Thesis Approved:



Thesis Adviser









Dean of the Graduate College

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Dr. J. P. Chandler, my major adviser in the Computer Science Department, for his kind guidance during this thesis work. Special thanks go to Dr. K. A. M. Gasem, my advisor in Chemical Engineering for his support and precious advice. None of this work would have been possible without Dr. J.P. Chandler and Dr. K. A.M. Gasem's consistent advice and support. I also truly appreciate my committee members Dr. G. E. Hedrick and Dr. H. K. Dai for their service on my committee and advice.

My parents, Shunyu Gao and Xiaolan Wang, deserve my deep gratitude for being a constant source of support and love in my life.

Last, but not least, my appreciation goes to my wife, Liwei Cai, for her love, encouragement, and support.

TABLE OF CONTENTS

Chapter	Page
I INTRODUCTION.....	1
1.1 Motivation.....	1
1.2 Plan of this study.....	3
II LITERATURE REVIEW AND CONCEPTUAL DESIGN.....	5
2.1 Programming language for graphic user interface.....	5
2.1.1 Graphic user interface	5
2.1.2 Sequential programming language vs. event-driven programming Language	7
2.1.3 Object-oriented programming technology.....	8
2.2 Backend calculation.....	10
2.3 Database model selection.....	12
2.3.1 Selection of a database mangement system..	14
2.4 Overall structure conceptual design.....	15
III INTRODUCTION TO GEOS 7.0.....	17
IV PROJECT DEVELOPMENT.....	21
4.1 Database Design.....	21
4.1.1 Pure Component Relation Design.....	22
4.1.2 Binary Systems Relations Design.....	24
4.1.3 Reference Relation Design.....	24
4.2 GUI Design.....	26
4.2.1 Main Menu Design.....	26
4.2.2 Log File Forms.....	27
4.2.3 Data File Form.....	29
4.2.4 Other Forms.....	30
4.2.5 Design of The Automatic Plotting.....	30
V Summary.....	44
LITERATURE CITED.....	45
APPENDIX A GEOS 7.0 MENU.....	48
APPENDIX B ONE CASE STUDY USING THIS SOFTWARE.....	62

LIST OF FIGURES

Figures	Page
1 Structure of Current Design.....	16
2 The Interface for GEOS 7.0.....	18
3 Example Log File.....	19
4 Example Data File.....	20
5 Relationship of the Designed Database.....	26
6 GEOS Menu.....	32
7 Items Under File in the Menu Bar.....	32
8 The First Tab in SSTab Control.....	33
9 Models Section as Specified in Log File.....	34
10 Options Section as Specified in Geos 7.0 Menu.....	34
11 Project Summary Window.....	35
12 File Insertion Window.....	35
13 Data File Creation Window – Component Selection.....	36
14 Prompt If the Selected System Is Not In the Database.....	37
15 Sample Design of The Graph.Txt File	38
16 Interface of Automatic Plotting	40
17 Dialog To Open A Proper File To Plot	40
18 Interface To Show Available Choice	41
19 Sample Plot of X vs P For H2 / N-C20 System.....	42
20 Sample Plot of C_{ij} vs T_c Across The Systems.....	43

CHAPTER I

INTRODUCTION

1.1 Motivation

In Chemical Engineering education, thermodynamics is essential. To design the processes, like separations (distillations, adsorptions, etc.) and reactions, knowledge of thermodynamics is needed to calculate the equilibrium properties, that is, to set the limit that can be reached ideally by nature. Those processes normally have a large impact on economics. Frequently quoted statistics (Zeck and Wolf, 1992) suggest that in conventional chemical plants as much as 70% of the capital costs and 90% of the operating expenses are associated with phase separations. Therefore, accurate predictions of phase equilibrium are essential for proper design, operation and optimization of chemical processes. However, thermodynamics, is relatively more abstract than other courses and very often one of the most difficult disciplines to understand and master. A powerful tool is needed for thermodynamics education. Although commercially mature software, like ASPEN+, are available, they are generally too complicated and not designed to help understand thermodynamic principles.

For chemical engineering thermodynamic research, instead, special software designed for model development is needed. The software needs to accommodate new models easily, do the calculations, regress the model parameters, and generate necessary statistical information for model development efforts.

Fortunately, Gasem (1997) has developed such software, Generalized Equations Of State (GEOS), over the years and the current version of it is 7.0. GEOS 7.0 includes 37 files, more than 6000 lines of source code. It includes most of the models in chemical engineering thermodynamics and is structured in such a way as to adopt new models into the software easily. Options exist like calculating isotherm by isotherm, or system by system. The software also generates a complete list of statistic information for model evaluations for each run.

However, the complexity of GEOS 7.0 is huge, not only because of the large number of files and lines in the source code, but also in the detailed background understanding of the theory of each method and the algorithm associated with it. It takes a long time for the students to properly use this software and take full advantage of it.

A major reason for the difficulty to use this software is that the interface between this software and the users is written in FORTRAN and the level of its user-friendliness is not enough. Users need to have a good understanding of the setup of the software to use it. The interface is reviewed in Chapter III to show how the interface in FORTRAN is used.

This user-friendly problem in FORTRAN is not new. People have recognized this problem a long time ago. The software developed in traditional, sequential programming language has not been user-friendly. One typical example to show the user-friendly problem is the long learning curve for DOS system compared to that of Windows system used in personal computers (PCs). A Graphical User Interface (GUI) is the key to solve this problem. The Windows operating systems (3.x, 95, etc) are such successful applications of GUI to solve the user-friendly problem associated with DOS.

They have greatly eased the burden of users to learn the operating systems. They have been so successful that they quickly dominate the market of operating systems for PC. GUI is now mature and ready to be applied to many software applications. There are many companies to build the GUI to make user-friendly programs.

In this thesis, the GUI will be built for GEOS 7.0 to make this software user-friendly. The following section addresses what will be planned to do.

1.2 Plan of this study

In this thesis, a software package will be proposed to be used in chemical engineering thermodynamic research and calculations. At the beginning, a relational database will be designed and implemented in the MS Access 7.0 database system to store the related data. The graphic user interface (GUI) for this software will be built using Visual Basic 4.0 as the front end for the GEOS 7.0 calculation using FORTRAN. The designed database will be connected as backend to the GUI to provide automatically the necessary data as input file for GEOS 7.0. A database management program will be developed using Visual Basic 4.0 to convert the data in plain text file to the current MS Access 7.0 database. The FORTRAN program will use the most recent console version of the GEOS 7.0 software developed by Gasem (1998). A powerful regression subroutine, MARQ 3.0, developed by Chandler (Chandler, 1981) is incorporated in GEOS 7.0 to regress the model parameters if the calculation is in regression mode. This subroutine has a well-designed interface with other parts of the program and can be manipulated individually. It includes four methods for regression, that is, the modified

Gauss-Newton method, Gauss-Newton method, a modified form of Marquards method, and Marquardt-s method. Normally the modified Marquard method is the best choice. In Chapter II, the choice of Visual Basic 4.0 as the GUI developing tools, the choice of database model, and the conceptual design for this study will be discussed. In Chapter III, GEOS 7.0 will be introduced to help the understanding of the Chapter IV. In Chapter IV, the project development in this study will be discussed. In Chapter V, conclusions and discussion about this work will be given.

Upon the successful completion of this work, all the user will see is the graphic user interface for GEOS 7.0. The learning curve for GEOS 7.0 will be greatly reduced. The preparation of the data file will be automated. All the operations on the database and calculations from the FORTRAN part will be specified in the interface. The user will not be allowed to see the database and FORTRAN code. Only authorized persons will access and update the database and FORTRAN code. The burden of plotting the results will be eased.

CHAPTER II

LITERATURE REVIEW AND CONCEPTUAL DESIGN

2.1 Programming Language for Graphic User Interface

In this section, the development of the graphic user interface will be analyzed briefly. The programming language suitable for the GUI will be reviewed. The programming language used for this study and the conceptual design will be recommended.

2.1.1. Graphic User Interface

For software written in traditional programming languages such as FORTRAN and C, the user needs to have a good understanding of the program in order to use it. This has severely limited some good application software from being widely used. Efforts such as graphic user interface development have been made to overcome this inherent difficulty over the years. A graphic user interface is part of the program by which a backend calculation and the user communicate with each other. It allows users to see the environment in a visual way and therefore it is easy to control the program. In this way both the user and the program control the output of the program. This is called a user-friendly program. In particular, windowed interfaces provide more feedback about what the user can (and cannot) do (Leavens, 1994). These interfaces provide a virtual

image of what the computer and the application can offer, in addition to providing a consistent look and feeling across all Windows applications. As Marcus (1995) has pointed out, the GUIs reduced the user's information load in the following ways:

- Present commands, options, or data to the user on the appropriate application display.
- Display prompt information to each option or button on the screen, so the user can be informed which one to choose relevant to his or her needs.
- Provide immediate feedback of the user click. Check if the program supports this choice combination. If so, provide the solution. (Marcus, 1995).

Interaction is the means by which the user controls the execution of an application. The concept of pointing to an object and then selecting it, often referred to as point-and-select, is an essential factor in achieving effective human application interaction (Marcus, 1995). The interaction is achieved by the use of the keyboard or mouse.

There is no certain guideline as to how a GUI should be designed. However, some commonly desired characteristics of such an interface design include (Hopper and Newman, 1986; Raghunathan, 1996):

1. letting the user control the outcome (event-driven),
2. addressing the user's level of skill and experience,
3. being consistent,
4. protecting the user from the inner workings of the hardware and software,
5. providing on-line documentation,
6. minimizing the burden on the user's computer, and
7. following the principles of good graphic design.

In this study, these general guidelines will be followed to design the GUI. Visual Basic 4.0 is a good candidate for a language in which to develop a GUI in these respects.

2.1.2. Sequential programming language vs event-driven programming language

The convention languages, such as FORTRAN and C, are called sequential languages, in which all the calculations are executed sequentially. Execution begins at the top of the program, then flows through function calls and control-flow statements. The behavior of the program is predictable. The program is in control, while the user plays a subordinate role of entering keystrokes when requested. It is still possible to present menus to the users, have the user make any choices from the menus, and have the program carry out the user's commands, but the timing of the presentation is determined by the program rather than by the user. Another programming mechanism is called event-driven (also called message-driven), and is used in such programming as Windows programming and Visual Basic programming. An event is generated by clicking the mouse, or by pressing a key. Everything the user does generates an event. Internal or other external devices may be used to generate events. A user may define his/her own events depending on the application. A special function called the callback function is used to handle the message for each event. In an event-driven program, in contrast to a sequential program, the user and external events control what parts of the program are executed. The user generates events using a mouse or other device, and the program is expected to make an appropriate response to each click. For example, a user resizes a window, and then the callback function detects and sends the message to the program

indicating the new window size. The program will act accordingly. Therefore, the implementation sequence does not depend on the programmer, but on the user. However, the response to each event is defined by the programmer. This is key in understanding an event-driven program.

The complexity of the event-driven program is reduced because (a) the codes are organized in separate modules to serve a specific task, and (b) the number of global variables is reduced. Also, maintenance is easier for event-driven programs because message passing is centralized and rarely needs to be changed. An event-driven program is ready at any time to process any sort of event, but a traditional program is not (Leavens, 1994). Because the event-driven interface can respond to events at any given time, it makes developing a program with this interface much simpler in the case where user interaction is required.

The Visual Basic programming language uses the event-driven programming style. A Visual Basic program is a set of independent pieces of code that are activated by, and so respond to, only the events they have been coded to recognize. This combination of underlying programming code and graphic user interface makes Visual Basic applications possible and useful.

2.1.3. Object-Oriented Programming Technology

One of the modern technologies in programming languages is object-oriented programming (OOP), which supports classes, instances, and message passing as well as inheritance. It combines data and functions into a single unit called an object or a block.

According to Yourdon, object-oriented technique is one of the most important development in computer science since the introduction of structured techniques during the 1970s and 1980s (Yourdon, 1994). The origin of OOP can be traced back to Simula in 1967. This new technique was cited to be the key to increased productivity and improved reliability (Timothy, 1991). During the mid-1980s it started to come into general use (Florentin, 1991). OOP is a style of programming that models data in terms of real world objects. It is a combination and normalization of ideas that have been around for many years (Hu, 1990). OOP is a new way of thinking about what it means to perform computation, about how the programmer can structure information inside a computer. OOP is often referred to as a new programming paradigm. Some other programming paradigms are the imperative-programming paradigms (Pascal language or C language support this paradigm), the logic-programming paradigm (Prolog), and the functional-programming paradigm (FP or ML) (Timothy, 1991).

Although Visual Basic does not fully take advantage of all OOP features, such as polymorphism, as Visual C++ does, it does utilize some concepts in OOP. The codes are organized by the unit of each object. Inheritance is also implemented in Visual Basic. For example, a Command Button is a button. It inherits all the basic features associated with a button. Then a designer can have many Command Buttons in his program, and all inherit the Command Button properties. All the user needs to do is to add the specific function each button needs. Although there are advanced features that let users further explore the inheritance in Visual Basic programs, they are generally not used to retain simplicity in Visual Basic programs.

Based on the above analysis, Visual Basic has been chosen for the current software development. The working of a Visual Basic application can be explained in the following four steps (Cornell, 1993):

1. Visual Basic (VB) monitors the windows and the controls placed on each window for all the events that each control can recognize (mouse clicks, movements, keystrokes etc.)
2. When VB detects an event, it examines the application to identify related code, if any, associated with the event.
3. If such event code is present, VB executes the code that makes up the procedure and goes back to step 1.
4. If there is no code present related to that event, VB waits for the next event and goes back to step 1.

The programming language built into Visual Basic (an extension of the one available in QBasic or QuickBasic) has easy-to-use graphic statements, powerful built-in functions for string manipulation and sophisticated file-handling capabilities. In addition, it utilizes modern modular programming style, which leads to less error-prone applications.

2.2 Backend Calculation

A user-friendly problem in software application can be distinguished in the following five different levels (Chandler, 1999):

A Application programs in which the user has to provide part of the code, e.g., MARQ;

- B Pure batch jobs that process data files in a fixed procedure;
- C Applications in which the user controls by commands in a batch file, such as the BMD statistical package from UCLA, the IBM linear programming package MPSX, SAS, etc.;
- D Menu-driven applications such as PKZIP, ASSIST/I and almost all other DOS applications;
- E Event-driven applications.

A conventional programming language such as FORTRAN is excellent for engineering calculation, but the user friendliness of the program often lies in between A, like Marq, and C, like GEOS 7.0. This is disadvantage for the developed program to be widely used. However, much of the code written in the conventional programming language developed over years is of enormous value. FORTRAN still rules in the scientific and engineering fields as an ideal tool for the creation of technical software. It cannot be ignored because the structure of the language lends itself to extremely efficient computing, especially for precise vector and statistical analyses. The code written in FORTRAN often fits well into a modular development environment. Nevertheless, an increasing amount of new development is being done with newer languages. The user-friendliness in the new language will be in level E, like Visual Basic and Visual C++ program. It will be a great benefit to include the code in conventional language to those new developments. Fortunately, the new language like Visual Basic can connect to the conventional language like FORTRAN through the Dynamic-Link Library (DLL) files. Through DLL files, the FORTRAN code is encapsulated into Visual Basic objects. Such

objects retain the functionality of the original code while featuring the simplicity of implementation and use that is characteristic of code created with Visual Basic. However, a certain step must be done before the FORTRAN code is connected with Visual Basic. First, change the code in FORTRAN into a DLL file. Then, wrap the DLL in a Visual Basic OLE object and test the DLL within an application based on Visual Basic. Third, turn the DLL into an independently executing object that provides the services of the DLL to any OLE-enabled development tool.

2.3 Database Model Selection

A database is a program that stores information in an orderly way so that data can be retrieved quickly and organized in different ways. A computer can use the information in a database very fast and efficiently. For example, it can sort the people in the database by name, but it can also sort them by team, by birthdate, birthplace or by any other information the database contains. It can use the information about birthdates to calculate the average age of each team.

A database is an organized collection of related data. A database hides some of the details of the actual organization of the data. When a program uses a database, the location and structure of the data can change without requiring modifications to programs. For example, a file can be moved to another disk volume or new fields can be added to a file. This is called physical data independence (Korth and Silberschatz, 1991). In both of these examples, programs require no changes to reflect the changes to the data. Databases also eliminate pathnames of files from an application. This allows a system

administrator to move files of a database to another data storage volume or computing platform without the need for application modification. Each database can specify a transaction log, a file that logs changes to the database. These changes include all record insertions, updates, deletions, etc.

Different models for design and implementation of a database are available in the literature. These include, but are not limited to, flat-file, hierarchical, network, relational and object-oriented data models. The differences between the flat-file and the relational data models are highlighted in this section. Further information on the various data models can be obtained from McFadden and Hoffer (1993).

In the past, the flat-file approach was the predominant method to database design and implementation because of its ease of use and portability across platforms. Even now most small engineering databases are stored using a flat-file approach. A flat-file database consists purely of text-based files. The name of each file reflects the type of data that is stored in that file. Each file contains column headings that denote different data attributes while each row represents records. The data for chemical engineering thermodynamics used to be stored in plain text files. While this is simple, it has disadvantages. These include error-proneness, data redundancy, inconvenience and lack of data integrity, which can lead to update and deletion anomalies (McFadden and Hoffer, 1993). Also, these databases are not amenable to quick data access, updating and deletion, and complex coding has to be done to achieve such desirable characteristics.

The relational data model has seen widespread acceptance since its conception at IBM in 1969 by Codd (1969). This model owes its existence to the need for a formal

database theory and has its roots in mathematical set theory. The relational data model consists of the following three major components (Fleming, 1989):

1. Data structure – data are organized in the form of tables and are related to other tables.
2. Data manipulation – operations are used to manipulate data in the database.
3. Data integrity – validation rules and relations are provided to specify rules that maintain the integrity of data during manipulation.

The three components outlined above make an implementation with the greatest efficiency and ease of use. These components are discussed in detail by Fleming (1989) and Sanghavi (1995).

From the discussion above, the advantages of the relational data model and the need for a relational database are self-evident.

2.3.1 Selection of a database management system

A good database management system (DBMS) is the key to the development of a good database. Even though all relational database management systems (RDBMS) are based on the same relational data model, there are vast differences in the implementation. A number of criteria have to be taken into consideration when selecting a RDBMS. Some of the criteria to be considered are data types available, speed, query capabilities, data integrity, domain validation, data sharing, security, safety and amenability to interface design (Sanghavi, 1995).

Microsoft Access 7.0 has been chosen as the RDBMS for this study after a careful evaluation of the criteria discussed above. Another major reason for the selection of Access 7.0 is its seamless integration with the Visual Basic programming system through the JET (Joint Engine Technology) database engine. This means that the proposed software will be able to run on the computer without the presence of MS-Access software.

