To the Graduate Council:

I am submitting herewith a thesis written by Daniel C. Evans entitled "Pyxis: A Java Program to Provide a Windows Environment for Performing Linked Engineering Calculations." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Nuclear Engineering.

Ronald E Pevey

Ronald Pevey, Major Professor

We have read this thesis
and recommend its acceptance

Lawrence W. Townsend

Lawrence J. Mills

Accepted for the Council:

Associate Vice Chancellor
and Dean of the Graduate School

Pyxis: A Java Program to Provide a Windows Environment for Performing Linked Engineering Calculations

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Daniel C. Evans

December 1999

Dedication

This thesis is dedicated to my parents Ron and Betty Evans

and to my wife Amy

who have given me tremendous support throughout my academic career.

Acknowledgements

This research was performed under appointment to the U.S. Department of Energy Nuclear Engineering and Health Physics Fellowship Program sponsored by DOE's Office of Nuclear Energy, Science and Technology. I am very grateful for the opportunity that this fellowship provided me.

I would like to offer special thanks to Mr. Steven M. Bowman at the Oak Ridge National Laboratory who provided oversight and assistance for summer work on this project. The faculty and students of the Nuclear Engineering Department have all offered assistance whenever it was needed. I am especially indebted to Dr. Ronald Pevey for his substantial contributions to this work as well Dr. H. L. Dodds for his support and encouragement.

Abstract

Pyxis is a software program designed to create an improved computational working environment for performing linked engineering calculations. Pyxis, programmed in Java, creates a platform-independent environment for managing the elements of engineering analyses, with characteristic icons representing items such as applications, data files, control structures and directories in a visual foldering system. The engineering analyst can visually arrange these elements in a familiar point-and-click windows environment to create a complete computational analysis. In addition to providing functional and visual linkage for elements of linked analyses, the product assists in peer review, tool sharing and archiving analyses. While the product may be useful in many fields, the product's design gives special consideration to the needs of nuclear engineers. The intent is to improve the quality of the analyst's working environment in order to improve the quality of the analyst's work.

Preface

The Pyxis computer code provides an improved computational environment for performing engineering calculations. Through a graphical interface it allows engineering analysts to automate many of the more tedious aspects of performing a multi-step analysis and provides a common look and feel useful for archiving and sharing analysis results. This thesis describes the Pyxis product. Chapter one discusses the reasons for choosing this project as well as the objectives of the program. The chapter goes on to provide definitions of terms appearing which may need clarification.

Chapter two is a review of similar applications and similar work in the field of nuclear engineering. The chapter discusses programs that have similar features to those of Pyxis and describes the differences between these programs and Pyxis. The chapter discusses the ongoing debate about the use of Java for engineering programming and highlights some work currently being done with Java in the field of nuclear engineering.

Chapter three discusses the features of Pyxis, dividing them into three categories. The features that allow the windows feel for the environment are discussed first. This is followed by a discussion of features providing graphical and functional linkage within the environment. Finally, additional significant features of the product are highlighted. Chapter four walks through a detailed example using Pyxis, highlighting many of the features discussed in chapter three.

Chapter five discusses the future outlook for Pyxis. It begins with a discussion of potential near term use of the product, followed by possible changes and potential long-term use of the product both in nuclear engineering and beyond. The Appendices

provide a feel for the source code and user help files and include notes for future programmers.