2.4 Overall Structure Conceptual Design

In this study, Visual Basic 4.0 will be used to build the interface. The interface will use standard Windows features, as users are familiar with, such as File (Open, save, etc) or Edit (copy, paste, etc) menu items. All the flag controls used by the FORTRAN calculation will be displayed on the interface with explicit instructions, with each major section in GEOS 7.0 shown in a separate form. Certain options (like options related to theta methods when theta methods will not be used) will be dimmed according to the options a user has chosen, to avoid mistakes. In this way, the options will be easily understood and set by the users. A separate window will be used to let the user choose which systems will be chosen. A MS Access 7.0 database will be hooked up to the window to provide the necessary input data for the systems that need calculations. The access to the database will be automated and users just click the corresponding button. The FORTRAN code is connected through a DLL file to provide the actual calculation. A structure of the current design is shown as in Figure 1. When this software is used,

only the GUI section is visible to the users. Manipulations on the database and FORTRAN program are specified on the user-interface and are invisible to the users.

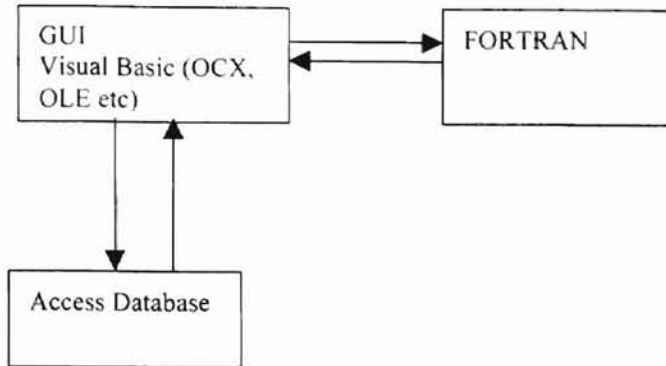


Figure 1. Structure of Current Design

The first step to use this software is to let users select the necessary flags in the interface to decide the conditions and properties that need to be calculated. Then the queries are sent to the database to collect the necessary data. When the data are successfully collected, control returns to the GUI. Then users check the setup and send the control to the FORTRAN module to begin the calculation. After the calculation is completed, the results and control are sent back to the GUI and the result is displayed. Then the program waits until the user acts again.

CHAPTER III

INTRODUCTION TO GEOS 7.0

In this chapter, one case study using GEOS 7.0 is shown to help understand the design of the database and GUI in this study.

The options for calculations using GEOS 7.0 are set in a log file before the program can be run. Usually an example log file is distributed to the users. Then users can customize the necessary parameters. One example log file is shown in Figure 3. Generally, a menu will be distributed along with the executable GEOS files, the example log file, and data file. The menu is shown in Appendix A. The first part of the menu is used to set up the log file. The second part is to set up the data file, like HC10.dat. To properly set up a log file according to the menu turns out to be a non-trivial task. Much confusion exists when a user has not fully understood the program. The users need to understand what each number means and what combinations of options can be chosen. Detailed explanations are given in the menu. The users are assumed to be senior or graduate level students whose major is in chemical engineering. An example data file "HC10.dat" is shown below in Figure 4. The example data file is for a binary mixture of hydrogen and n-decane vapor-liquid equilibrium (VLE) calculations. The data file format may vary with the properties to be calculated, which database in the log file used, etc. The second and third lines are the physical properties of each pure substance. The fourth and fifth lines are for vapor pressure model constants. These are usually checked from related books, have the units converted if necessary, and are manually input to

follow exactly the format specified by the FORTRAN program. The individual data points are from published articles or our experiments. The data points are input manually according to the input format. It is very easy to make mistakes, either by misplacing the data, or inputting the wrong data. The data points may be input twice or may be missed with no indication.

When all the data files and log file are ready, the .EXE file of the FORTRAN GEOS 7.0 will be run, as follows (In fact, this is the only interface between the program and users):



Figure 2. The interface for GEOS 7.0

Then the log file name is typed in. After the calculation is completed, check the output file, such as out.put, out2.put, stat5.put to view the results. Then transfer the files that needs to be plotted to such software as MS Excel, change the format, and plot.

The above summarizes the steps to use GEOS 7.0. In the next chapter, the GUI and database will be designed to help ease the burden of the preparation process by visualizing the content of the options in the log file and by automatically preparing the data file and plots.

```
ANALYSIS OF EQUILIBRIUM DATA USING PENG-ROBINSON EOS
4,0,0,0,0      NF,MODE,IGN,IPRP,IDB
0,1,1,500     INTL,ICD,ITRVL,NITR
0,0,0         IHEN,IVT,IPF
1,-3,20,20    IW,NTRAC,MAXIT, MAXSUB
5,1,5,1,2,8   IOPTN,METHOD,IEOS,MODEL,IFUGR,IVPM
1,1,0,0,0     INPT,IUNIT,IRIT,IHS,IGPA
0,0,0,0,1 0,0,0 0,0 0,0,0,0      CASES
1,0, 0,0,0,0, 0,0,0,0 0,0 PRNT OPTS: P,Y,L,V,HL,HV,K1,K2,A1,A2,S1,S2
0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
Hc10.dat
Hc20.dat
Hc28.dat
Hc36.dat
```

Figure 3. Example Log File

Oklahoma State University

HYDROGEN+N-DECANE

```

0 5 1 4 1 3 2 0 0
2 1 7 1 1 0 0 1 0 8
H2      33.2000  13.0000  -.2180  20.4000  .3060  2.0200  1
N-decane 618.8000 21.1400  0.4890  447.300  .2490  142.286  10
H2      13.6333 164.9000  3.1900  .0000  .0000
N-C10   16.0114 3456.8000 -78.6700 .0000 .0000
.000000D+00 .000000D+00 .000000D+00 .000000D+00 .000000D+00
.000000D+00 .000000D+00
.000000D+00 .000000D+00 .000000D+00 .000000D+00 .000000D+00
.000000D+00 .000000D+00
-0.30000
.00000
.00000 .00000
.00000 .00000
1.00000 1.00000 1.00000 1.00000 1.00000
1.00000 1.00000 1.00000 .00000 1.00000
0 0 1 1 1 1 1 1 1 1 1 1
344.26 44.600 .036690 14 8 1.0000 .0000 .0000
344.26 71.300 .057600 14 8 1.0000 .0000 .0000
344.26 86.000 .068200 14 8 1.0000 .0000 .0000
462.45 29.536 .040800 14 9 .9446 .0000 .0000
462.45 50.055 .067800 14 9 .9676 .0000 .0000
462.45 255.24 .283200 14 9 .9891 .0000 .0000
503.35 19.414 .028900 14 9 .8021 .0000 .0000
542.95 202.35 .324900 14 9 .9391 .0000 .0000
542.95 255.14 .382500 14 9 .9446 .0000 .0000
583.45 50.561 .105900 14 9 .6355 .0000 .0000
583.45 101.02 .222000 14 9 .7825 .0000 .0000
583.45 152.09 .324700 14 9 .8303 .0000 .0000
583.45 204.37 .409800 14 9 .8505 .0000 .0000
583.45 246.93 .501300 14 9 .8581 .0000 .0000

```

[8] Park, J. K., Ph. D. Dissertation, 1993. Binary Vapor-Liquid Equilibrium Measurements for Selected Assymmetric Mixtures and Equation of State Development, Oklahoma State University, Stillwater, Oklahoma.

[9] Sebastian, H. M., Simnick, J. J., Lin, H. M., and Chao, K. C., 1980. Gas-Liquid Equilibrium in the Hydrogen + n-Decane Systems at Elevated Temperatures and Pressures, J. Chem. Eng. Data, 25: 68-70.

Figure 4. Example Data File

CHAPTER IV

PROJECT DEVELOPMENT

4.1 Database Design

Data are needed each time to prepare a data file as input file to GEOS 7.0. As manual input is not efficient and is error prone, efforts will be made to take advantage of the development of database to solve this problem.

A good relational database design is to generate a set of relation schemes that provide us with the following properties (Lu, 1998):

1. The redundancy in the stored data (information) is reduced to a minimum (by normalization theorems) to save storage space and to simplify updating (insertion, deletion, and updating) processes. The simplified updating process should result in easy maintenance of data integrity;
2. The dependencies between attributes are preserved;
3. The lossless join properties are maintained during decomposition, such that the data (information) can be easily retrieved without distortion.

These principles will be followed to design the current database. The normalization theorems, correct (independent) decompositions of relations, and methods to check whether the set of schemes provides a lossless join property, will be implemented in this study.

4.1.1 Pure Component Relation Design

The properties needed for each pure compound are: Component name (CName), Tc, Pc, w, Tb, Zc, MW, Reference for physical properties (Ref); vapor pressure equation (VPE), VPE coefficients (assume max 7 of them, Ant_1, Ant_2, Ant_3, Ant_4, Ant_5, Ant_6, Ant_7), VPE Reference (RefVPE); enthalpy equation (HE), HE equation coefficients (assume max 7 of them, HC_1, HC_2, HC_3, HC_4, HC_5, HC_6, HC_7), Reference for HE (RefHE); Temperature, Pressure, Volume, Density of liquid, Density of vapor, Phase, Source. Those properties will consist of the information for a pure component needed for the GEOS 7.0 data file.

The relation in the 1st normal form will be:

Relation r(R) with R = { CName, Tc, Pc, Zc, Tb, MW, Ref, VPE, Ant_1, Ant_2, Ant_3, Ant_4, Ant_5, Ant_6, Ant_7, VPERef, HE, HC_1, HC_2, HC_3, HC_4, HC_5, HC_6, HC_7, RefHE, T,P,V,DV,DL,PH,S}

Functional Dependencies F: { CName, Ref-> Tc, Pc, ω , Zc, Tb, MW; CName, VPE, VPERef-> Ant_1, Ant_2, Ant_3, Ant_4, Ant_5, Ant_6, Ant_7; CName, HE, RefHE -> HC_1, HC_2, HC_3, HC_4, HC_5, HC_6, HC_7; CName, S, T, P -> V, DV,DL,PH}

The primary key for the above relation is CName, Ref, VPE, VPERef, HE, RefHE, S, T, P. However, problems exist for this design: (1) redundancy is stored in the data, (2) more work to do in updating process. Therefore, reduction to 2nd normal form is needed. A relation is in the second normal form if and only if it is in the first normal form and every nonkey attribute is irreducibly dependent on the primary key (Date, 1995). Therefore, we

Okajima S. 1995

will eliminate the non-irreducible dependencies as follows. We decompose the above relation into r1 (R1), r2 (R2), r3(R3) and r4 (R4).

Relation r1(R1) with R1 = { CName, Tc, Pc, ω , Zc, Tb, MW, Ref }

Functional Dependencies F1: { CName, Ref-> Tc, Pc, ω , Zc, Tb, MW }

Relation r2(R2) with R2 = { CName, VPE, VPERef, Ant_1, Ant_2, Ant_3, Ant_4, Ant_5, Ant_6, Ant_7, }

Functional Dependencies F2: { CName, VPE, VPERef -> Ant_1, Ant_2, Ant_3, Ant_4, Ant_5, Ant_6, Ant_7 }

Relation r3(R3) with R3 = { CName, HE, HERef, HC_1, HC_2, HC_3, HC_4, HC_5, HC_6, HC_7, HCRF }

Functional Dependencies F3: { CName, HE, HERef -> HC_1, HC_2, HC_3, HC_4, HC_5, HC_6, HC_7 }

Relation r4(R4) with R4 = { CName, S, T, P, V, DV, DL, PH }

Functional Dependencies F4: { CName, S, T, P -> V, DV, DL, PH }

Now the large relation in 1st normal form for the pure component becomes four small relations in 2nd normal form. These relations are enough to represent the pure component effectively. According to the definition of the 3rd Normal Form, "a relation is in the 3rd normal form (NF) if and only if it is in the second normal form and every nonkey attribute is non-transitively dependent on the primary key." It is easy to prove that those four relations are also in 3rd NF. No further efforts will be made to decompose the relations.

4.1.2 Binary System Relations Design

The properties needed for a mixture are: component1 (C1), component2 (C2), Temperature (T), Pressure (P), System_ID (SID), Reference_ID (RID), composition 1 in liquid phase (X1), composition 1 in vapor phase (Y1), vapor phase density (DV), liquid phase density (DL).

The 1st NF of this relation will be:

Relation r(R) with $R = \{ C1, C2, T, P, SID, RID, X1, Y1, DV, DL \}$

Functional Dependencies F: $\{ SID, RID, T, P \rightarrow X1, Y1, DV, DL; SID \rightarrow C1, C2 \}$

Decompose it into 2nd NF:

Relation r1(R1) with $R1 = \{ C1, C2, SID \}$

Functional Dependencies F1: $\{ SID \rightarrow C1, C2 \}$

Relation r2(R2) with $R2 = \{ T, P, SID, RID, X1, Y1, DV, DL \}$

Functional Dependencies F2: $\{ SID, RID, T, P \rightarrow X1, Y1, DV, DL \}$

These relations are enough to represent binary systems effectively. As can be proved, these relations are also in 3rd NF. No further efforts will be made to decompose the relations.

4.1.3 Reference Relation Design

For each reference relation, the needed properties are: Title, AID_1 (Author ID), AID_2, AID_3, AID_4, Journal Name, Date, Reference ID (RID). (Assume maximum of four Authors).

The relation is: { RID, Title, AID_1, AID_2, AID_3, AID_4, Journal Name, Date }

The functional dependence is: { RID -> Title, AID_1, AID_2, AID_3, AID_4, Journal Name, Date }

For the author relation, the needed properties are: First Name, Last Name, AID.

The relation is: { AID, First Name, Last Name }

The functional dependence is: { AID -> First Name, Last Name }

The above relations are in the 3rd normal form according to the definition. No further decomposition is needed.

The database for this study was named *Asymmetric_Systems.mdb*, using the relations and functional dependence designed above. A copy of the relationship in Access 7.0 is shown in Figure 5. A Visual Basic program named *AddData* was developed to insert the data in the flat text file format used in GEOS 7.0 to the database or convert from one database to another database. This database, *Asymmetric_Systems*, will be hooked up in the GUI to provide data to the data file.

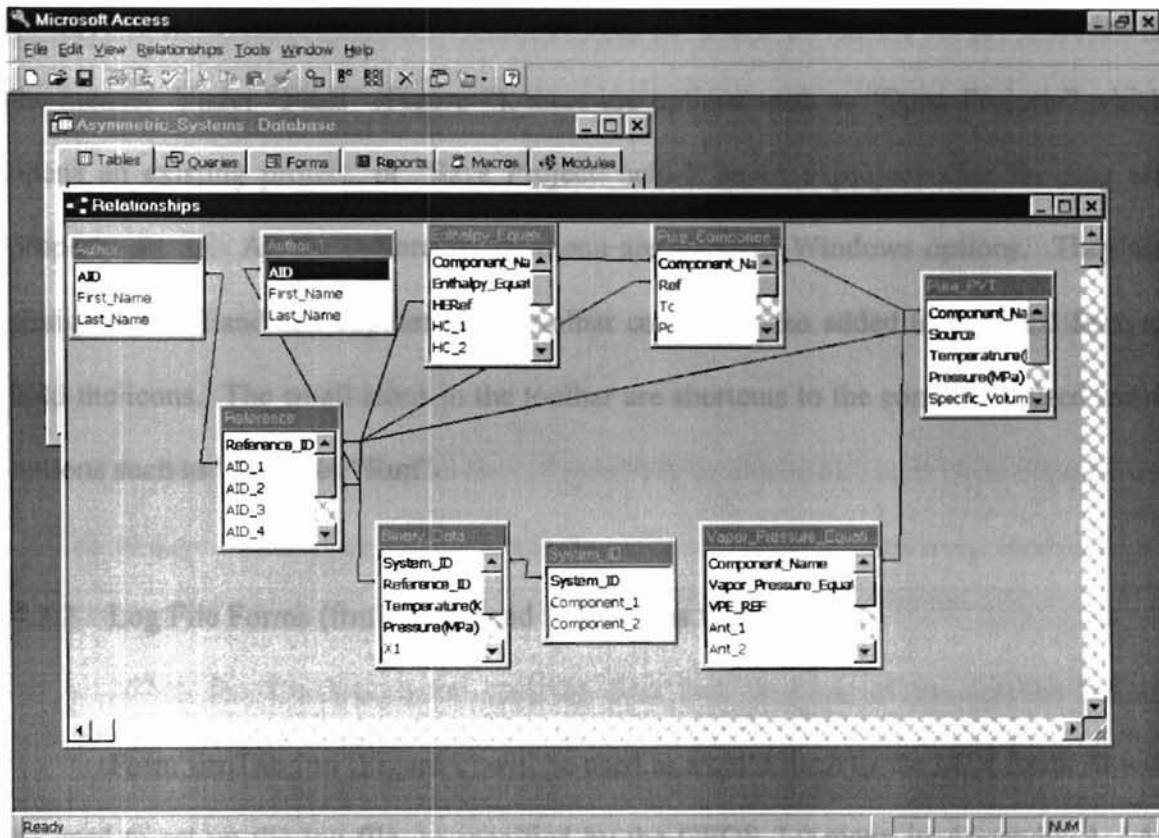


Figure 5. Relationship of the Designed Database

4.2 GUI Design

4.2.1 Main Menu Design (MDI.frm)

The Main menu will be the central control form. It will be the main form users deal with. It decides where the program control goes. It will be as shown in Figure 6. Multiple Document Interface (MDI) in Visual Basic is used to build this form. It will be a parent form and all other forms will be either single form or MDI child forms that

attached to it. Standard Windows features such as "FILE", "View", etc are included in the menus. Under "FILE" (Figure 7), there are options such as "Open Projects" which opens an existing project, or "Save Project" which saves a project after the flag are properly set up. All the options in the menu are standard Windows options. They are straightforward and self-explained. A toolbar control is also added to the MDI form to hold the icons. The small icons in the toolbar are shortcuts to the commonly used menu options such as "Save" or "Run".

4.2.2 Log File Forms (frmTab.frm and frmOptions.frm)

Form frmTab.frm (Figure 8) will be used as a child form to the MDI form. It will be used to set up the log file, as specified by the GEOS 7.0 menu in Appendix A. As there are many flags that need to be set, a standard control in Visual Basic, SSTab, will be used to present the choices. The SSTab control provides an easy way of presenting several dialogs or screens of information on a single form using the same interface seen in many commercial Microsoft Windows applications. The SSTab control presents a group of tabs, each of which acts as a container for other controls. Only one tab is active in the control at a time, displaying the controls it contains to the user while hiding the controls in the other tabs. Using the properties of this control, programmers can: (a) determine the number of tabs, (b) organize the tabs into more than one row, (c) set the text for each tab, (d) display a graphic on each tab, (e) determine the style of tabs used, (f) set the size of each tab. The first four major sections on the selected SSTab control are divided according to the menu for the log file. In each major section, for example, the

“Properties” section, frames will be used to separate each option of the properties, with each frame corresponding to a single number in the text version log file. For example, in the “calculate” frame, all the options are shown with detailed explanation. The users just click on the one that is appropriate, instead of using numbers as specified in the GEOS 7.0 menu and log file. The interface for the model section in the log file is shown in Figure 9.

Another form, frmOptions.frm (Figure 10), is connected, as a child form, to the SSTab to give more space to show options. The settings are similar to what is in the frmTab.frm.

When the frmOptions.frm and the first four sections of the frmTab.frm are selected, the log file will be finished except for the data files. Then users will click on the “Project Summary” to view what has been selected (Figure 11). A project title needs to be typed in. Then users can insert an existing data file by clicking “Insert”, or a new data file can be created (details are reviewed in the next section) by clicking on the “Create” button. Users can remove a data file from the project by highlighting a data file and then clicking on the “Remove” button to remove the file from the log file. When a user clicks on “Insert”, a standard dialog window (Figure 12) will pop up to show the available data file. When a log file is completed, user can view the log file by browsing through “Summary” in the project summary window or can run it by clicking on the “Run” icon. Only after the calculations are successfully completed will the “Results” be undimmed.

Ukiyama 04/11

4.2.3 Data File Form (frmDataFile.frm)

A new data file can be created by clicking on “Create” button in Figure 11. Then the frmDataFile.frm (Figure 13.) will pop up. A SSTab control is used in this form. It has three sections. The first one is the “Component Selection”. A listbox control in Visual Basic is used to list all the available components in the database. A user can select a component by clicking on the component directly. If a user wants to select a second component, press Ctrl then click on the component. This is the standard operation for all Windows applications to select multiple components. For each selected component, the physical properties will be displayed in a gridbox control in Visual Basic. If the selected binary is not in the database, a message box will pop up to remind the user (Figure 14) and then let the user select a system again. The user can also specify the temperature and pressure range. After a binary system is selected, the user can go to the next section “Data View” to view the selected binary data. The last section is the “Model Parameter” section. The user can view the default initial values of the model parameters. The user can also change an initial value by changing the value in the text box. When all these steps are done, the user clicks on the “Save” button to save the data file selected. Then control goes back to the “Project View” section in frmTab.frm form. All the connections with the Access 7.0 database will be implicit during the process to prepare the data file. The users cannot manipulate the database directly.

4.2.4 Other Forms (frmAbout.frm, frmAbout1.frm)

Forms frmAbout.frm and frmAbout1.frm are used to show some general information about this Visual Basic version of GEOS 7.0. Several Label controls are used express the contents. These two forms are linked to the Help menu to show information about this program.

4.2.5 Design Of The Automatic Plotting

While running the GEOS itself is not time-consuming, it is for plotting the results. It is desirable to plot the results automatically. There are two kinds of software available during this study that can be connected to Visual Basic. One is the Graphics Servers from Pinnacle Software. Another is Microsoft Excel. Excel was chosen because: (1) most of the users are familiar with the Excel; (2) after a plot is made automatically, user can easily customize it according to his own needs; (3) it provides all the necessary plotting functions; (4) it is widely available on the platform Visual Basic Program runs.

The properties that need to be plotted are written into a file called graph.txt. One sample file is shown in Figure 15. This is a "0" in the first line before each system to signal the beginning of a new system. Then, line 2 has three numbers to show how many systems have been chosen in this run, which option and which case have been chosen to run the data for the following system. The next two lines (3 and 4) show the component name and physical properties for each component. The rest of the lines shows the input data and the corresponding calculated properties for the component. The first data in

each line of this section is the ID number for this data point, followed by temperature, pressure, mole fraction of the component in the liquid, calculated properties (different depending on the chosen options and cases), absolute deviation of the calculated properties and percent deviation. If the ID number is “-1”, the data on the line are temperature, interaction parameters (C_{ij} and D_{ij}) and the percent deviation for this isotherm or system depending on which case is running on this system.

When the calculation is completed, the “Results” section will be available. If a user click on the “Plot” button, Excel application will pop up. Then Graph.xls should be opened. The interface is shown in Figure 16. User can click on the “Open A File” button on the interface to choose a file to be plotted (Figure 17). The available systems, cases, and plots for this file will be shown in Figure 18. The users can highlight on the system, case, and plot in the “Listbox” controls, then click on the button “Plot The Chosen Plot”. The default name for the plot is “t1.xls”. If the users want to save this file, the file can be saved to a different name and directory. Then the users can change the plot any way they want like they normally do with Excel files. Figure 19 and 20 show two plots made using this program. If the users want to plot all the figures, they can click on the button “Plot All Plots”, then all the plots available will be plotted and saved in a file named “t1.xls”.

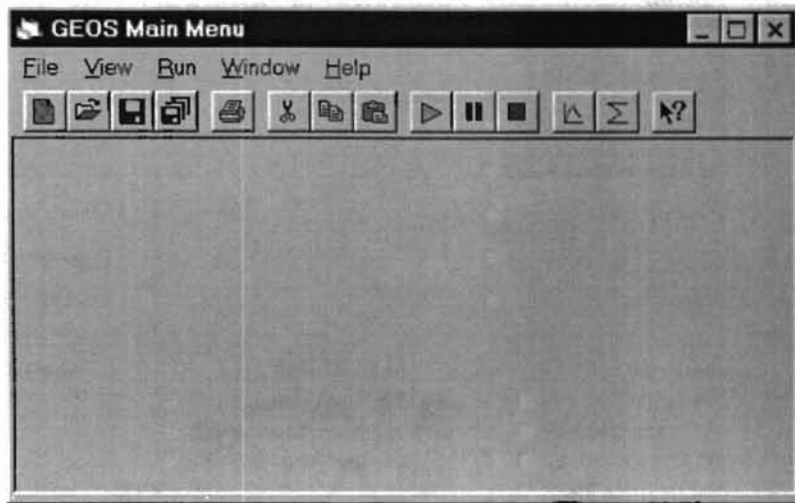


Figure 6. GEOS Menu

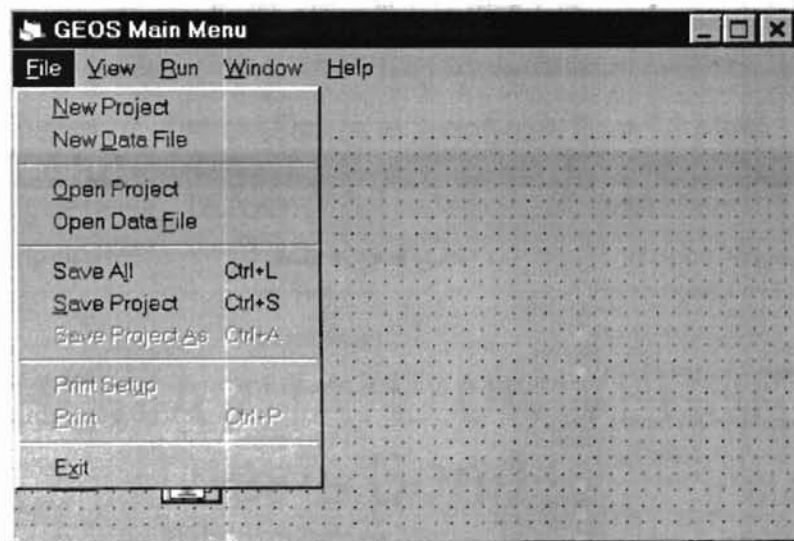


Figure 7. Items Under File in the Menu Bar

Prithvima Chit...

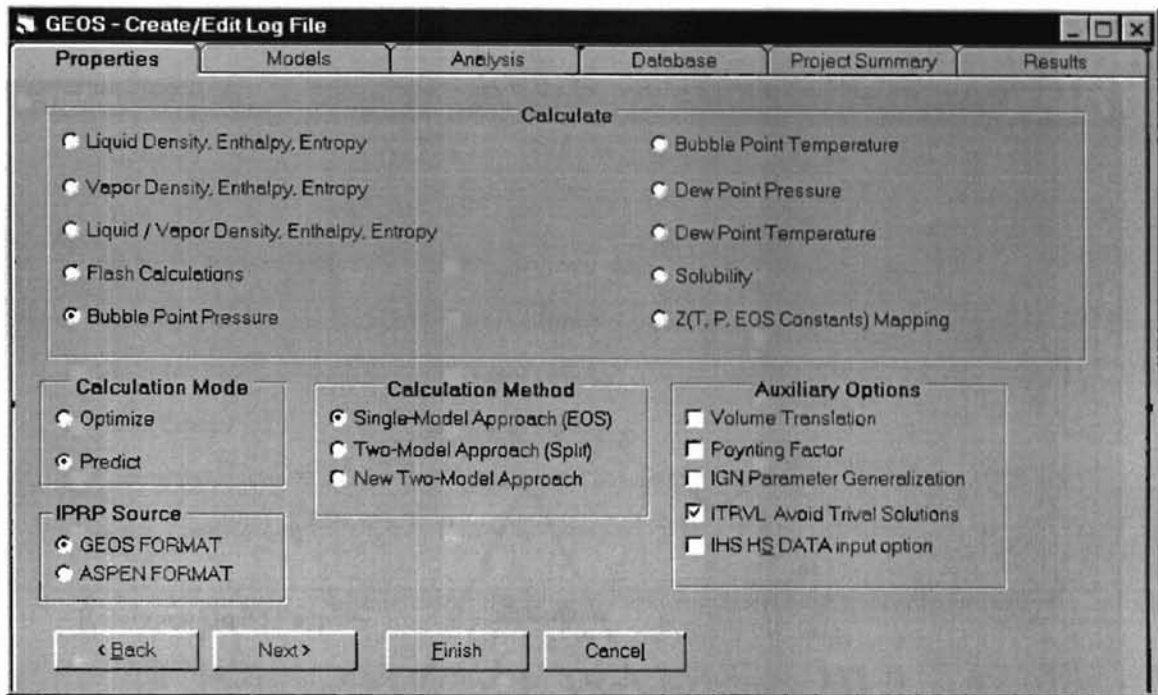


Figure 8. The First Tab in SSTab Control

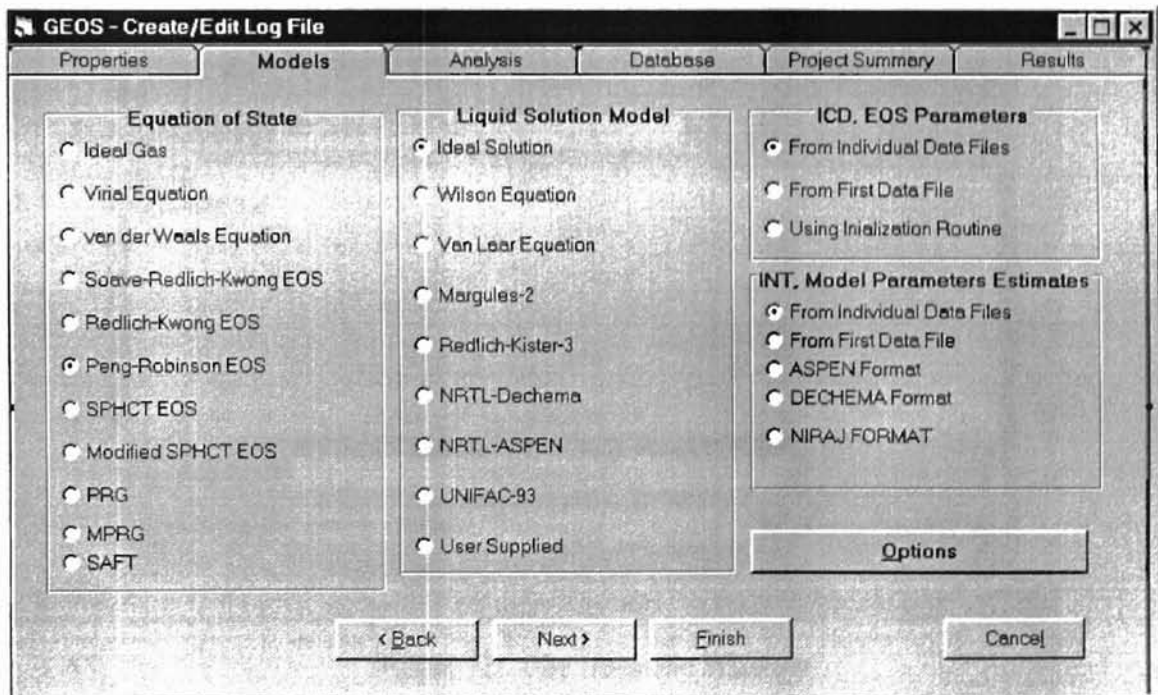


Figure 9. Models Section as Specified in Log file

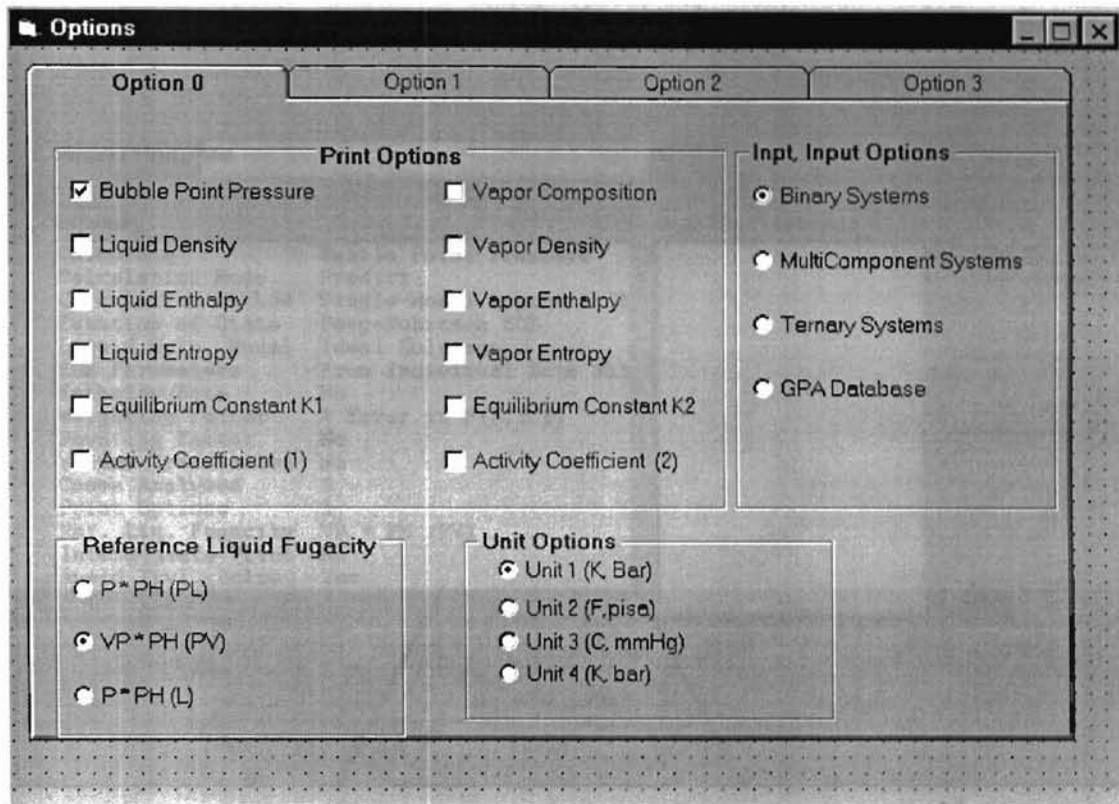


Figure 10. Options Section as Specified in Geos 7.0 Menu

Pritham Ch...