TABLE OF CONTENTS

Chapter One - Introduction *1*

Objectives of the Project 1

Rationale 1

Definitions 2

Chapter Two - Review of Related Work *5*

Current State of Nuclear Engineering Codes 5

Similar Applications Currently Available 6

The Use of Java for Engineering Applications 7

Other Nuclear Engineering Applications Being Developed in Java 10

Chapter Three - Features of the Product *12*

Windows Oriented Features 12

Graphical and Functional Linkage of Files 13

Other Important Features of the Product 14

Chapter Four - An Example of Pyxis Use *18*

Problem Introduction 18

Getting Pyxis Started 20

Setting up Folders 20

Incorporating Existing Tools into the Pyxis Environment 21

Creating New Files in the Pyxis Environment 22

Importing Items into the Session Environment 23

Setting up the Calculational Series 25

Running the Calculations 28

Comparison of Results 29

Archiving the Calculation 31

Chapter Five - Importance of the Product *32*

Applications for the Pyxis Product in its Current State 32

Modifications for Extended Product Use 33

Use of Pyxis Coding in Other Projects 34

Future Applications for Later Product Versions 34

Summary and Conclusions 35

REFERENCES *37*

APPENDICES *38*

APPENDIX A - Documentation of Public Classes in the Pyxis Code *39*

APPENDIX B - Online Help Documentation *60*

APPENDIX C - Notes to Future Pyxis Developers *88*

VITA *90*

LIST OF FIGURES

Figure 1. Operating Systems That Embed the Java Base Platform[4] 9

Figure 2. The Pyxis Foldering System 13

Figure 3. A Typical Pyxis Session Window 14

Figure 4. Preparing to Encapsulate Grouping 15

Figure 5. Session Window after Encapsulation 16

Figure 6. The Text Editor 17

Figure 7. Example Input File 19

Figure 8. The New - Folder Menu 20

Figure 9. Naming Newly Created Folders 21

Figure 10. Incorporating Existing Applications 22

Figure 11. Creation of Input Files 23

Figure 12. Importing an Application into a Session 24

Figure 13. The Session Menu 25

Figure 14. Setting the Application Properties 26

Figure 15. Add Link Menu Item 27

Figure 16. Creation of Links 28

Figure 17. The Finalized Analysis 29

Figure 18. Examining the Results 30

LIST OF ABBREVIATIONS

DOE – Department of Energy

DOORS – Discrete Ordinates of Oak Ridge System

DOS – Disk Operating System

GUI – Graphical User Interface

GRAVE – Graphical Rendering and Visualization Environment

HTML – Hyper Text Markup Language

JIT – Just In Time

KOFFI – Keyword Ordered Free Form Input

NFCA – Nuclear Fuel Cycle Analyzer

ORNL – Oak Ridge National Laboratory

RSICC – Radiation Safety Information Computational Center

SCALE – Standardized Computer Analyses for Licensing Evaluation

Chapter One

Introduction

Objectives of the Project

Pyxis is a software program designed to create an improved computational working environment for performing linked engineering calculations. The product's design gives special consideration to the needs of nuclear engineers. It is intended to assist in tool sharing, peer review, archiving and manipulation of linked calculations. While Pyxis could be useful in many fields, its design makes it an excellent tool for an analyst in the nuclear criticality safety field since peer review and archiving are significant parts of any criticality analysis. The purpose of Pyxis is to improve analyst efficiency and accuracy by creating an easy to use visual environment.

Rationale

In nuclear engineering analysis, many of the codes in common use were written over fifteen years ago at national labs before the advent of windows graphical environments. These codes often are used in linked analyses with one code's output becoming another code's input. When these codes were written, few of the modern graphical tools for user-friendly design were available. Most are programmed in Fortran, a computer language that is not well equipped for producing user-friendly front ends. Pyxis creates an environment for easier use of these types of codes.

A nuclear engineer may use one or more of these codes along with several simple tools to perform a single analysis. For example, a simple tool (like a word processor) might be used to generate an initial input file. This file might be the input for a major code that produces a lengthy text output file. The output file might be processed by another tool to generate a report or graph. This is a simple example of a linked calculation. Often a single analysis may involve many input and output files linked to multiple applications and tools; managing all of these inputs and outputs can create a logistical problem. It can also be difficult to communicate the flow of the calculation to others using only filenames and verbal descriptions. If an analyst leaves the calculation and returns to it after some time, the analyst may have difficulty remembering which files had been processed by which tools. For these reasons it is desirable to have a product which can visually link the multiple elements involved in modern nuclear engineering analyses.

The need for a product similar to this is becoming increasingly evident as more and more nuclear engineers are trained. Although these students are highly computer literate, they are forced to interact with the computer in an inefficient manner because many of the computer codes which they need to use are not available in the windows based, point-and-click type environment familiar to the student. Pyxis can help to bridge the gap between the modern user-friendly computing environments and the older codes prevalent in nuclear engineering.

Definitions

The following terms may be unfamiliar to the reader or have special meaning in this thesis:

Application – A computer program that transform one or more inputs into useful outputs. The inputs can be a mixture of input files or interactive input; likewise, the outputs can be files or interactive output. Applications are generally more complex than utilities (see below).

Class - "A type that defines the implementation of a particular kind of object.[1]"

Graphical user interface - "The techniques involved in using graphics, along with a keyboard and a mouse, to provide an easy-to-use interface to some program.[1]"

Linked analysis - A process in which the output from one application becomes the input for another application.

Object - "The principal building blocks of object-oriented programs. Each object is a programming unit consisting of data (instance variables) and functionality (instance methods).[1]"

Object oriented – "A software design method that models the characteristics of abstract or real objects using classes and objects.[1]" These pieces or objects interact with one another and with the user in a completed program. The objects may be expanded, and may be used by others in subsequent programs.

Platform independent - An adjective to describe a computer code that may be run on any type of computer (PC, MAC, Unix, etc.) without loss of functionality.

Portability - The ability to move a computer code from one type of computer to another without loss of functionality.

Pyxis – 1. The name for the product described by this thesis. "Pyxis" is not an acronym.
2. The constellation named after the"compass box" on old naval vessels.

User-friendly - An adjective to describe applications that incorporate modern graphical devices to create an easy working environment for users.

Utility - An application written by an analyst to process input or output files or to perform any simple task.

Chapter Two

Review of Related Work

Current State of Nuclear Engineering Codes

Code developers in the field of nuclear engineering are giving increasing attention to making newly created codes easier to use. Unfortunately, modern mechanisms for generating highly graphical codes were not available during the creation of many applications in common use today. The usual situation encountered by a new user of these old codes is to be forced to painstakingly create a lengthy text input file and deal with a lengthy text output file.

Some efforts have been made to retrofit older codes to make them more useable. Input and output processors have been created for both discrete ordinates and Monte Carlo type codes. There are a myriad of new visualization tools available to view and edit geometry data in both input and output files for several of the codes. Engineers have also written their own individual tools and processors to make the existing codes more usable to them. As new visualization tools and processors become available it becomes more important to have a working environment for incorporation of these new tools into a linked calculation. It should also be noted that many important codes might never become more user-friendly because of a lack of funding to update the codes in that way.

The problem of linking the elements needed to perform an engineering calculation in a convenient visual form has not yet been successfully addressed. The SCALE package links multiple applications such as the NITAWL and BONAMI

resonance processors (as well as many other tools) and other applications in the various SCALE sequences in an intelligent and useful manner. The combination of applications is done in a text-based way through the use of batch files. The applications are combined according to the way that analysts commonly use them. The linkages involved in the SCALE package are often completely transparent to the user who may view the entire sequence as a single application.