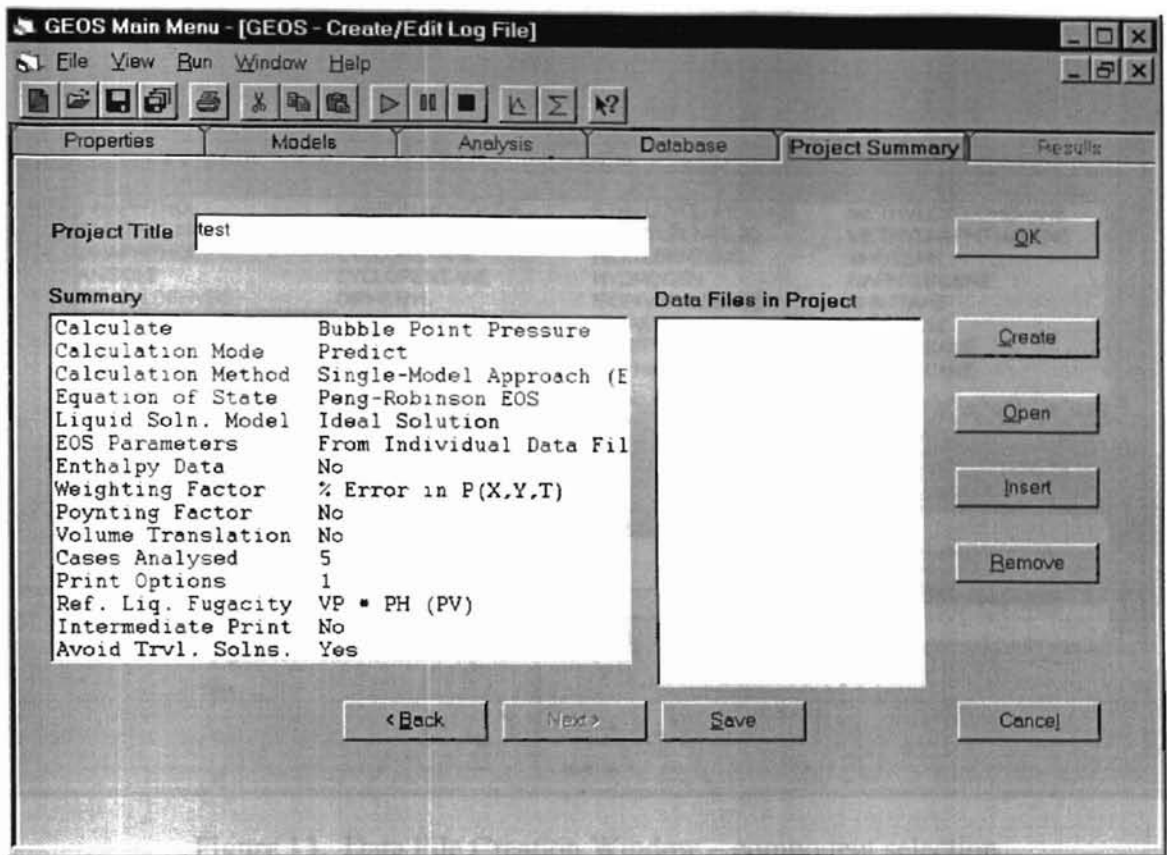


Figure 11. Project Summary Window

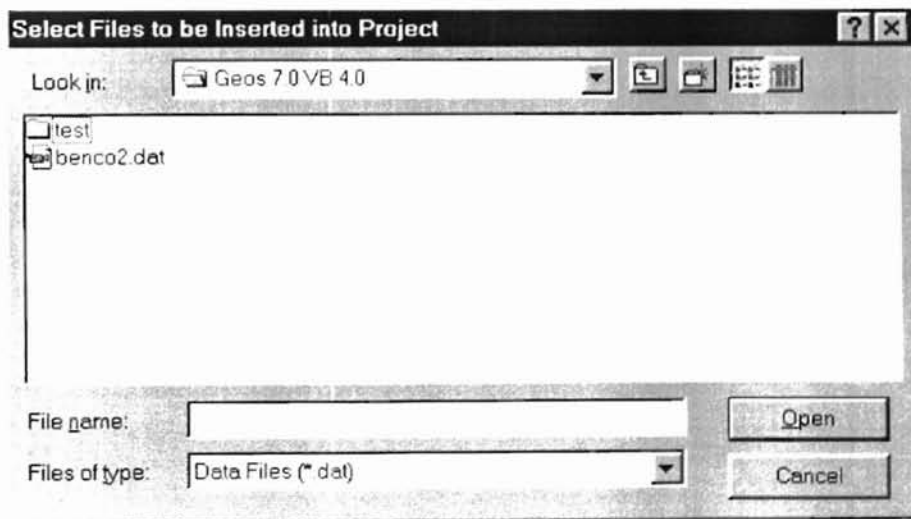


Figure 12. File Insertion Window

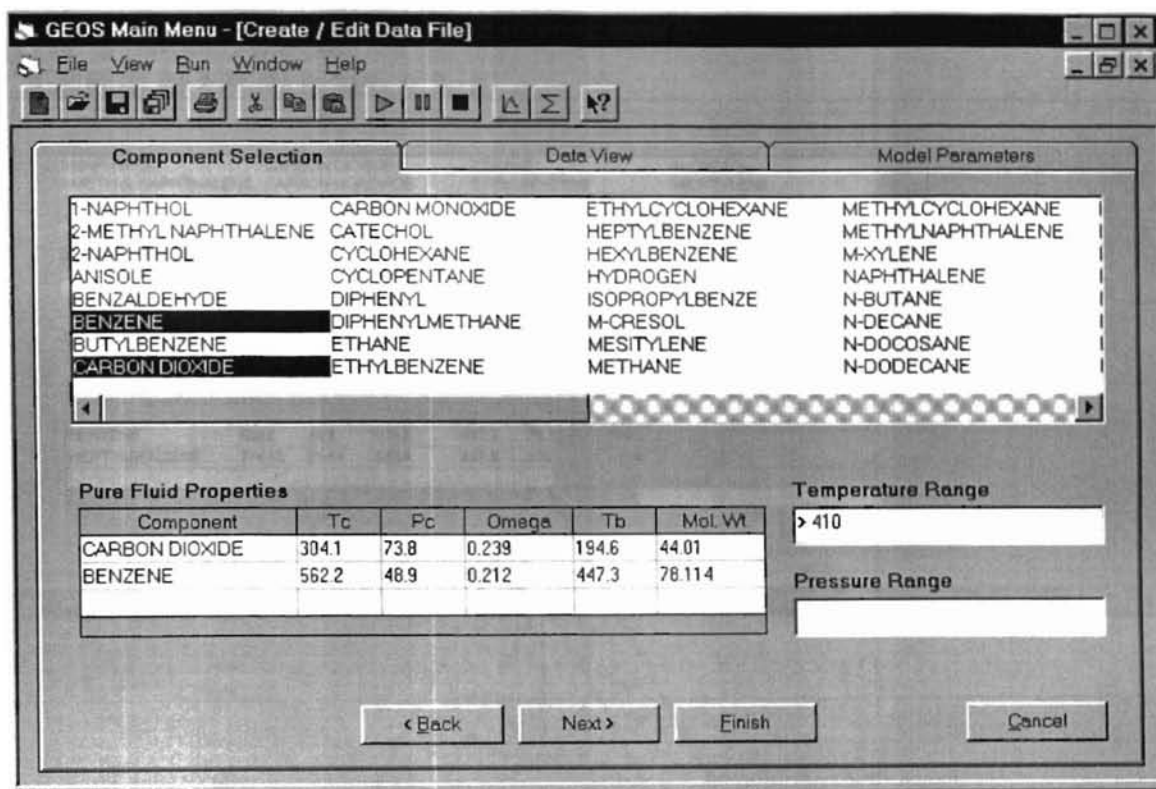


Figure 13. Data File Creation Window – component selection

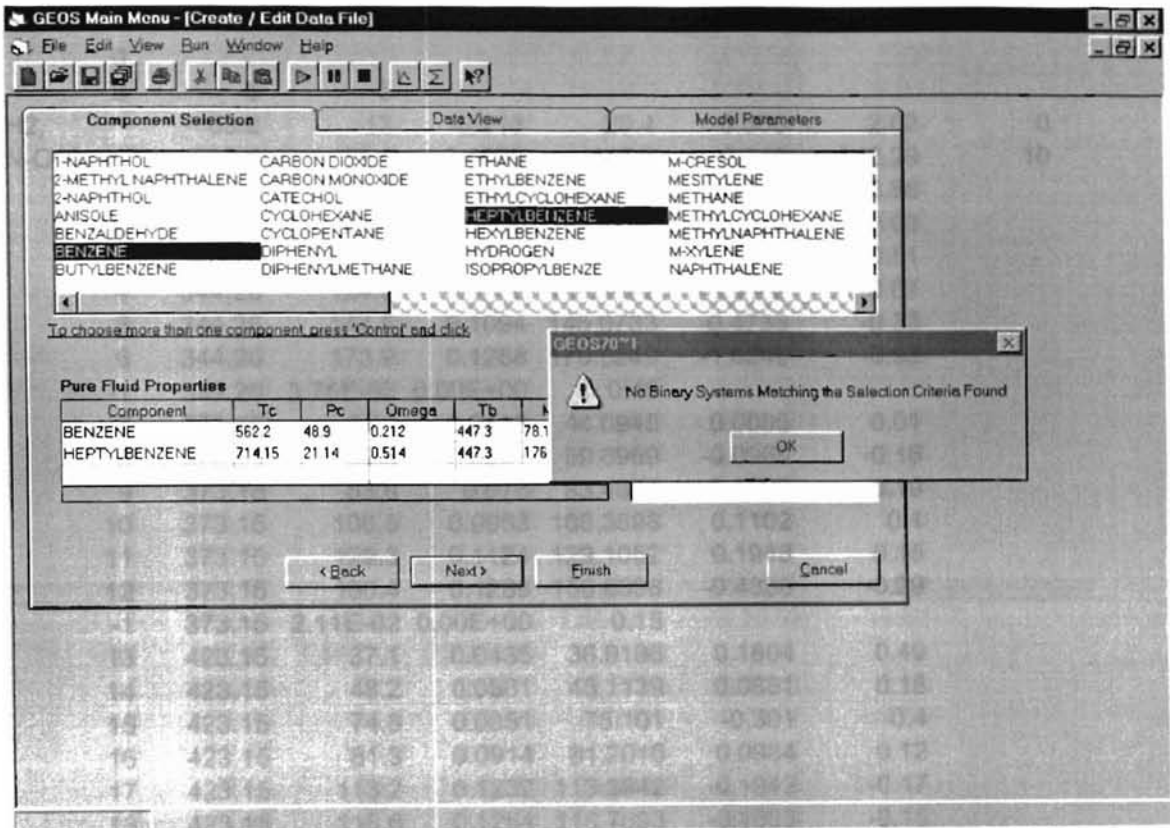


Figure 14. Prompt If the Selected System Is Not In the Database

	0						
	3	5	5				
H2,	33.2	13	-0.218	20.4	0.306	2.02	0
N-C10,	618.6	21.4	0.4882	447.3	0.249	142.29	10
1	344.26	44.6	0.0367	44.2181	0.3819	0.86	
2	344.26	71.3	0.0576	71.2763	0.0237	0.03	
3	344.26	86	0.0682	85.56	0.44	0.51	
4	344.26	124.6	0.0958	124.6882	-0.0882	-0.07	
5	344.26	144.6	0.1094	145.0733	-0.4733	-0.33	
6	344.26	173.9	0.1288	175.5249	-1.6249	-0.93	
-1	344.26	3.76E-02	0.00E+00	0.46			
7	373.15	44.1	0.0418	44.0945	0.0055	0.01	
8	373.15	59.6	0.0557	59.6969	-0.0969	-0.16	
9	373.15	83.6	0.076	83.4374	0.1626	0.19	
10	373.15	108.5	0.0963	108.3898	0.1102	0.1	
11	373.15	129.3	0.1124	129.1052	0.1948	0.15	
12	373.15	150.4	0.1286	150.8336	-0.4336	-0.29	
-1	373.15	2.11E-02	0.00E+00	0.15			
13	423.15	37.1	0.0435	36.9196	0.1804	0.49	
14	423.15	48.2	0.0561	48.1139	0.0861	0.18	
15	423.15	74.8	0.0851	75.101	-0.301	-0.4	
16	423.15	81.3	0.0914	81.2016	0.0984	0.12	
17	423.15	113.2	0.1232	113.3942	-0.1942	-0.17	
18	423.15	116.6	0.1264	116.7693	-0.1693	-0.15	
-1	423.15	2.69E-02	0.00E+00	0.25			
	0						
	3	5	5				
H2,	33.2	13	-0.218	20.4	0.306	2.02	0
N-C20,	766.9	10.94	0.8883	617	0.2281	282.56	0
1	323.15	32.6	0.032	32.3088	0.2912	0.89	
2	323.15	34	0.0333	33.6763	0.3237	0.95	
3	323.15	67.7	0.0644	67.7771	-0.0771	-0.11	
4	323.15	70.2	0.0663	69.9507	0.2493	0.36	
5	323.15	105.1	0.0964	105.8937	-0.7937	-0.76	
6	323.15	107.1	0.0978	107.6377	-0.5377	-0.5	
7	323.15	129.1	0.1152	129.8874	-0.7874	-0.61	
-1	323.15	7.64E-02	0.00E+00	0.6			
8	373.15	22.3	0.0273	22.2235	0.0765	0.34	
9	373.15	24.1	0.0296	24.1811	-0.0811	-0.34	
10	373.15	30.9	0.0371	30.5614	0.3386	1.1	
11	373.15	58.1	0.0686	58.5682	-0.4682	-0.81	
12	373.15	67.3	0.0776	66.9496	0.3504	0.52	
13	373.15	70.1	0.0811	70.257	-0.157	-0.22	
14	373.15	86.9	0.0989	87.5087	-0.6087	-0.7	
15	373.15	104	0.1147	103.4538	0.5462	0.53	
16	373.15	118.2	0.1289	118.3223	-0.1223	-0.1	

Figure 15. Sample Design of The Graph.Txt File (to be continued)

-1	373.15	-7.19E-03	0.00E+00	0.52			
17	423.15	28.1	0.041	27.9262	0.1738	0.62	
18	423.15	39.7	0.0573	39.6964	0.0036	0.01	
19	423.15	53.3	0.0756	53.4014	-0.1014	-0.19	
20	423.15	62.4	0.0874	62.528	-0.128	-0.21	
21	423.15	77.5	0.1064	77.7267	-0.2267	-0.29	
22	423.15	93	0.1246	92.9008	0.0992	0.11	
-1	423.15	-9.88E-02	0.00E+00	0.24			
0							
3	5	5					
H2,	33.2	13	-0.218	20.4	0.306	2.02	0
N-C30,	855.9	7.94	1.2883	617	0.2281	282.56	0
1	323.15	32.6	0.032	32.3088	0.2912	0.89	
2	323.15	34	0.0333	33.6763	0.3237	0.95	
3	323.15	67.7	0.0644	67.7771	-0.0771	-0.11	
4	323.15	70.2	0.0663	69.9507	0.2493	0.36	
5	323.15	105.1	0.0964	105.8937	-0.7937	-0.76	
6	323.15	107.1	0.0978	107.6377	-0.5377	-0.5	
7	323.15	129.1	0.1152	129.8874	-0.7874	-0.61	
-1	343.15	7.64E-02	0.00E+00	0.6			
8	373.15	22.3	0.0273	22.2235	0.0765	0.34	
9	373.15	24.1	0.0296	24.1811	-0.0811	-0.34	
10	373.15	30.9	0.0371	30.5614	0.3386	1.1	
11	373.15	58.1	0.0686	58.5682	-0.4682	-0.81	
12	373.15	67.3	0.0776	66.9496	0.3504	0.52	
13	373.15	70.1	0.0811	70.257	-0.157	-0.22	
14	373.15	86.9	0.0989	87.5087	-0.6087	-0.7	
15	373.15	104	0.1147	103.4538	0.5462	0.53	
16	373.15	118.2	0.1289	118.3223	-0.1223	-0.1	
-1	373.15	-9.19E-03	0.00E+00	0.52			
17	423.15	28.1	0.041	27.9262	0.1738	0.62	
18	423.15	39.7	0.0573	39.6964	0.0036	0.01	
19	423.15	53.3	0.0756	53.4014	-0.1014	-0.19	
20	423.15	62.4	0.0874	62.528	-0.128	-0.21	
21	423.15	77.5	0.1064	77.7267	-0.2267	-0.29	
22	423.15	93	0.1246	92.9008	0.0992	0.11	
-1	423.15	-9.88E-02	0.00E+00	0.24			
0							

Figure 15. Sample Design of The Graph.Txt File (continued)

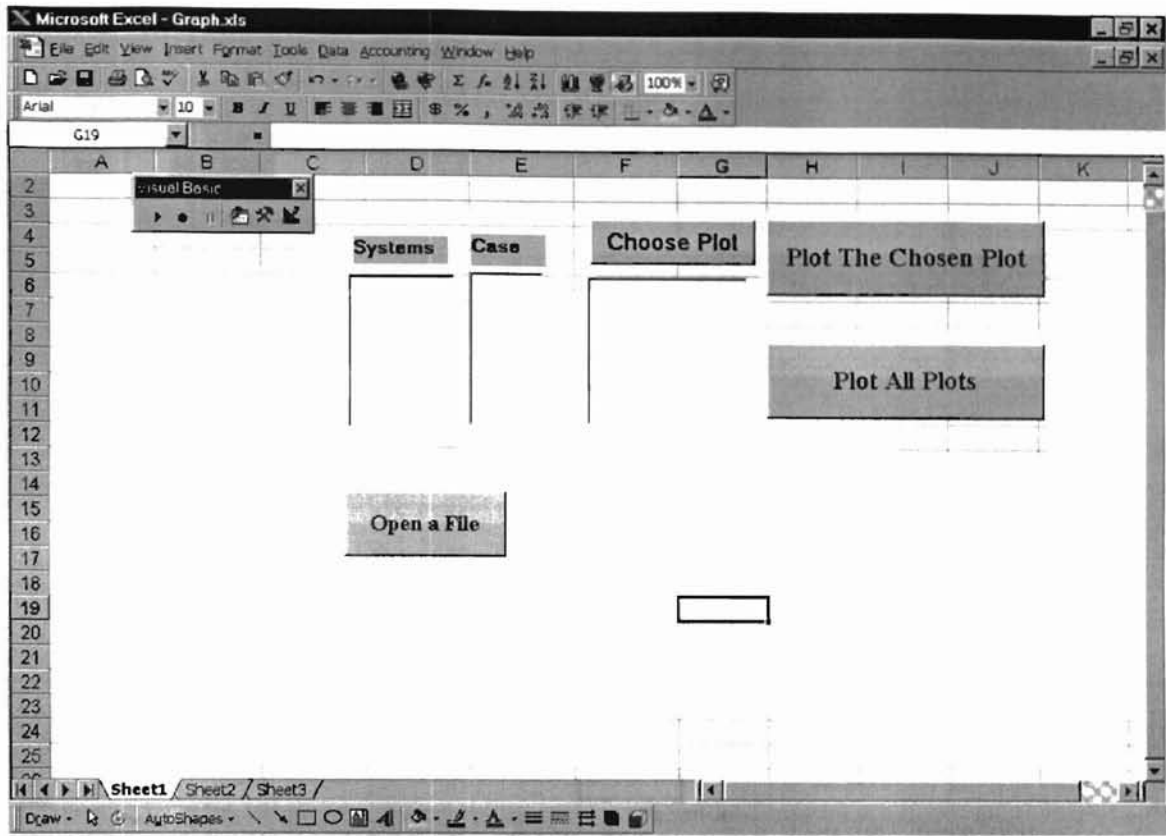


Figure 16. Interface of Automatic Plotting

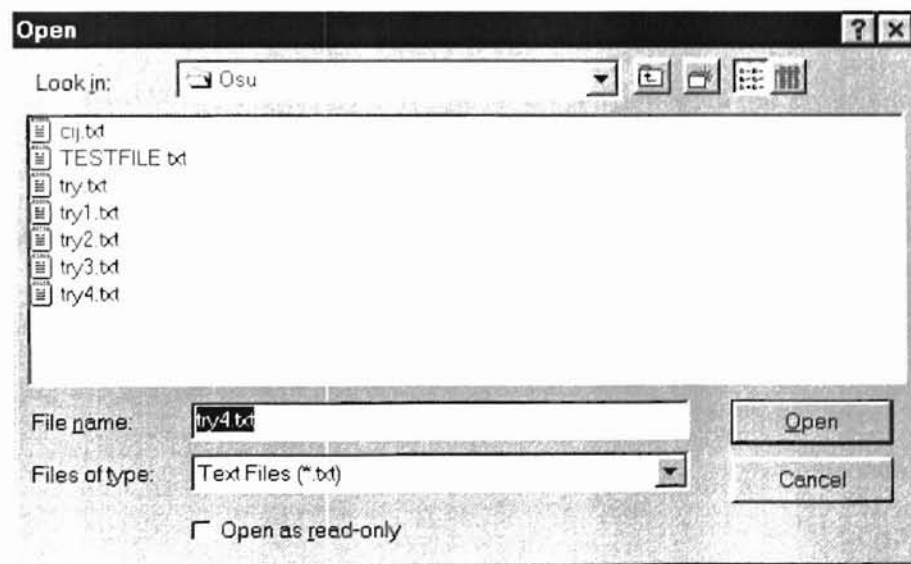


Figure 17. Dialog To Open A Proper File To Plot

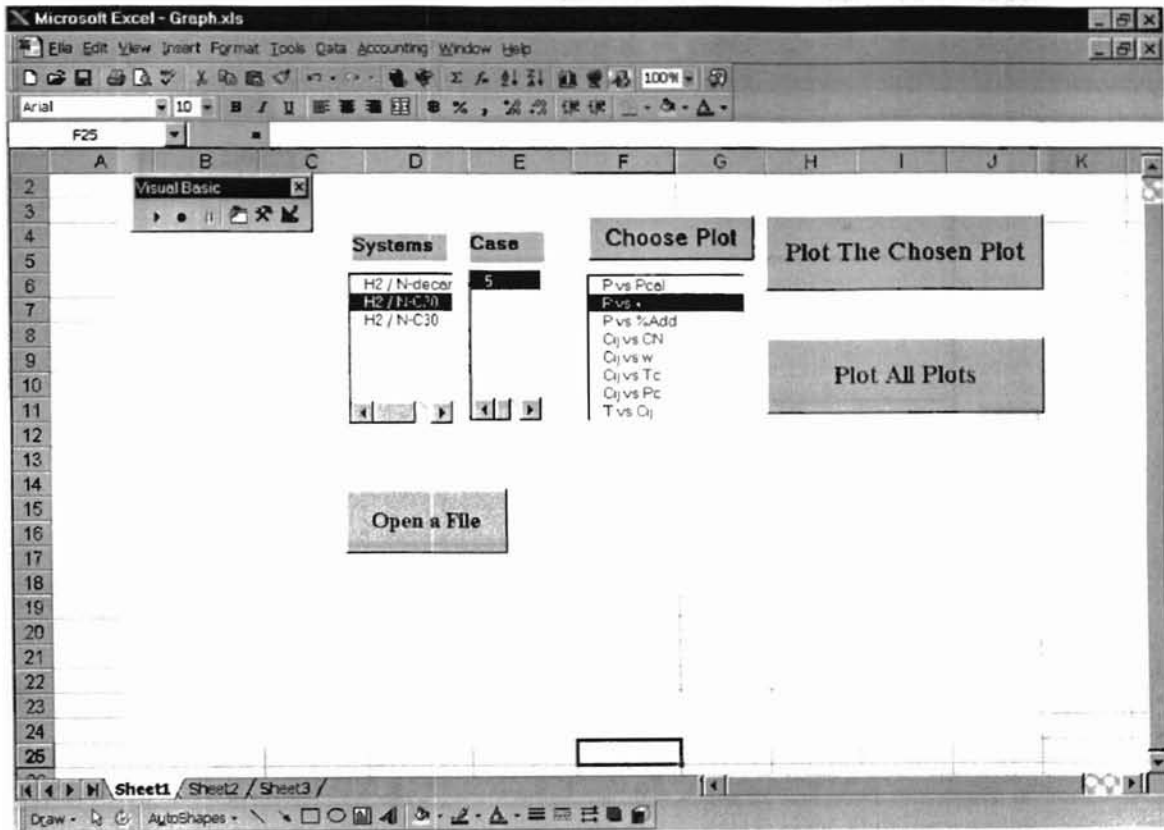


Figure 18. Interface To Show Available Choice

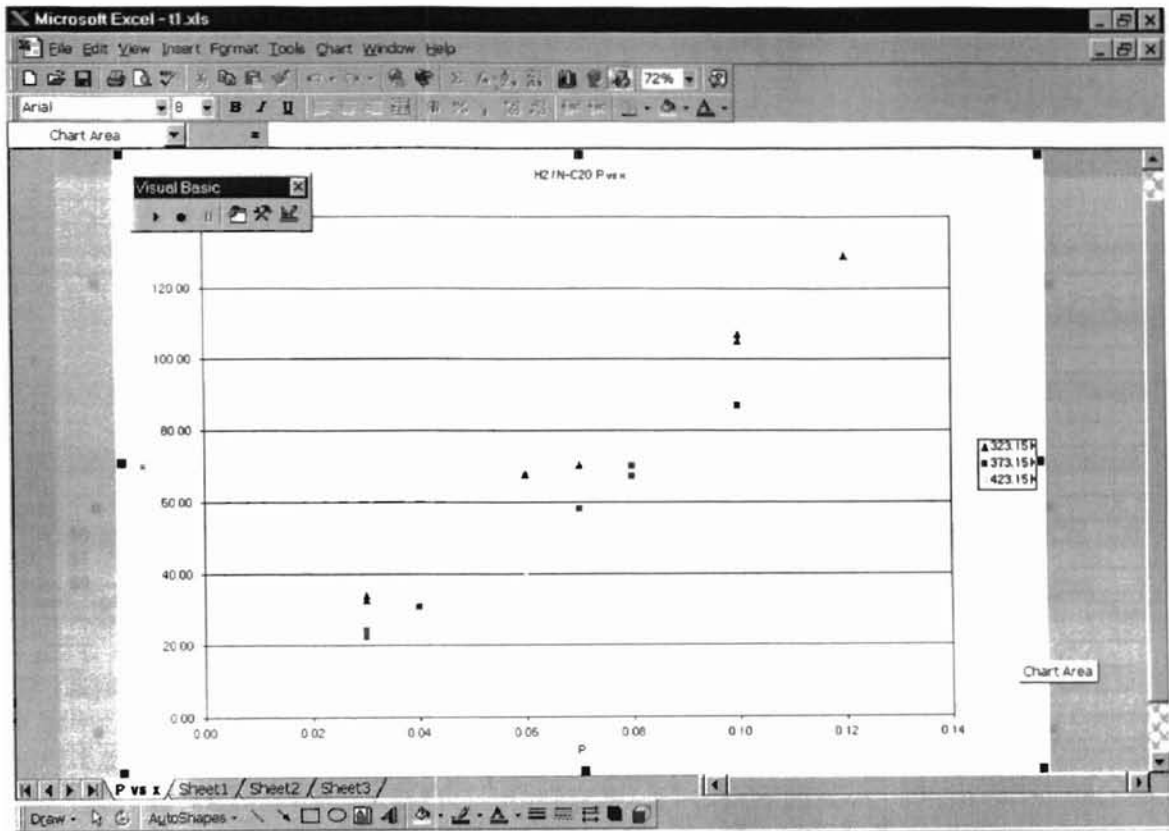


Figure 19. Sample Plot of X vs P For H2 / N-C20 System

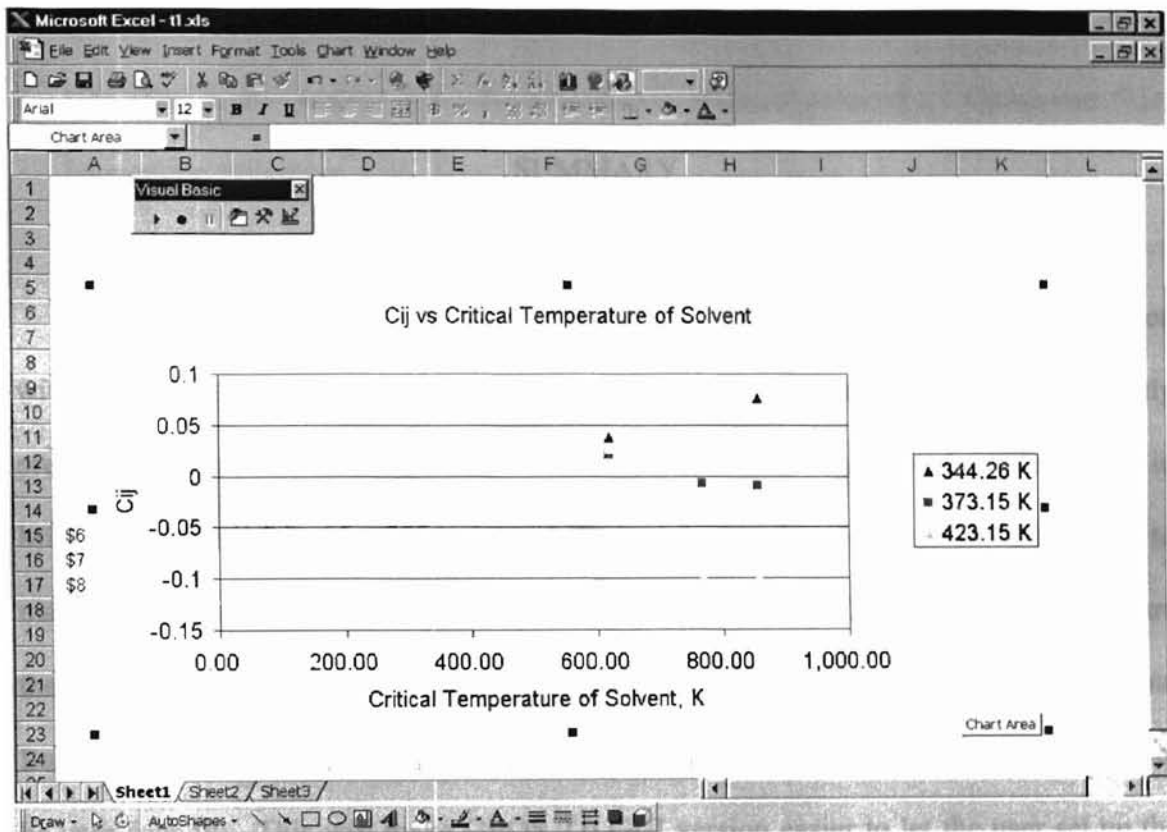


Figure 20. Sample Plot of C_{ij} vs T_c Across The Systems

CHAPTER V

SUMMARY

In this study, GEOS 7.0 was reviewed and the interfacing problems associated with it were analyzed. To solve the problem of properly keeping the data and efficiently preparing the input data file, a relational database was designed and implemented in Microsoft Access 7.0. A GUI in Visual Basic 4.0 was also designed for GEOS 7.0 to make it user-friendly. The database was integrated into the GUI to automatically prepare the data files by using the Microsoft Jet Engine and embedding SQL code into the Visual Basic application. The GUI design uses the general Windows features that are proved to be user-friendly. This will make GEOS 7.0 GUI version easier to let the user set up the log file. Graphing functions are also added for certain output files.

The following works are recommended if this study will be continued:

1. The GUI will be updated to Visual Basic 6.0 and use the new features to add it to the web. It will be a client and server system.
2. Other graphing software can be incorporated into Visual Basic GUI. Current plotting software needs the MS Excel to be installed on the computer.
3. The current design is only for binary systems. The future work can expend this software to multi-component multi phase calculations.

Literature Cited

1. Chandler, J. P. *MARQ 3.0*. [ftp a.cs.okstate.edu/pub/jpc/marq.f](ftp://a.cs.okstate.edu/pub/jpc/marq.f). Oklahoma State University, Stillwater, OK. May, 1981.
2. Chandler, J. P. *Personal communication*. Oklahoma State University, Stillwater, OK. 1999.
3. Codd, E. F. *Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks*. IBM Research Report, RJ599, San Jose, CA. 1969.
4. Cornell, G. *The Visual Basic 3.0 for Windows Handbook*. Berkeley, Osborne McGraw-Hall. 1993.
5. Date, C. J. *An Introduction to Database Systems*. 6th Ed., Addison-Wesley, 1995.
6. Fleming, C. C.; von Halle, B. *Handbook of Relational Database Design*. New York, Addison-Wesley. 1989.
7. Florentin, J. J. *Object-oriented Programming Systems*. Chapman & Hall, London, 1991.
8. Gasem, K. A. M. *GEOS Software 7.0*. Oklahoma State University. Stillwater, OK. 1998.
9. Hopper, K.; Newman, I. A. *Foundation for Human-Computer Communication*. New York, North-Holland Publisher. 1986.
10. Hu, D. *Object Oriented environment in C++*. MIS Press, Portland, OR, 1990.
11. Jackson, L. *A Comparison of Selected Gradient Methods for Solving the Nonlinear Least Squares Problem*. M. S. Thesis, Computer Science Department. Oklahoma State University, Stillwater, OK. 1974.

12. Korth, H.F.; Silberschatz, A. *Database System Concepts*. New York, McGraw-Hill, Inc., 1991.
13. Kowalik, J.; Osborne, M. R. *Methods of Unconstrained Optimization Techniques*. New York, American Elsevier Pub. Co., 1968.
14. Leavens, A. *Designing GUI Applications for Windows*. M&T Books, New York, 1994.
15. Lu, H. COMSC 5423 Lecture Notes. Oklahoma State University. Stillwater, OK. 1998.
16. Marcus, A. *The Cross-GUI Handbook: for Multi-Platform User Interface Design*. Addison-Wesley. MA 1995.
17. McFadden, F. R.; Hoffer, J. A. *Modern Database Management*. Benjamin/Cummings. New York. 1993.
18. Marquardt, D. W. An Algorithm for Least Squares Estimates of Nonlinear Parameters. J. SIAM. Vol. 11, 1953. 431-441
19. Marquardt, D. W. *Least Squares Estimation of Parameters*. Wilmington, Delaware: E. I. Dupont DeNemours and Co. Inc. SHARE (IBM user's organization) Distribution No. 3094, March, 1964.
20. Raghunathan, S. *EOS Model Evaluation for Asymmetric Mixtures and A Graphical Interface Implementation*, M. S. Thesis, Chemical Engineering Department. Oklahoma State University, Stillwater, OK. 1996.
21. Sanghavi, A. *Design and Development of a Thermodynamic Properties Database Using the Relational Data Model*, M. S. Thesis, Chemical Engineering Department. Oklahoma State University, Stillwater, OK, 1995.

22. Timothy, B. *Object Oriented Programming*. Addison-Wesley, MA. 1991.
23. Yourdon, E. *Object Oriented System Design*. Yourdon Press. Englewood Cliffs, NJ. 1994.
24. Zeck, S.; Wolf, D. *Requirements of Thermodynamic Data in the Chemical Industry*. Paper presented at the Sixth International Conference for Fluid Properties and Phase Equilibria for Processing and Design, Cortina, Italy, July, 1992.

APPENDIX A

5=RELATIVE ERROR IN (P, γ 1, γ 2)
 6=RELATIVE ERROR IN (P,Y1,Y2, γ 1, γ 2)
 7=RELATIVE ERROR IN (X)
 (IOPTN=6,8) 1=RELATIVE ERROR IN (T)
 2=RELATIVE ERROR IN (T,Y1)
 3=RELATIVE ERROR IN (T,Y1,Y2)
 4=RELATIVE ERROR IN (γ 1, γ 2) OR (K1,K2)
 5=RELATIVE ERROR IN (T, γ 1, γ 2)
 6=RELATIVE ERROR IN (T,Y1,Y2, γ 1, γ 2)

NTRAC PRINT OPTION FOR NUMERICAL ASPECTS OF REGRESSION

 -4=NO OUTPUT EXCEPT FATAL ERROR MESSAGE
 -3=NO OUTPUT EXCEPT ERROR MESSAGES
 -2= NO OUTPUT EXCEPT DIAGNOSTIC MESSAGES
 -1=STANDARD OUTPUT EXCEPT FINAL FIT VALUES
 0=STANDARD OUTPUT
 1=ALSO PRINTS RESULTS OF EACH ITERATION
 2=ALSO PRINTS THE COEFFICIENT MATRIX, QSAV
 3=ALSO PRINTS THE JACOBIAN MATRIX, P

MAXIT MAXIMUM NUMBER OF ITERATIONS FOR REGRESSION

MAXSUB MAXIMUM NUMBER OF SUBITERATIONS FOR REGRESSION

LINES 7-8

 IOPTN,METHOD,IEOS,MODEL,IFUGR,IVPM
 INPT,IUNIT,IRIT,IHS

IOPTN >>**PVTX-HS PROPERTIES**

 1=LIQUID DENSITY, ENTHALPY AND ENTROPY
 2=VAPOR DENSITY, ENTHALPY AND ENTROPY
 3=LIQUID/VAPOR DENSITY, ENTHALPY AND ENTROPY

 >>**EQUILIBRIUM PROPERTIES**

 4=ISOTHERMAL FLASH
 5=BUBBLE POINT PRESSURE
 6=BUBBLE POINT TEMPERATURE

	7=DEW POINT PRESSURE	
	8=DEW POINT TEMPERATURE	
	9=SOLUBILITY	
	10=COMPRESSIBILITY FACTOR (Z) MAPPING	
METHOD	1=SINGLE-MODEL EOS APPROACH (ϕ/ϕ)	
	2=TWO-MODEL SPLIT APPROACH (γ/ϕ)	
	3=NEW TWO-MODEL APPROACH (θ/θ)	
IEOS	EQUATION OF STATE	
	0=Ideal Gas	1=Virial-2
	2=van der Waals (VDW)	3=Redlich-Kwong (RK)
	4=Soave-Redlich-Kwong (SRK)	5=Peng-Robinson (PR)
	6=SPHCT	7=Modeified SPHCT
	8=Park-Robinson-Gasem (PRG)	9=BWR
MODEL	LIQUID SOLUTION MODEL	
	0=Ideal Solution	
	1=Willson	2=van Laar
	3=Margules-2	4=Redlich-Kister-3
	5=NRTL-DECHEMA	6=NRTL-ASPEN
	7=UNIFAC-93 (Predictive)	
IFUGR	REFERENCE LIQUID FUGACITY FOR (γ/ϕ) AND (θ/θ)	
	1= $P*\phi_i(P)$	2= $VP*\phi_i(VP)$
		3= $P*\phi_i(\text{mix}, P)$
	[For most applications use IRF=2; For Method 3 use IFUGR=3]	
IVPM	VAPOR PRESSURE MODEL	
	1=SVRC-4 (K,bar)	2=ANTOINE-3 (K, mmHg)
	3=ANTOINE -3(C,mmHg)	4=ANT.PHILLPS-4 (K,Pa)
	5=ANT.ASPEN-7 (K,Pa)	6=ANT.DIPPER-7 (K,Pa)
	7=GENERALIZED SVRC (predictive)	
	10=EOS MODEL WHEN PREDICTING HENRY'S LAW CONSTANT	
INPT	INPUT OPTION FOR	
	1=T,P,X1,ID,NRF,Y1,DL,DV	(Binary System)
	2=T,P,(ZE(I,J),J=1,NC)	(Multicomponent System)
	3=T,P,X1,X2,Y1,Y2,DL,DV	(Ternary System)
	4=NGPA,T,P,H,NRFC,PHS,(YE(I,J),J=1,NC)	(GPA Database)
	(Detailed definitions are given in Variable Index)	

IUNIT	<p>INPUT/OUTPUT UNITS</p> <p>1=T(K),P(bar),D(g/cc),H(kJ/mol),S(kJ/mol K)</p> <p>2=T(F),P(psia),D(g/cc),H(Btu/Lb),S(Btu/Lb F)</p> <p>3=T(C),P(mmHg),D(g/cc),H(kJ/mol),S(kJ/mol K)</p> <p>4=T(K),P(bar),TC(F),PC(psia)..</p>
IRIT	<p>INTERMEDIATE PRINT OPTION</p> <p>0=NO</p> <p>1=PRINT FROM UPDATE</p> <p>2=PRINT FROM VPRS</p> <p>3=PRINT FROM FUGR</p> <p>4=PRINT FROM FLASH</p> <p>5=PRINT FROM PTXY</p> <p>6=PRINT FORM SUMS</p> <p>7=PRINT FORM QUE (parameter initial estimates)</p> <p>8=PRINT FORM EQUIL (objective function value)</p>
IW	<p>REGRESSION WEIGTING OPTION (VALUE OF σ_y)</p> <p>0=NO WEIGHTING, $\sigma_y = 1$</p>
(IOPTN=1-4)	<p>1=RELATIVE ERROR IN (X,Y)</p>
(IOPTN=5,7)	<p>1=RELATIVE ERROR IN (P)</p> <p>2=RELATIVE ERROR IN (P,Y1)</p> <p>3=RELATIVE ERROR IN (P,Y1,Y2)</p> <p>4=RELATIVE ERROR IN (γ_1, γ_2)</p> <p>5=RELATIVE ERROR IN (P,γ_1, γ_2)</p> <p>6=RELATIVE ERROR IN (P,Y1,Y2,γ_1, γ_2)</p> <p>7=RELATIVE ERROR IN (X)</p>
(IOPTN=6,8)	<p>1=RELATIVE ERROR IN (T)</p> <p>2=RELATIVE ERROR IN (T,Y1)</p> <p>3=RELATIVE ERROR IN (T,Y1,Y2)</p> <p>4=RELATIVE ERROR IN (γ_1, γ_2)</p> <p>5=RELATIVE ERROR IN (T,γ_1, γ_2)</p> <p>6=RELATIVE ERROR IN (T,Y1,Y2,γ_1, γ_2)</p>
ITRVL	<p>OPTION USED TO AVOID TRIVIAL SOLUTIONS</p> <p>1=YES 0=NO</p>
IHS	<p>ENTHALPY AND ENTROPY DATA INPUT OPTION</p> <p>1=YES 0=NO</p>