There are several limitations to the type of linkage currently available in the SCALE package. The text-based linkages in the SCALE package sequences are firm and are difficult to modify by analysts. The sequences may not be easily opened into their original parts. The various parts of the sequences tend to become invisible to the analyst, who needs to understand the stages of the process involved in a SCALE sequence in order to use it effectively. The SCALE sequences are not easily expandable into new sequences by typical analysts. If the analyst wishes to incorporate additional tools into the calculation, he must abandon SCALE and individually run the constituent SCALE applications and manage the intermediate data files himself.

Similar Applications Currently Available

No currently available application accomplishes the goal set out for this program: to create a visual environment for linked calculations utilizing a visual foldering organization of constituent elements, although there are tools that provide portions of this functionality.

Obviously, there are tools that provide a windows-type operating system as a working environment. Tools such as Microsoft's Windows '98 are able to incorporate stand-alone applications into a visual foldering system. Files are represented by

characteristic icons and can be easily moved, opened and closed, etc. These tools are generally not platform independent, however, and do not allow for linkage (graphically or functionally) of applications and other files.

There are also tools that allow graphical linkage of items within the working environment. These tools may present the linkage with some sort of flow diagram. This is not a functional linkage, but simply a visual linkage for the purpose of diagramming some process.

There are also ways to link applications and other files non-graphically. The writing of scripts or batch files can be used to set up linked analysis with great flexibility. The SCALE package provides functional linkage of applications and other files through use of the SCALE driver. These methods for providing functional linkage of files work very well, but they do not provide any visual representation of the linkage and can be difficult to modify quickly.

The Pyxis product is not unique in its use of a windows-type foldering system, or in the functional linkage of applications and other files, or even in the visual linkage of objects. Pyxis is unique, however, because it combines all three of these features into one platform independent tool.

The Use of Java for Engineering Applications

There is an ongoing debate in the computer science and engineering fields about the use of the newly developed Java language for engineering or computationally intense applications. Since Pyxis is programmed in Java, a discussion of the Java language, its benefits, potential drawbacks and suitability for this project is relevant.

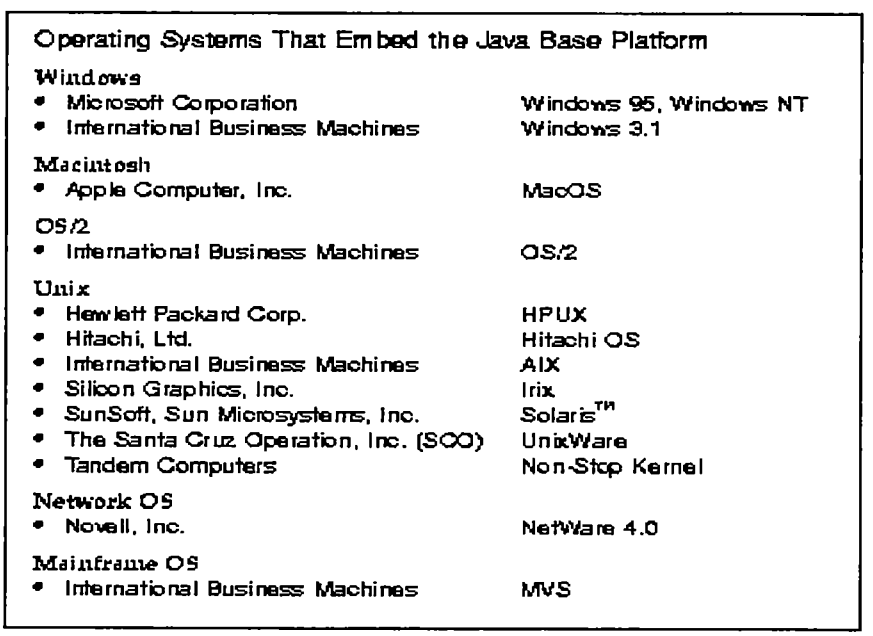
Java is described by its creators as "A simple, object-oriented, network-savvy, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, dynamic language.[2]" The Java language was built to closely resemble C++ with many rarely used, poorly understood, or confusing features of C++ omitted (e.g., pointers, multiple inheritance, templated functions). Automatic garbage collection was added to Java to automatically free up memory allocated to variables no longer used. Java is an interpreted language, which means its bytecodes are translated on the fly to native machine instructions and generally not stored anywhere. This makes the development process more rapid, but can result in a loss of speed for the final code because it is interpreted each time it is run. Java is an object-oriented code with a robust collection of objects available for use by programmers. This collection of objects is constantly expanding and is especially strong in providing objects for graphical user interfaces.[3]

The biggest advantage of Java technology is its portability. Portability makes it easier for companies to change/upgrade operating systems and platforms without losing their investment in applications software. Portability makes a wider variety of software available on all computers. Even for users of a single platform (e.g., IBM-compatible personal computers) software portability is an important consideration for software developers and users because of evolving operating systems. Software written in Java runs without change as computers move from MS-DOS to Windows 3.1 to Windows 95 to Windows 98 to Windows NT, and even on to Windows CE. This characteristic of Java saves users from having to buy new applications software over and over again when Windows changes. In addition, applications programmed in Java are highly portable among the various computer platforms in general use. Figure 1 shows a list of platforms

that incorporate the Java base platform. This list is constantly growing as Java strives to become completely platform independent.

There are potential drawbacks to the use of Java for engineering applications. One drawback is that the language is not as established as C++ or Fortran. Because the language is new and rapidly evolving, programmers may find they have to constantly update their product to keep up with changes in the programming language itself. This was the case with the programming of Pyxis. While newer versions of the Java language are generally compatible with older versions, there were situations with Pyxis where the source code needed modification to run properly as the Java language evolved.

The primary drawback of using Java for engineering applications is loss of speed. Computationally intense codes compiled in C++ or Fortran tend to run much



Operating Systems That Embed the Java Base Platform

Windows	
• Microsoft Corporation	Windows 95, Windows NT
• International Business Machines	Windows 3.1
Macintosh	
• Apple Computer, Inc.	MacOS
OS/2	
• International Business Machines	OS/2
Unix	
• Hewlett Packard Corp.	HPUX
• Hitachi, Ltd.	Hitachi OS
• International Business Machines	AIX
• Silicon Graphics, Inc.	Irix
• SunSoft, Sun Microsystems, Inc.	Solaris™
• The Santa Cruz Operation, Inc. (SCO)	UnixWare
• Tandem Computers	Non-Stop Kernel
Network OS	
• Novell, Inc.	NetWare 4.0
Mainframe OS	
• International Business Machines	MVS

Figure 1. Operating Systems That Embed the Java Base Platform[4]

faster that Java codes. This is because Fortran and C++ codes are compiled into machine bytecode for a given computer platform. The usual situation for Java source code is to be interpreted into bytecodes on the fly as a program is run, which is a slower process. However, for some platforms there are Just In Time (JIT) compilers available to generate machine code as a Java program is run. With JIT compilers, a given portion of coding is interpreted the first time it is encountered, but the bytecodes are reused if the same section of coding is executed again. Since so many loops are generally present in engineering and scientific codes, this results in substantial savings. According to Sun Microsystems, "Perfomance of bytecodes converted to machine code is roughly the same as native C or C++.[4]" This means that through the use of JIT compilers, which are becoming available for an increasing number of platforms, the speed disadvantage of using Java is greatly mitigated.

This project, while not utilizing computationally intense Java programs, illustrates that Java can be useful to the engineering analyst by performing non-numerically intensive tasks. Because Pyxis is programmed in Java, it is platform independent. This means that a user working on a PC or working on a Sun Workstation will have the same functionality. Pyxis appears slightly different on different platforms, but performs the same basic functions equally well on all platforms.

Other Nuclear Engineering Applications Being Developed in Java

There are a couple of other nuclear engineering applications that are being developed using the Java language of which I have become aware. Mr. Ronald W. Lee, a computer scientist working for the Radiation Safety Information Computational Center (RSICC) at Oak Ridge National Laboratory (ORNL) has developed a Java code entitled

NFCA (Nuclear Fuel Cycle Analyzer) which visually links parts of a complete fuel cycle such as mines, enrichment and purification facilities and reactors. NFCA allows users to see the complete fuel cycle as well as learn more about individual parts of the cycle if they so desire. Mr. Lee also develops applications for the Department of Defense in Java. In his extensive experience with Java, Mr. Lee has not found computational speed to be a significant problem. He recommends the use of Java to provide the front end for some Fortran codes.[5]

Mr. Edward D. Blakeman, another researcher at ORNL, is in the process of completing the programming for a TORT geometry viewer in Java. He chose Java because of the platform independence, recognizing that this choice would also cost a reduction in speed. The program is currently entitled GRAVE, which is an acronym for Geometry Rendering and Visualization Environment. The code also includes some C++ that is wrapped in Java. This program will be expanded into a DOORS graphical user interface also being developed by Mr. Blakeman in Java. The work of Mr. Lee and Mr. Blakeman demonstrate the potential for an increasing use of Java in the field of nuclear engineering.[6]

Chapter Three

Features of the Product

Windows Oriented Features

Pyxis is a platform-independent windows environment for managing the elements of engineering analyses. The features associated with it are similar to, although not as extensive as, those seen in Microsoft's Windows or the Apple operating environment. Pyxis resembles a typical windows foldering system in both appearance and use. Some of its features are:

1. Items such as applications, data files, control structures, and directories are represented visually by characteristically shaped icons. These icons are determined by the file extension and may be modified by the user.
2. Folder icons represent directories on the computer. Folders may be opened into a folder window by double clicking on a closed folder icon.
3. Files and folders may be moved within a folder window or to a new folder through typical drag and drop using a mouse.
4. The user may rename or delete files, create and name folders, and run simple applications or batch files through double click execution.
5. An extensive user prompt system exists to notify the user of problems, and to instruct the user about possible solutions.
6. There are pull-down menu items that invoke common windows functions; these functions may be accessed by either mouse click or by shortcut keys.

Figure 2 shows two open folder windows along with several closed folders in the Pyxis environment. Figure 2 also shows many characteristic icons in the folder windows and the menubars at the top of folder windows. (The Albert Einstein image is the default icon for file types not associated with another icon image.)