LINE 9

CASES TO INCLUDE IN THE DATA ANALYSIS

THREE TYPE OF ANALYSES ARE POSSIBLE

SYSTEM-BY-SYSTEM	ALL THE DATA FOR A GIVEN BINARY ARE ANALYZED SIMULTANEOUSLY, E.G.,
REF-BY-REF	ALL THE DATA FROM A GIVEN SOURCE (REFERENCE) ARE ANALYZED SIMULTANEOUSLY(IOT=2,REF/REF)
ISO-BY-ISO	DATA FOR EACH ISOTHERM ARE ANALYZED SEPARATELY (IOT=1,T/T)

ICASE(L) CASE STUDIES PERFORMED
 0=NO 1=YES

<u>L</u>	<u>CASE</u>
1	COMMON CIJ FOR ALL SYSTEMS
2	COMMON CIJ AND DIJ ALL SYSTEMS
3	CIJ SYS/SYS ANALYSIS
4	CIJ REF/REF ANALYSIS
5	CIJ T/T ANALYSIS
6	CIJ/DIJ SYS/SYS ANALYSIS
7	CIJ/DIJ REF/REF ANALYSIS
8	CIJ/DIJ T/T ANALYSIS
9	NRTL-DECHEMA REF/REF (3 PARAM)
10	NRTL-ASPEN SYS/SYS (4 PARAM)
11	NRTL-ASPEN SYS/SYS (4 PARAM) AND CIJ (1 PARM)
12	TWO-PARAMETER ACT. COEFF. (γ) MODEL (REF/REF)
13	THREE-PARAMETER ACT. COEFF. (γ) MODEL (REF/REF)
14	THREE-PARAMETER ACT. COEFF. (γ) MODEL (REF/REF) AND CIJ (1 PARM)

LINE 10
SUMMARY OUTPUT PRINT OPTION

POSSIBLE SUMMARY OUTPUTS IN SEQUENCE INCLUDE

P: BUBBLE POINT P/T	FOR EOS PREDICTIONS & SPLIT PREDICTIONS
Y: VAPOR COMPOSITION	FOR EOS PREDICTIONS & SPLIT PREDICTIONS
L: LIQUID DENSITY	FOR EOS PREDICTIONS ONLY
V: VAPOR DENSITY	FOR EOS PREDICTIONS ONLY
HL: LIQUID ENTHALPY	FOR EOS PREDICTIONS ONLY
HV: VAPOR ENTHALPY	FOR EOS PREDICTIONS ONLY
K1: EQUIL. CONST. (1)	FOR EOS PREDICTIONS & SPLIT PREDICTIONS
K2: EQUIL. CONST. (2)	FOR EOS PREDICTIONS & SPLIT PREDICTIONS
A1: ACTIV. COEFF. (1)	FOR SPLIT PREDICTIONS ONLY

GEOS-7

A2: ACTIV. COEFF. (2)
SL: LIQUID ENTROPY
SV: VAPOR ENTROPY

FOR SPLIT PREDICTIONS ONLY
FOR EOS PREDICTIONS ONLY
FOR EOS PREDICTIONS ONLY

LINE 11

MASKING OPTION FOR OPTIMIZATION

THIS OPTION IS INVOKED WHEN PARAM. GENERALIZATION FLAG
IGN=1

MASK(J) 0=OPTIMIZE PARAMETER X(J)
 1=USING INPUT VALUES FOR PARAMETER

LINES 12 TO (12+NF-1)

NAMES OF FILES TO BE ANALYZED

INPUT DATA FILE

LINE 1
SYSTEM TITLE

LINES 2,3
OPTIONS TO SPECIFY:

MODE, IOPTN, METHOD, IEOS, MODEL, IFUGR, IVPM, IVPG, ITP
 NC, NPTS, INPT, IUNIT, IRIT, IW, ITRVL, IHS, NREF

MODE 0-1=USE LOG FILE INPUT OPTIONS
 2=USE DATA FILE INPUT OPTIONS

IOPTN >>**PVTX-HS PROPERTIES**

 1=LIQUID DENSITY, ENTHALPY AND ENTROPY
 2=VAPOR DENSITY, ENTHALPY AND ENTROPY
 3=LIQUID/VAPOR DENSITY, ENTHALPY AND ENTROPY

>>**EQUILIBRIUM PROPERTIES**

 4=ISOTHERMAL FLASH
 5=BUBBLE POINT PRESSURE
 6=BUBBLE POINT TEMPERATURE
 7=DEW POINT PRESSURE
 8=DEW POINT TEMPERATURE
 9=SOLUBILITY
 10=COMPRESSIBILITY FACTOR (Z) MAPPING

METHOD 1=SINGLE-MODEL EOS APPROACH (ϕ/ϕ)
 2=TWO-MODEL SPLIT APPROACH (γ/ϕ)
 3=NEW TWO-MODEL APPROACH (θ/θ)

IEOS EQUATION OF STATE

0=Ideal Gas	1=Virial-2	
2=van der Waals (VDW)	3=Redlich-Kwong (RK)	
4=Soave-Redlich-Kwong (SRK)	5=Peng-Robinson (PR)	
6=SPHCT	7=Modeified SPHCT	
8=Park-Robinson-Gasem	8=BWR	

MODEL	LIQUID SOLUTION MODEL 0=Ideal Solution 1=Willson 3=Margules-2 5=NRTL-DECHEMA 7=UNIFAC-93 (Predictive)	2=van Laar 4=Redlich-Kister-3 6=NRTL-ASPEN
IFUGR	REFERENCE LIQUID FUGACITY FOR (γ/ϕ) AND (θ/θ) 1= $P*\phi_i(P)$ [For most applications use IRF=2; For Method 3 use IRF=3]	2= $VP*\phi_i(VP)$ 3= $P*\phi_i(\text{mix}, P)$
IVPM	VAPOR PRESSURE MODEL 1=SVRC-4 (K,bar) 3=ANTOINE -3(C,mmHg) 5=ANT.ASPEN-7 (K,Pa)	2=ANTOINE-3 (K, mmHg) 4=ANT.PHILLPS-4 (K,Pa) 6=ANT.DIPPER-7 (K,Pa)
IVPG	GENERALIZED VPAOR PRESSURE PREDICTIONS 1=YES	0=NO
ITP	DATA ANALYSIS 0=ISOTHERMAL	1=ISOBARIC
NC	NUMBER OF COMPONENTS IN THE SYSTEM (MAX=20)	
NPTS	NUMBER OF DATA POINTS (MAX=100, or variable as set in COMMON.AAA)	
INPT	INPUT OPTION FOR 1=T,P,X1,ID,NRF,Y1,DL,DV 2=T,P,(ZE(I,J),J=1,NC) 3=T,P,X1,X2,Y1,Y2,DL,DV 4=NGPA,T,P,H,NRFC,PHS,(YE(I,J),J=1,NC)	(Binary System) (Multicomponent System) (Ternary System) (GPA DATABASE)
	(Detailed definitions are given in Variable Index)	
IUNIT	INPUT/OUTPUT UNITS 1=T(K),P(bar),D(g/cc),H(kJ/mol),S(kJ/mol K) 2=T(F),P(psia),D(g/cc),H(Btu/Lb),S(Btu/Lb F) 3=T(C),P(mmHg),D(g/cc),H(kJ/mol),S(kJ/mol K) 4=T(K),P(bar),TC(F),PC(psia)..	
IRIT	INTERMEDIATE PRINT OPTION	

	1=YES	0=NO
IW	REGRESSION WEIGTING OPTION (VALUE OF σ_y)	
	0=NO WEIGHTING, $\sigma_y = 1$	
(IOPTN=1-4)	1=RELATIVE ERROR IN (X,Y)	
(IOPTN=5,7)	1=RELATIVE ERROR IN (P)	
	2=RELATIVE ERROR IN (P,Y1)	
	3=RELATIVE ERROR IN (P,Y1,Y2)	
	4=RELATIVE ERROR IN (γ_1, γ_2)	
	5=RELATIVE ERROR IN (P, γ_1, γ_2)	
	6=RELATIVE ERROR IN (P,Y1,Y2, γ_1, γ_2)	
	7=RELATIVE ERROR IN (X)	
(IOPTN=6,8)	1=RELATIVE ERROR IN (T)	
	2=RELATIVE ERROR IN (T,Y1)	
	3=RELATIVE ERROR IN (T,Y1,Y2)	
	4=RELATIVE ERROR IN (γ_1, γ_2)	
	5=RELATIVE ERROR IN (T, γ_1, γ_2)	
	6=RELATIVE ERROR IN (T,Y1,Y2, γ_1, γ_2)	
ITRVL	OPTION USED TO AVOID TRIVIAL SOLUTIONS	
	1=YES	0=NO
IHS	ENTHALPY AND ENTROPY DATA INPUT OPTION	
	1=YES	0=NO
NREF	NUMBER OF TEXT LINES USED FOR REFERENCE CITATIONS	

FOLLOWING VLE DATA ENTRY, THE SEQUENCE FOR THE REFERENCES LIST MATCHES THAT GIVEN BY THE FLAG NRF IN THE VLE INPUT DATA

NEXT LINES (A LINE PER COMPONENT)

PURE FLUID PHYSICAL PROPERTIES:

NAME,TC(J),PC(J),W(J),TB(J),ZC(J),WT(J),IDN(J)

NAME	COMPONENT NAME
TC(J)	COMPONENT CRITICAL TEMPERATURE
PC(J)	COMPONENT CRITICAL PRESSURE
W(J)	COMPONENT ACENTRIC FACTOR
TB(J)	COMPONENT NORMAL BOILING POINT TEMPERATURE
ZC(J)	COMPONENT CRITICAL COMPRESSIBILITY FACTOR

WM(J) COMPONENT MOLECULAR WEIGHT
 IDN(J) COMPONENT ID NUMBER

IVPM VAPOR PRESURE MODEL
 1 {NAME, A,B,C FOR ANTOINE EQUATION}
 2 {NAME, ALFAC,DALFA,TS,PS FOR SVRC MODEL}
 3-6 {NAME, A-F, ID NO FOR PHILLIPS/ASPEN ANTOINE EQUATION}

NEXT LINES
ESTIMATES FOR INTERACTION PARAMETER C(I,J)
 (I=1,NC AND J=1,NC)

NEXT LINES
ESTIMATES FOR INTERACTION PARAMETER D(I,J)

NEXT LINES
ESTIMATES FOR ACTIVITY COEFFICIENT MODEL PARAMETERS, E1-EN

NEXT LINES
MASKING OPTION FOR OPTIMIZATION

MASK(J) 0=OPTIMIZE PARAMETERS X(J)
 1=USE INPUT VALUES FOR PARAMETERS

NEXT LINES
EXPERIMENTAL DATA (OR ESTIMATES FOR CALCULATED PROPERTIES)

INPT VLE DATA INPUT OPTION
 1 {T,P,X1,ID,NRF,Y1,DL,DV} (Binary System)
 2 {T,P,(ZE(I,J),J=1,NC)} (Multicomponent System)
 3 {T,P,X1,X2,Y1,Y2,DL,DV} (Ternary System)
 4 {NGPA,T,P,H,NRFC,PHS,(Y(I,J),J=1,NC)} (GPA DATABASE)

T(I) TEMPERATURE
 P(I) RESSURE
 X(I,1) LIQUID MOLE FRACTION OF FIRST COMPONENT
 ID COMPONENT ID NUMBER
 NRF SYSTEM REFERENCE NUMBER
 Y(I,1) VAPOR MOLE FRACTION OF FIRST COMPONENT
 DL LIQUID PHASE DENSITY
 DV VAPOR PHASE DENSITY
 Z(I,J) FEED MOLE FRACTIONS
 H ENTHALPY

NGPA GPA DATA POINT NUMBER
NRFC GPA REFERENCE NUMBER
PHS GPA PHASE CODE: L=LIQUID V=VAPOR

NEXT LINES

ENTHALPY AND ENTROPY EXPERIMENTAL DATA

IHS H&S DATA INPUT OPTION
1 {HL,SL,HV,SV,ID,NRF}
2 {HL,SL,HV,SV,ID,NRF} H&S are rebased using the first data
point.
The number of data points and sequence corresponds to that of
VLE data input controlled by INPT.

VARIABLE INDEX

(INCOMPLETE 1/97)

TC,PC,ZC,W	= PURE FLUID PROPERTIES (EOS INPUT VARIABLES)
T,P(PP)	= TEMPERATURE, PRESSURE
XE,YE,ZE	= COMPOSITION FOR LIQUID, VAPOR, AND FEED
LV	= 1 PREDICT LIQUID, LV =2 PREDICT VAPOR
K	= EQUILIBRIUM CONSTANT
D(L/V)	= DENSITY
F(L/V)	= FUGACITY
PH(L/V)	= FUGACITY COEFFICIENT
CIJ	= 1ST EOS INTERACTION PARAMETER
DIJ	= 2ND EOS INTERACTION PARAMETER
E1,...E4	= PARAMETERS USED FOR VOLUME TRANSLATION

APPENDIX B

One Case Study Using This Software

This appendix shows a series of interfaces when this software is used for one case study.