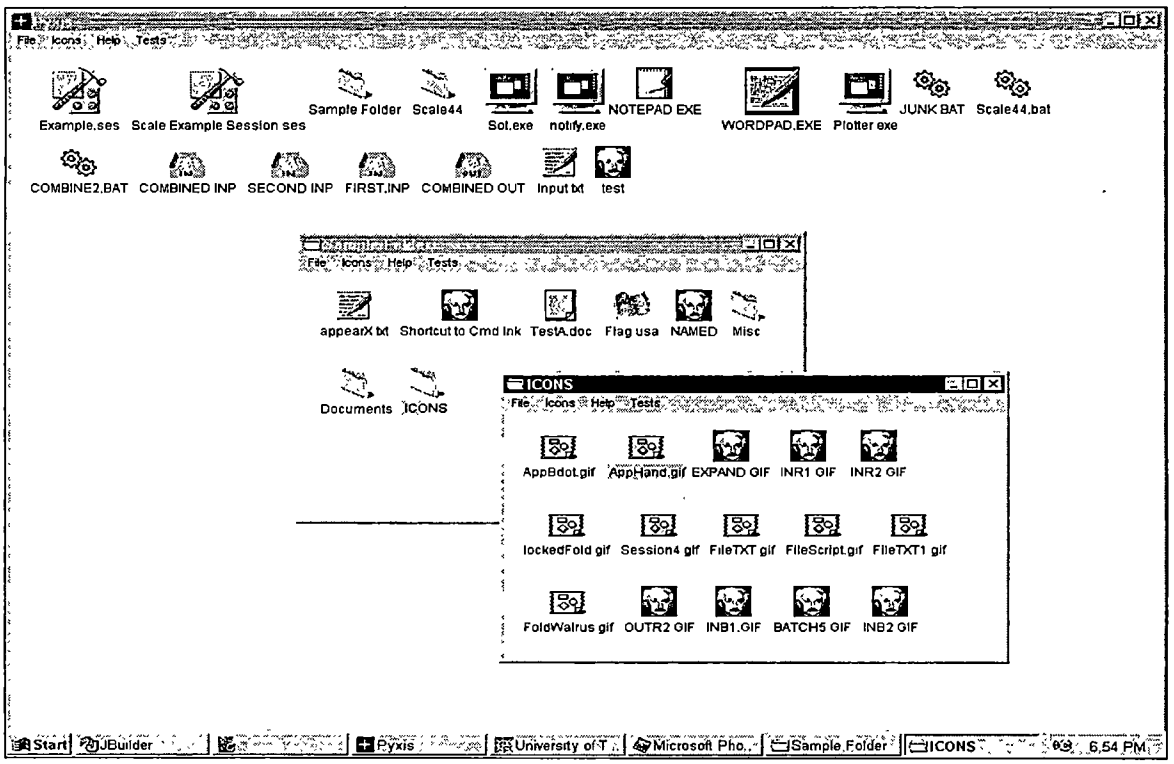


Figure 2. The Pyxis Foldering System

Graphical and Functional Linkage of Files

Engineering computations are performed in what is termed a "Session" environment. Sessions are special windows that do not represent underlying directories, but instead serve to display relationships among items which may be located anywhere on the computer. Pyxis allows visual linking of items within the session. Files read as input to an application are shown connected to the application by an arrow showing

direction of flow of the data; output files from an application are similarly depicted as linked by an outflow arrow. These output files may be linked to become inputs to other applications or tools. After the user completes the series of linked items, the user can run the calculation. After examining the analysis results, the user may choose to add control structures, add additional applications, modify the input and rerun the calculation. This allows the user to make small changes in one section of a series of calculations without having to set up the series again. It also allows easy visualization of the entire calculational sequence.

Figure 3 shows an example of the session environment. This single pass session consists of a single engineering application (SCALE44.BAT) which uses a single input file (COMBINED.INP) and creates a single output file (COMBINED.OUT). For this example, the input file is constructed by using a utility (COMBINE2.BAT) to concatenate two original input files (FIRST.INP and SECOND.INP). Three I/O applications (NOTEPAD.EXE, WORDPAD.EXE and PLOTTER.EXE) are used to examine the initial, intermediate and final data files. A final utility (NOTIFY.EXE) sends the analyst an e-mail message when the final output file (COMBINED.OUT) is generated.

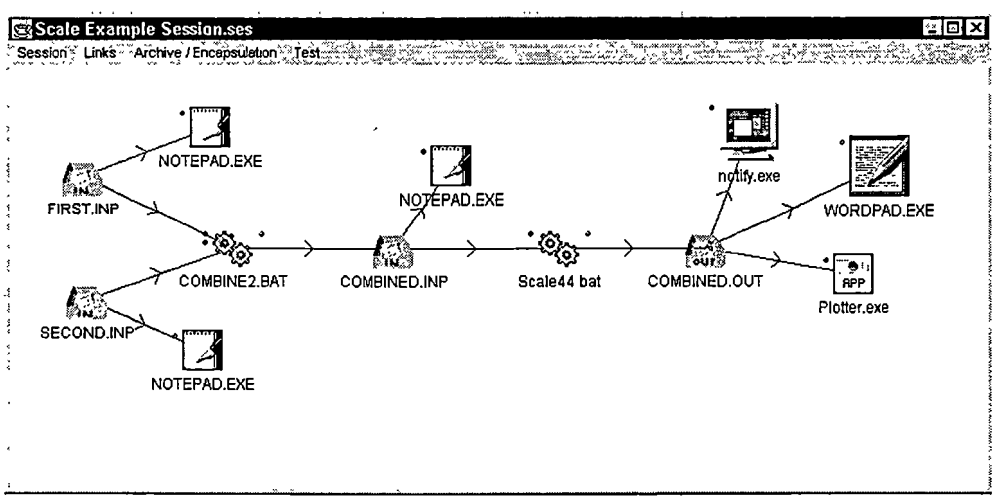


Figure 3. A Typical Pyxis Session Window

Other Important Features of the Product

Encapsulation is a feature of Pyxis that allows the user to group a series of linked items and create a new "super application." Encapsulation is available to the user when working in the Session Environment. The super application behaves as any other individual item would. The super application can then be used as the user sees fit, including saving the super application for use in other projects, and can be ungrouped at any time. These features are intended to simplify the task in order to save the analyst time. Figure 4 shows the session window as an analyst prepares to group several linked items. The user first creates a group box using the menu items and the mouse. Then the user selects "Encapsulate Grouping" from the "Archiving / Encapsulation" menu. The user then is prompted for needed information about linking to and from the encapsulation and to name the new super application.

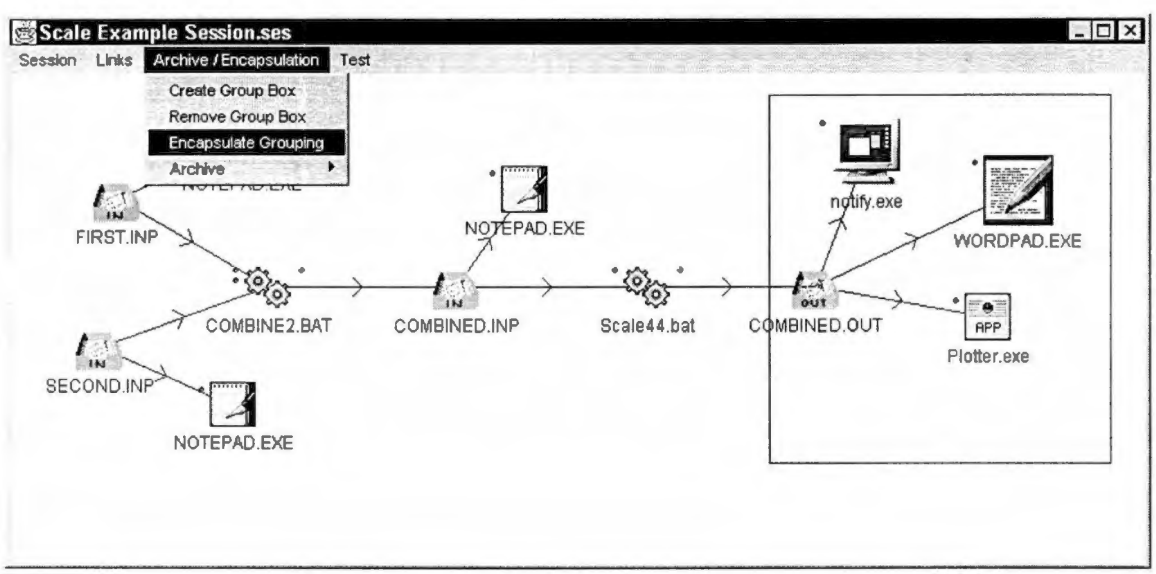


Figure 4. Preparing to Encapsulate Grouping

In this example, the new super application encapsulates the COMBINE.OUT data file with the Plotter, Wordpad and Notify utilities. This super application serves to provide a location for the SCALE output, which is displayed upon creation by the Plotter and Wordpad utilities. The analyst is also notified of the creation of a new output file by the Notify utility. The analysis performed by the session after creation of the super application is identical to the one performed before creation of the super application. The super application serves to group items often used together for use in future analyses and to simplify complex analyses by visually grouping closely related items. Figure 5 shows the session window as it would appear after the encapsulation. The analyst can select a new icon for the super application by selecting "Change Icon" from the "Session" menu. The user can also reopen the super application by selecting "Open Super Application" from the "Archive / Encapsulation" menu.

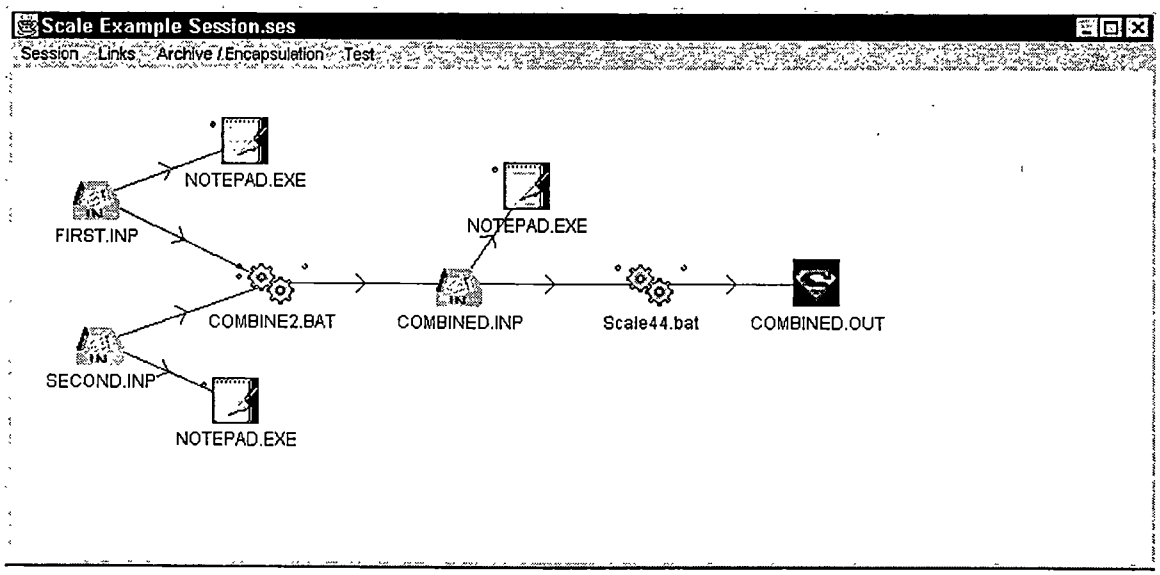


Figure 5. Session Window after Encapsulation

There is also a simple Java text editor included in the product for interactive creation or editing of text files. The default text editor is not a separate file in the Pyxis environment but may be accessed through the "File" menu of any folder window. The user may also incorporate other text editors to be used in Pyxis as imported applications. The default text editor is designed to be extremely user-friendly and intuitive. Figure 6 shows the text editor open for modification of a sample SCALE input file.