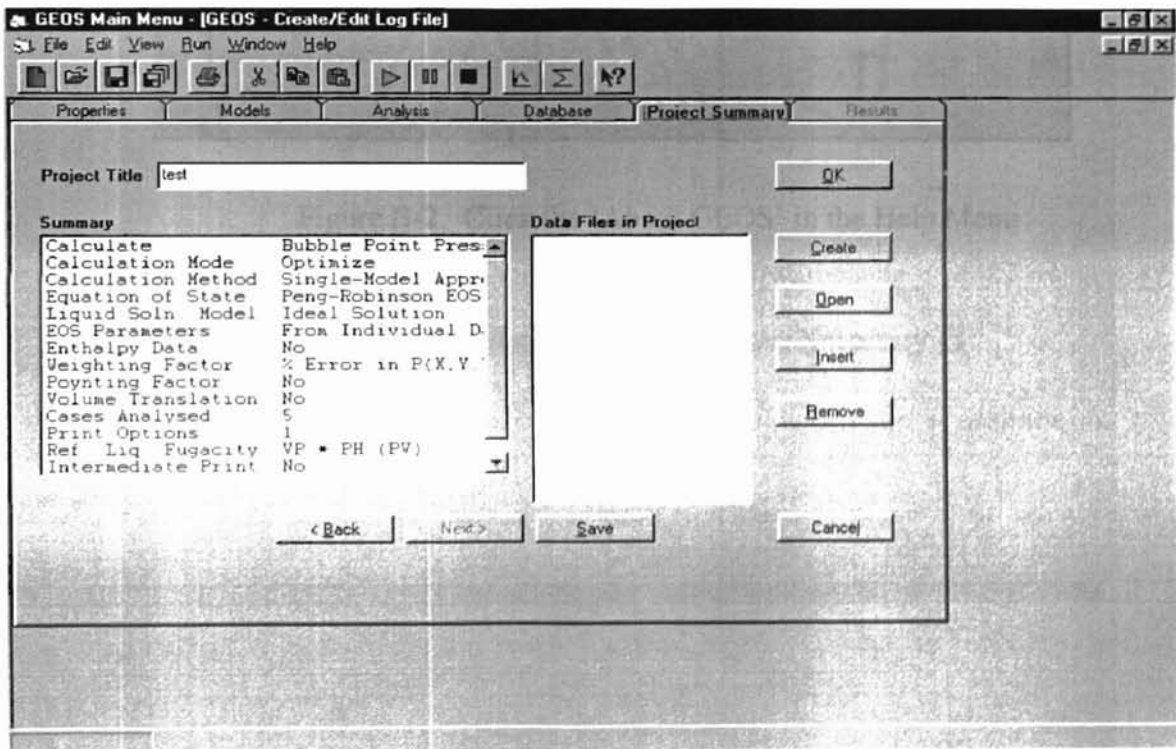


Figure B-1. First Screen When The Program Is Executed, Showing Default Options

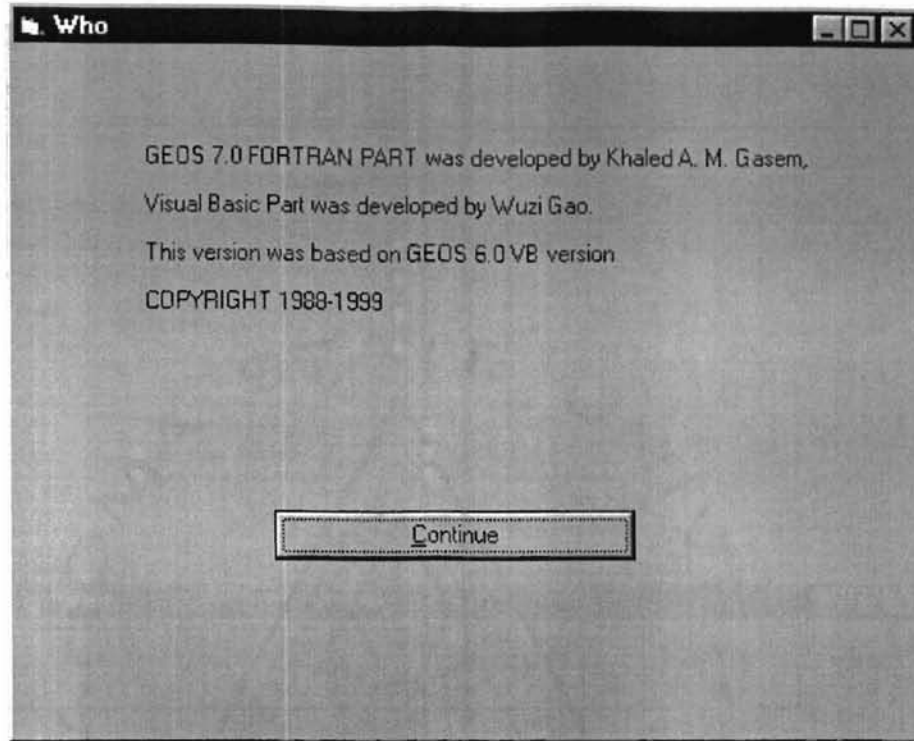


Figure B-2. Click the 'About GEOS' in the Help Menu

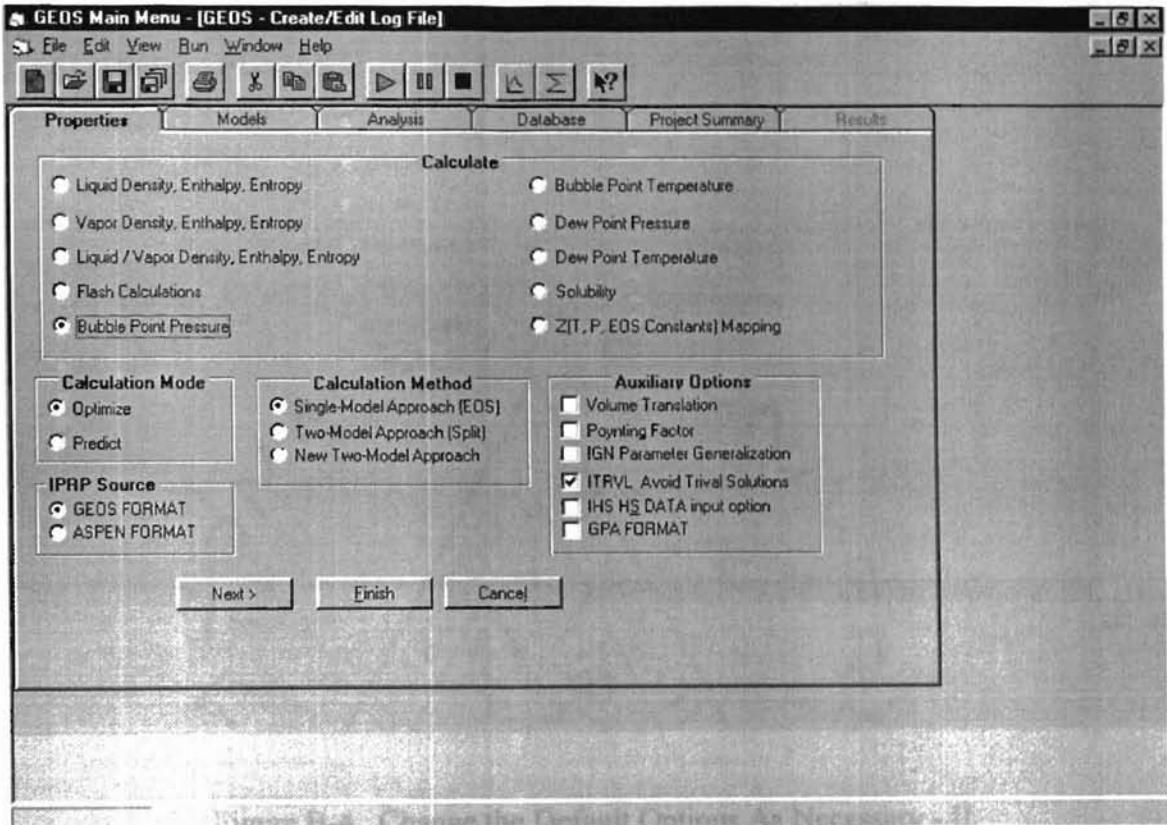


Figure B-3. Change the Default Options As Necessary - I

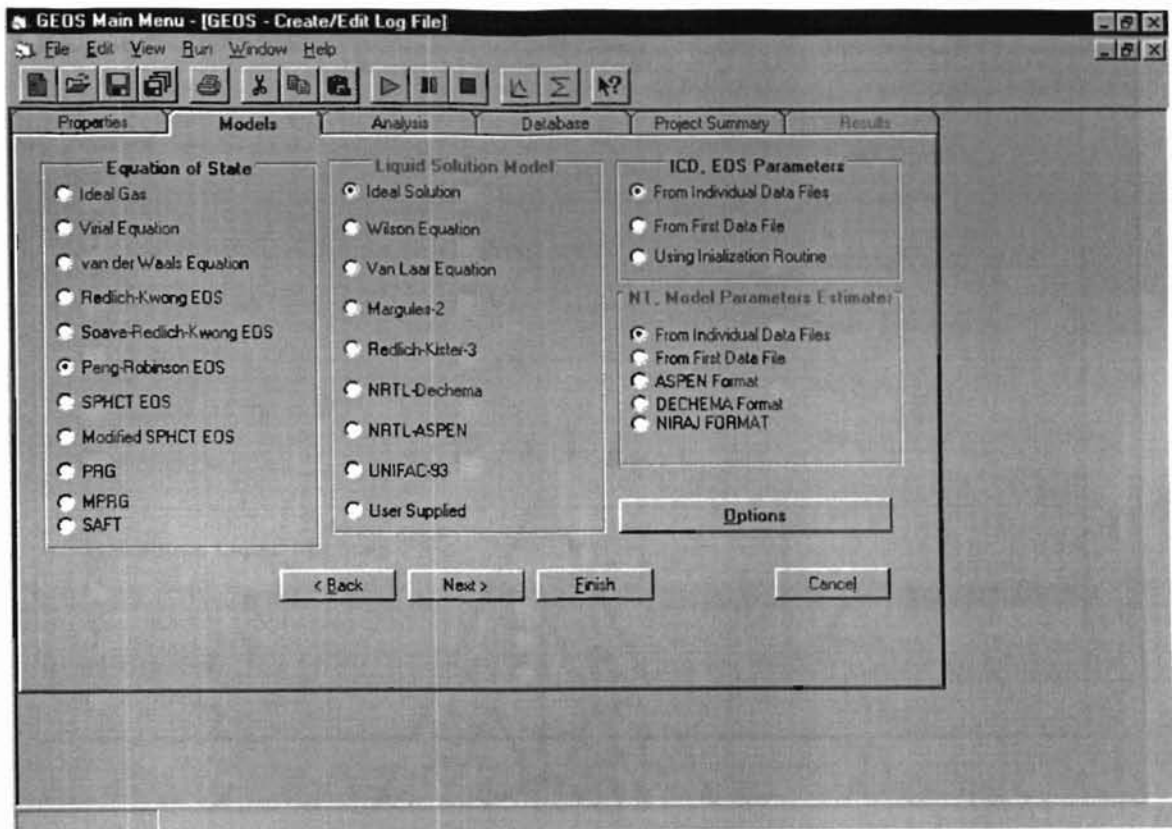


Figure B-4. Change the Default Options As Necessary - II

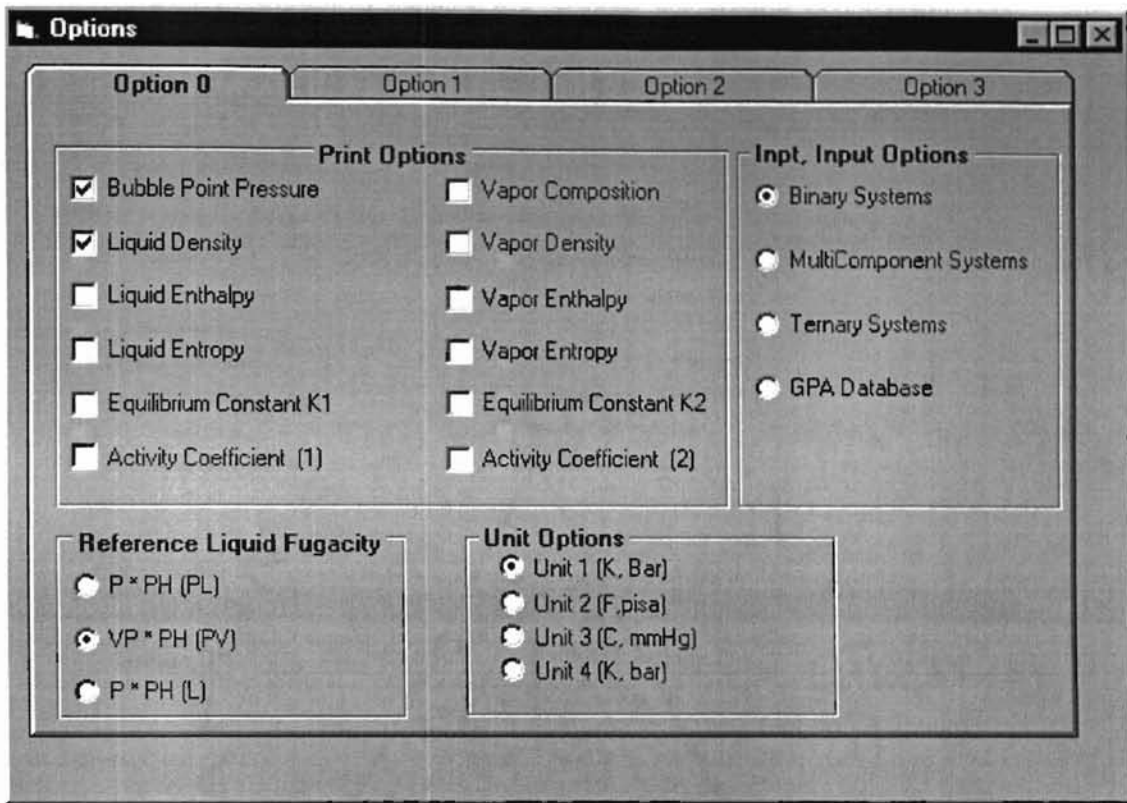


Figure B-5. Change the Default Options As Necessary - III

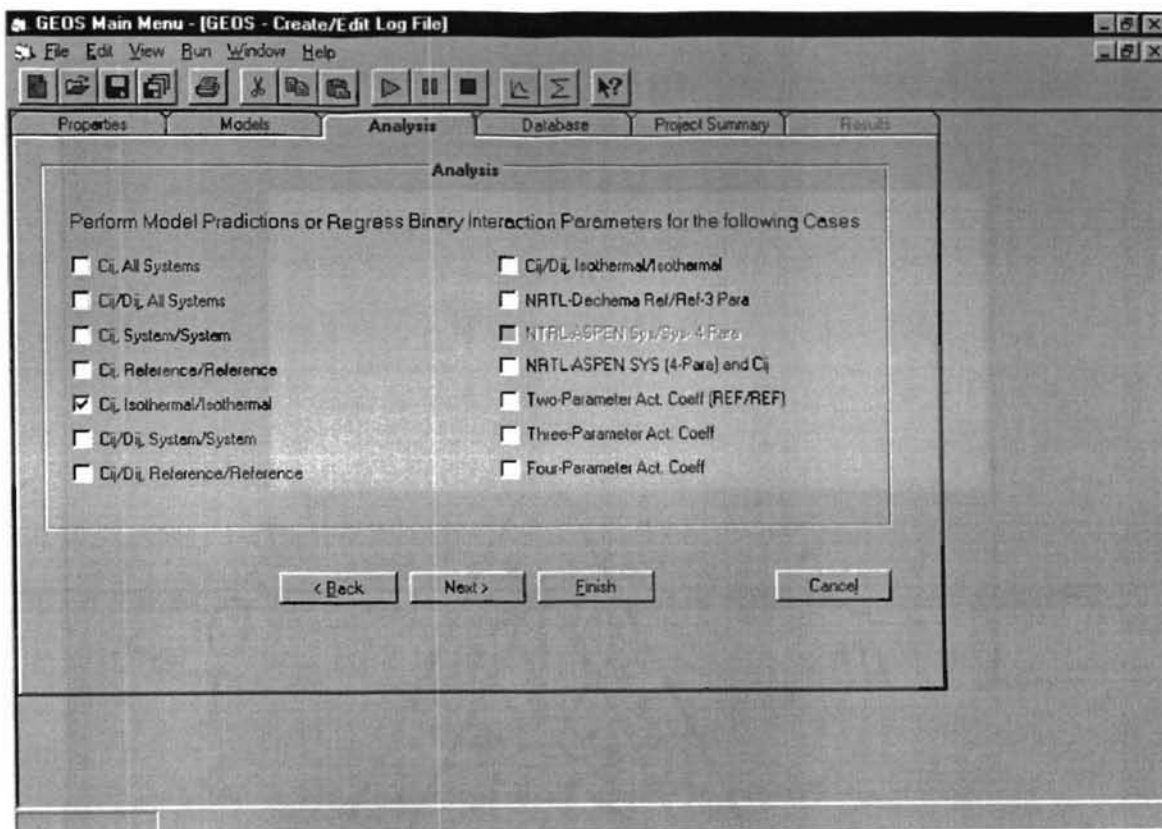


Figure B-6. Change the Default Options As Necessary - IV

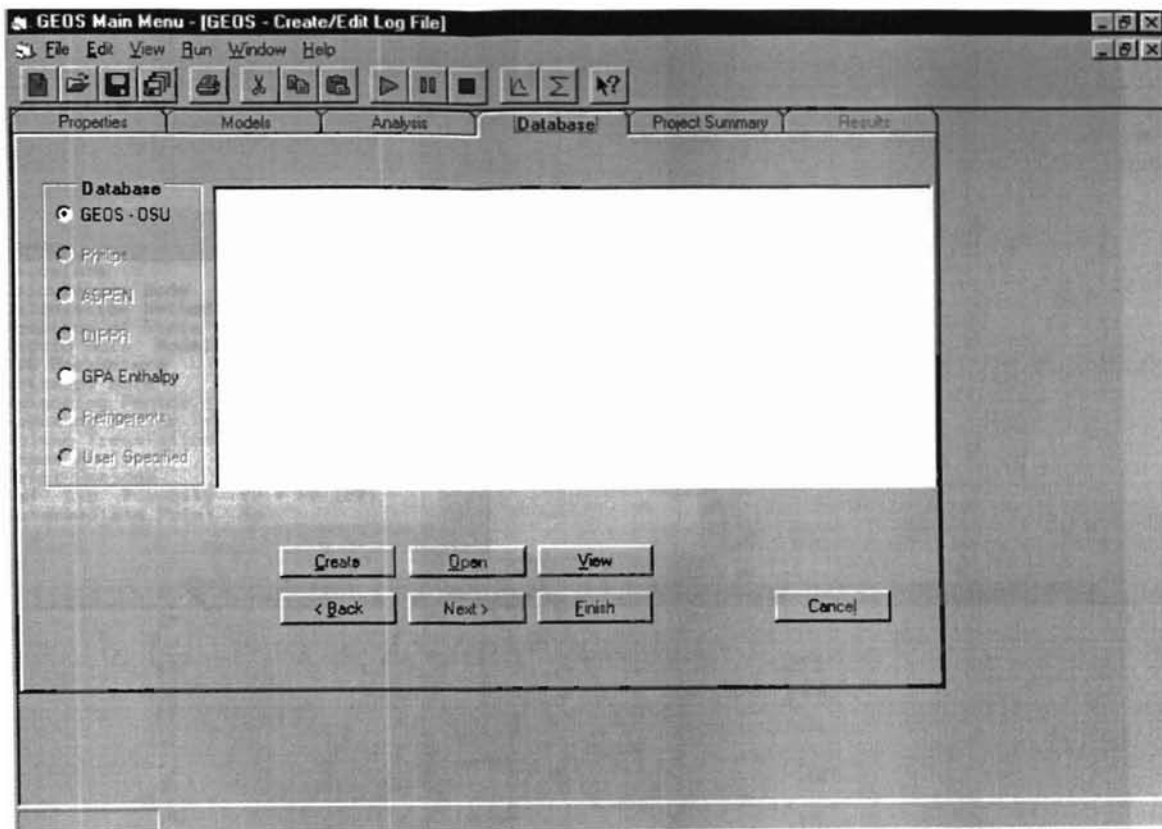


Figure B-7. Change the Default Options As Necessary - V

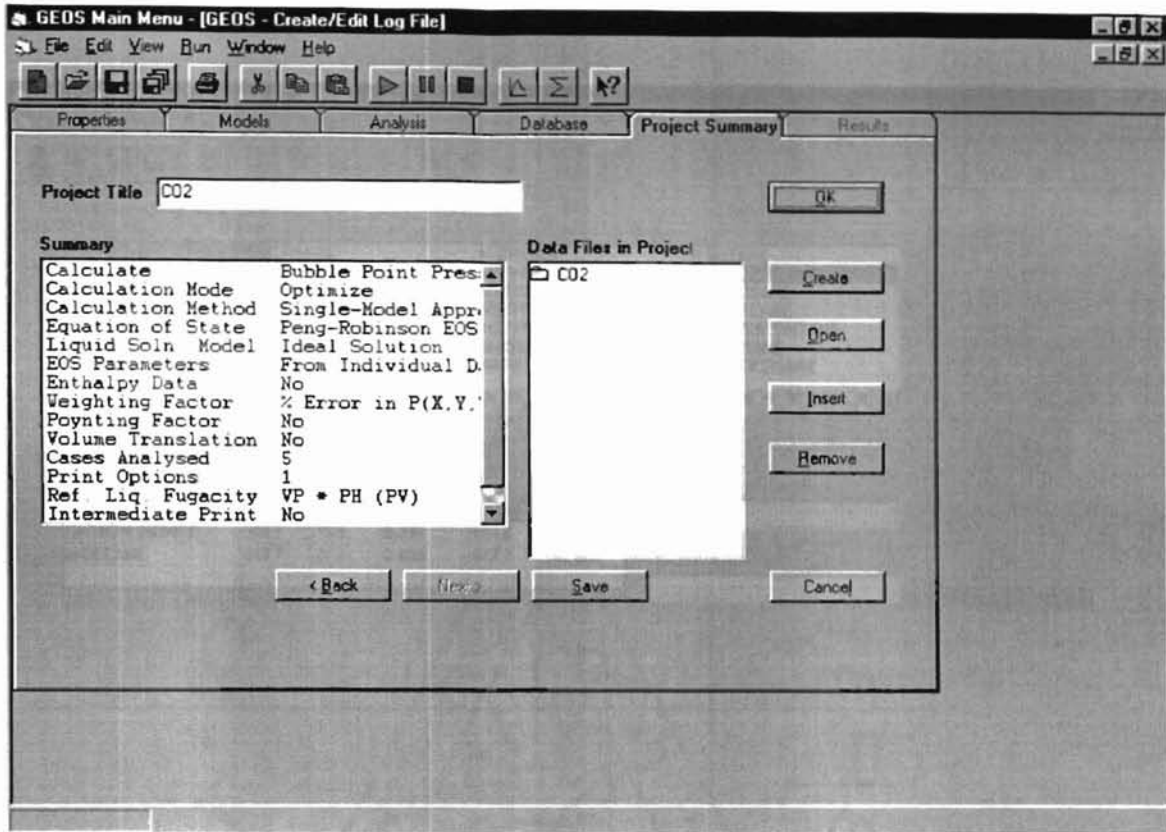


Figure B-8. Name the Project and View the Options

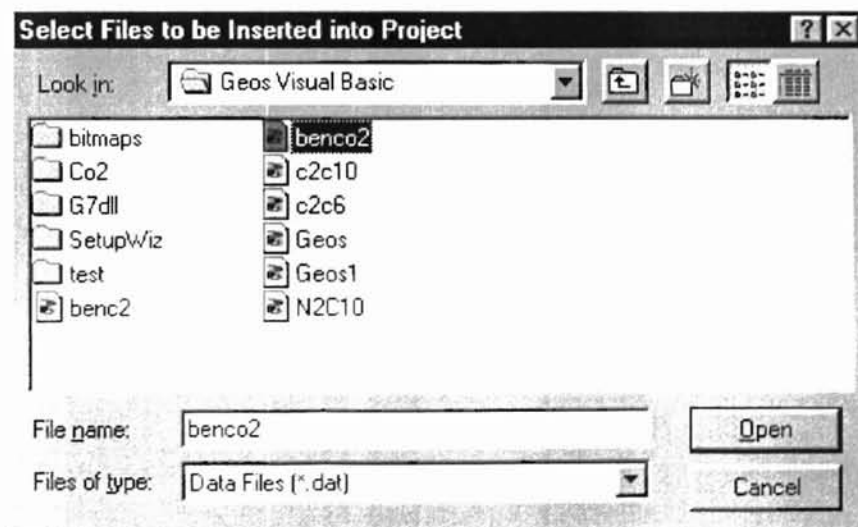


Figure B-9. Insert File Window

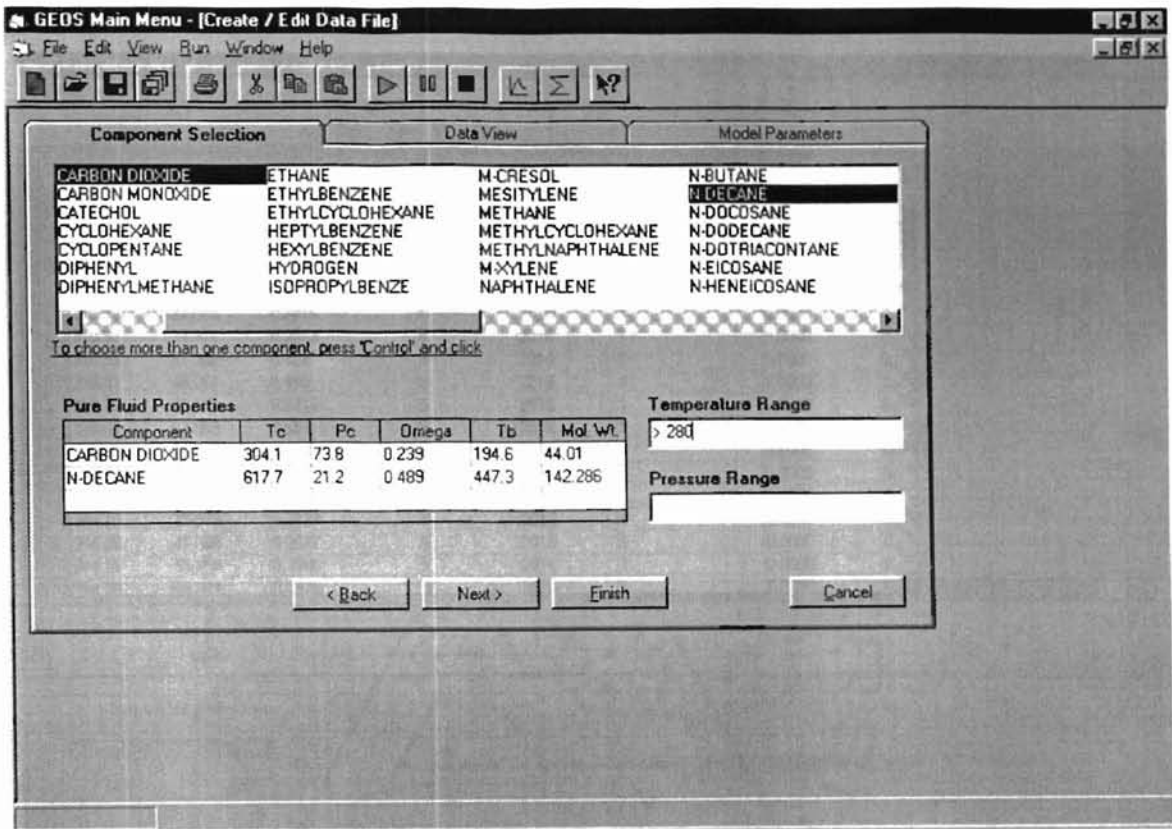


Figure B-10. Systems Selection Window After User Clicks on 'Create' Button

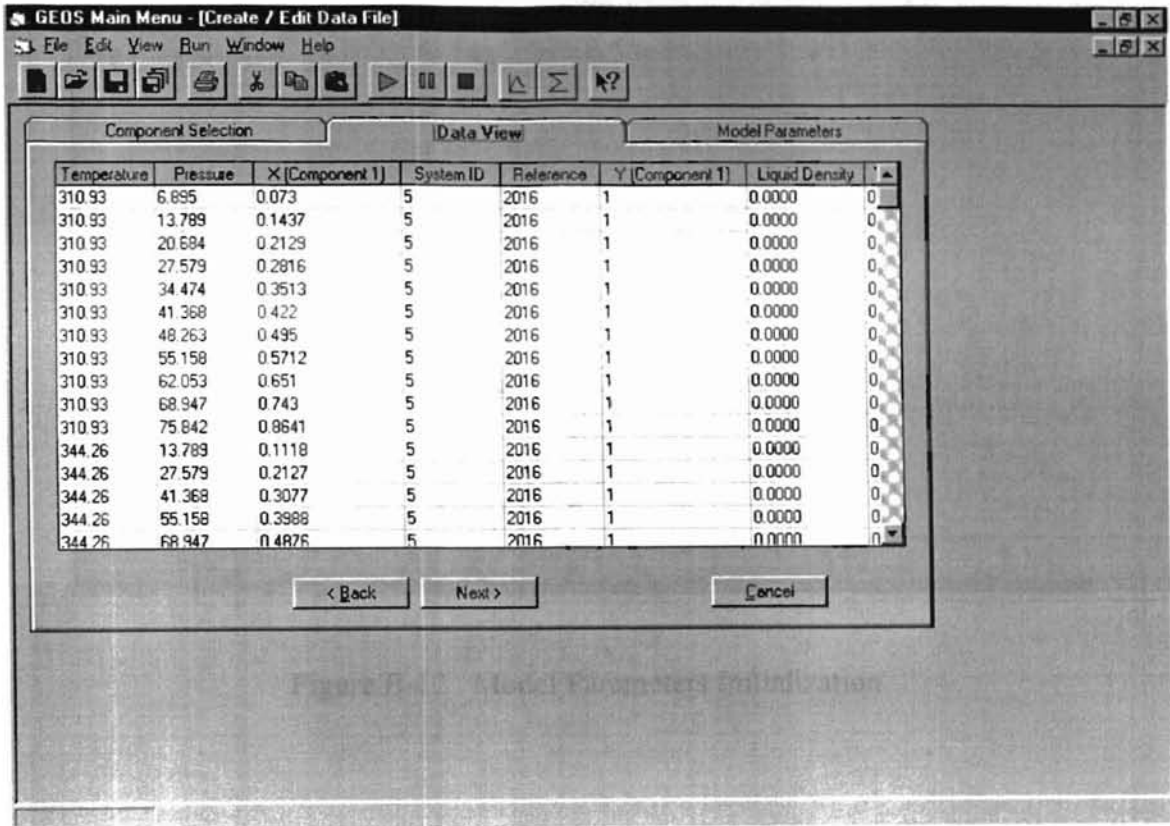


Figure B-11. Data View Window

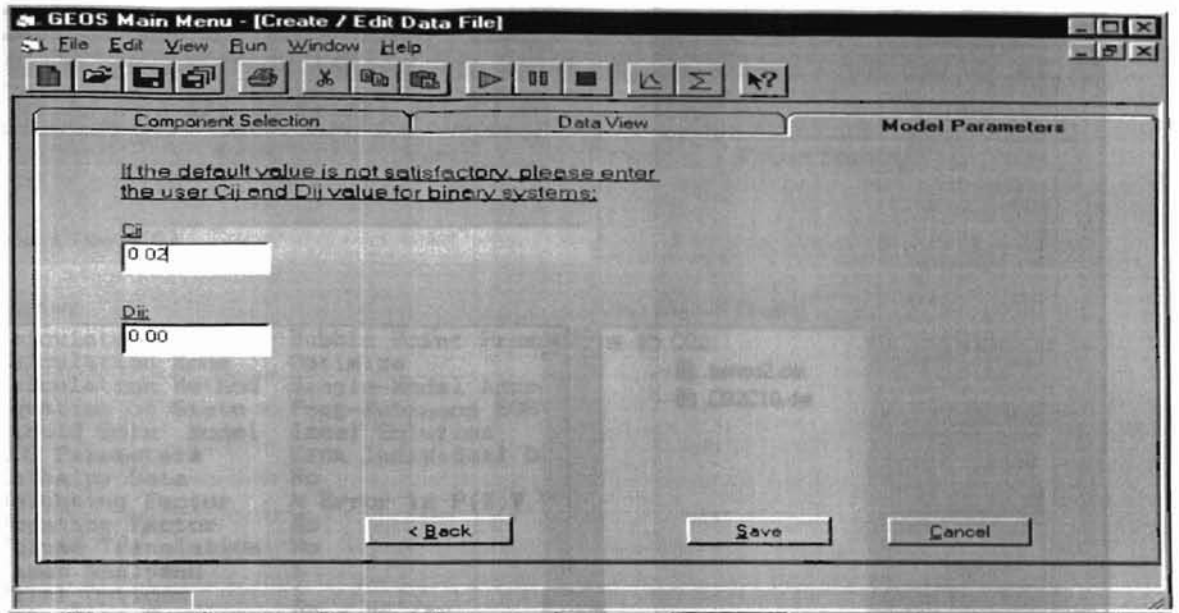


Figure B-12. Model Parameters Initialization

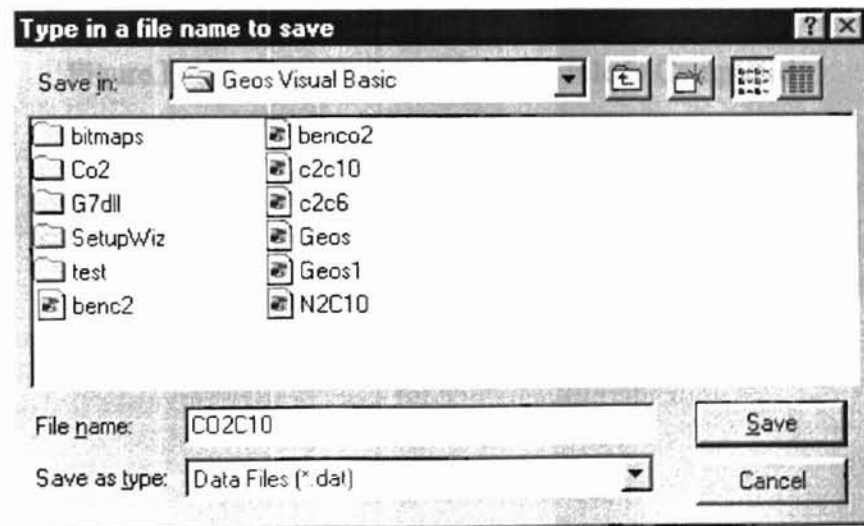


Figure B-13. Save Dialog Window

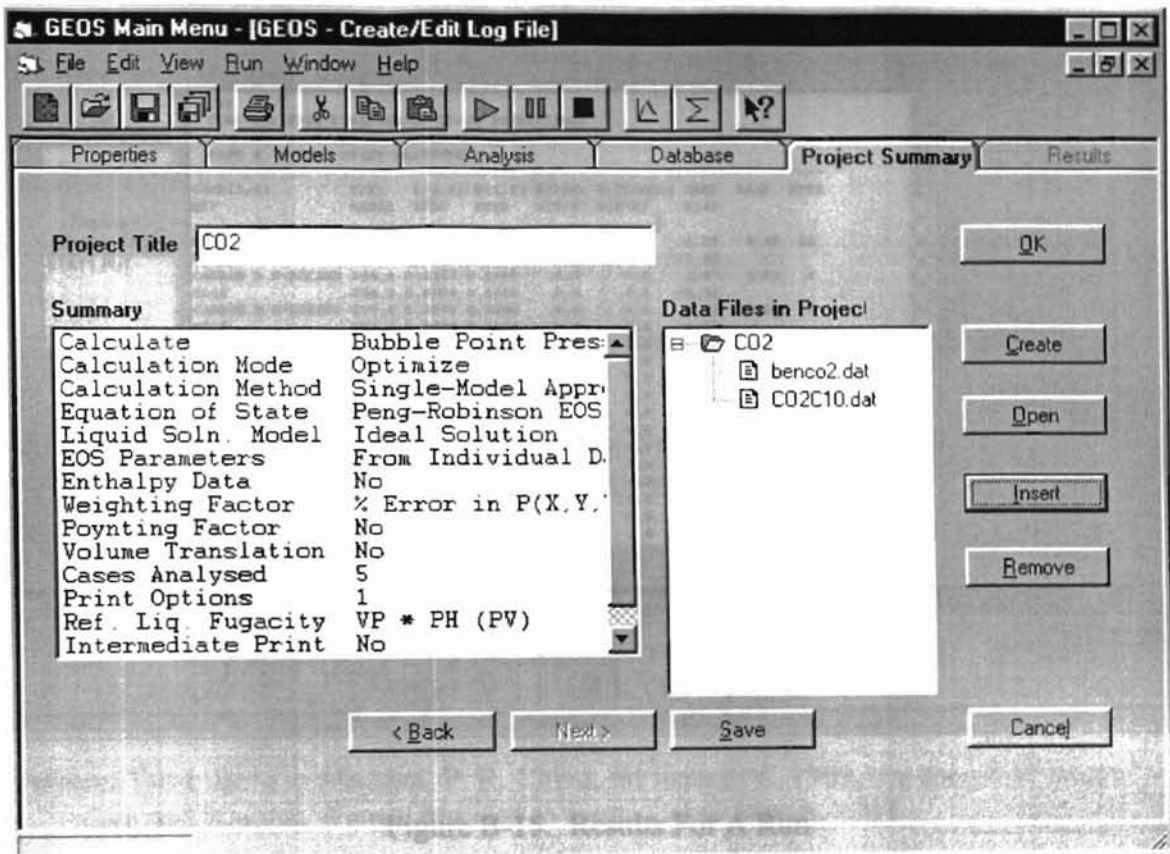