Finally, a complete HTML help index is provided with the product. This help menu covers all of the features of the product. The help index details how to incorporate new applications, how to use the text editor, how to get started, how to use the Session Window, and much more. The intent is for users to have all the information needed to successfully use the product available within the product itself. The help menu makes this possible because it serves as a complete user's manual within the product itself. A complete set of pages from the help index may be seen in Appendix B.

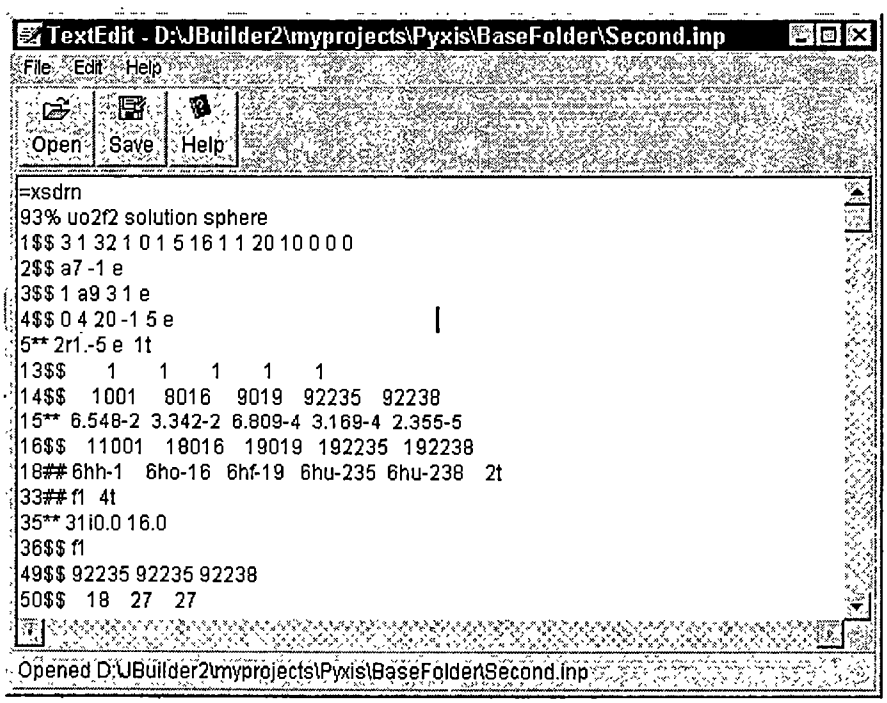


Figure 6. The Text Editor

Chapter Four

An Example of Pyxis Use

Problem Introduction

To demonstrate the usefulness of Pyxis we consider a simple SCALE sample problem. This problem is one of the first sample problems used in the Introduction to Nuclear Criticality Safety course (NE421) offered by The Nuclear Engineering Department at The University of Tennessee. The problem considers a sphere with a five centimeter radius consisting entirely of Plutonium-239.

This sample problem is normally analyzed using the CSAS1X sequence that runs BONAMI for unresolved resonance processing and NITAWL for resolved resonance processing. XSDRNPM completes the CSAS1X sequence, performing the actual 1D S_N spherical calculation.

The user input to SCALE shown in Figure 7 is minimal for this problem. The user specifies what group library will be used (27GROUPNDF4), the type of geometry used in resolving resonances (MULTIREGION), the material contents (100% Pu-239 @293°K, at 100% normal density for Pu from the SCALE materials library), and the sphere shell material and outer dimensions (1 shell of material #1 with an outer radius of 5cm). Material 2 is not actually used in the problem but is included because it is used in the Introduction to Nuclear Criticality Safety class for later variations of the sample problem.

The main usefulness of SCALE is that SCALE takes the relatively simple input discussed above and restructures it for the various components of the calculational sequence. Input not specified by the user is determined by SCALE using defaults and internal algorithms. In this example, input information determined by SCALE includes:

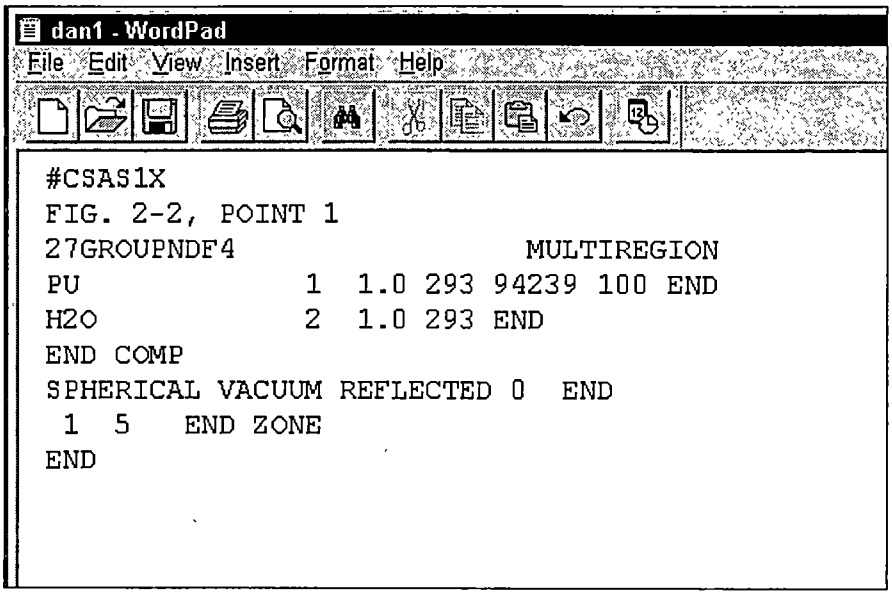


Figure 7. Example Input File

boundary conditions, convergence criteria, number of outer and inner iterations, the correction factor for Wigner rational approximations in resonance processing, a one dimensional approximate geometry for resolved resonance processing, the spatial grid used in the one dimensional S_N calculation, the angular quadrature used in the one dimensional S_N calculation, and the order of P_ℓ scattering.