Figure B-14. The Window After Log File Is Completed

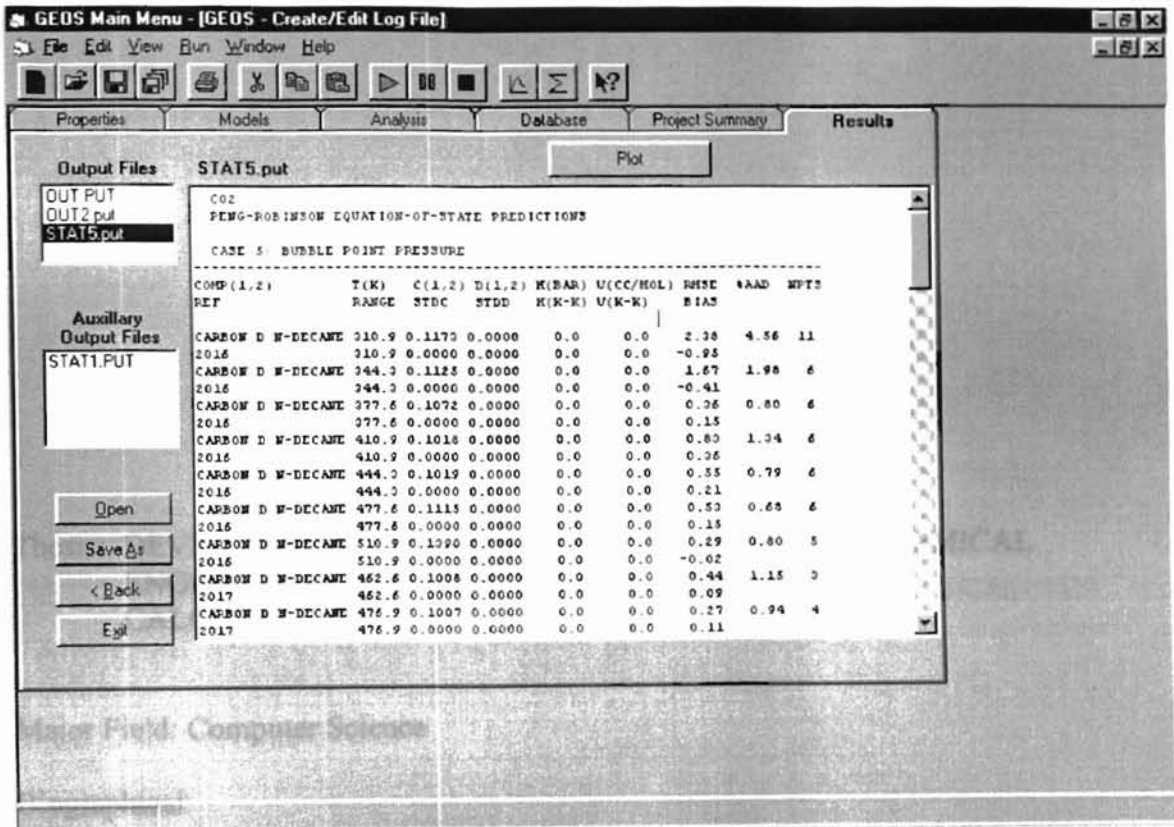


Figure B-15. Results For A Run

VITA

Wuzi Gao

Candidate for the Degree of

Master of Science

Thesis: DEVELOPMENT OF A SOFTWARE PACKAGE FOR CHEMICAL
ENGINEERING THERMODYNAMIC RESEARCH AND
CALCULATIONS

Major Field: Computer Science

Biographical:

Personal Data: Born in Shannxi, P. R. China, on January 6, 1969, the son of Shuenyu Gao and Xiaolan Wang. Married to Liwei Cai in March, 1995.

Education: Graduated from No. 3 High School, Zhouzhi, Shannxi, P. R. China in July, 1986; received Bachelor of Science in Mechanical Engineering from Xi'an Jiaotong University in 1990 and Master of Science in Mechanical Engineering from Beijing Institute of Chemical Technology, Beijing, P. R. China in July, 1993. Concurrently work for the PhD degree in Chemical Engineering. Completed the requirements for the Master of Science degree with a major in Computer Science and at Oklahoma State University in December, 1999.

Experience: Teaching and Research Assistant, Department of Chemical Engineering, Oklahoma State University, August, 1995, to December, 1999; Engineer, Beijing HuanQiu (HQ) Chemical Engineering Corporation, P.R. China, July, 1993, to November, 1995.