In Pyxis, this same problem is broken into its constituent parts, BONAMI for processing the unresolved resonances, NITAWL for processing the resolved resonances and XSDRNPM to run the actual S_N calculation. By breaking it up like this, Pyxis allows the analyst to view the input and outputs for each step of the sequence and provides an environment to easily modify any of the input parameters of the calculation, which increases the analyst's control.

Getting Pyxis Started

Pyxis is available for installation on all platforms that support Java technology. Once Pyxis is installed, it can be run as any other executable program on the host platform, (e.g., by double clicking on an icon or typing "Pyxis" on the command line). Because this process will vary from platform to platform, instructions will be made available to users with their installation package.

Setting up Folders

After initializing the program, the user may create a folder system to organize and store items that will later be imported into the Pyxis environment. To do this the user selects the "New – Folder" menu item from the "File" Menu of the Pyxis base folder window or any other folder window in the Pyxis environment. Figure 8 shows selection of the appropriate menu item for creating a new folder. The user may also hold down the "Control" key and press the "N" key as a shortcut to creating a new folder.

After selecting the "New – Folder" menu item a new folder will be created and will be seen on the screen. A default name such at "New Folder 1" will be assigned the folder and the user will be prompted to give the folder a new name. A directory is also

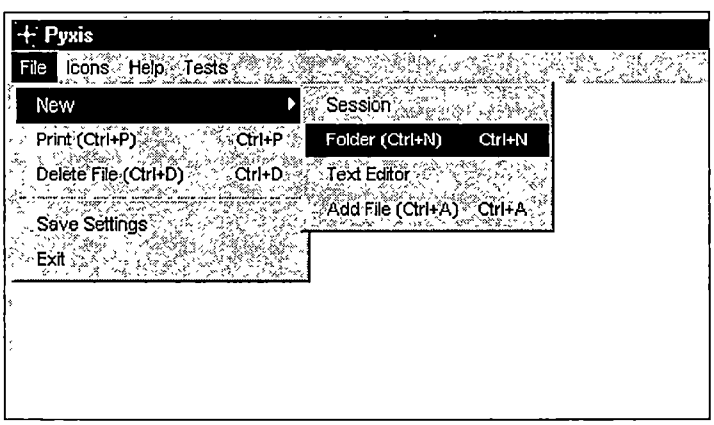


Figure 8. The New - Folder Menu

created on the computer with the same name as the folder. Figure 9 shows the creation of "New Folder 1" along with the user prompt to rename the folder. The user may create any number of folders in this manner and may create subfolders within the newly created folders in the same way.

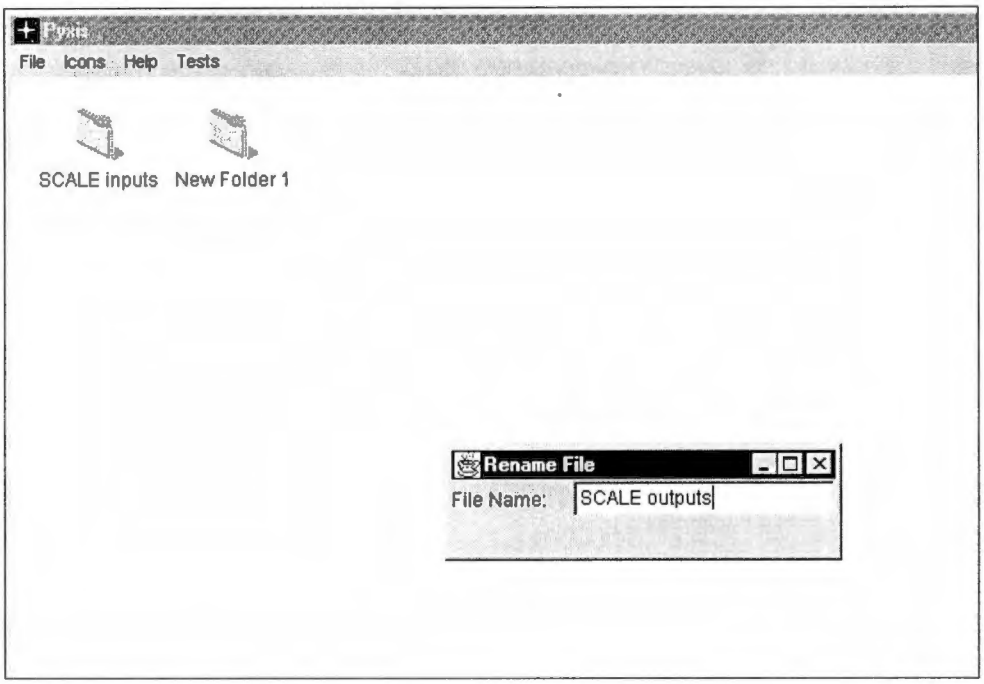


Figure 9. Naming Newly Created Folders

Incorporating Existing Tools into the Pyxis Environment

After creating some folders, the user will want to begin importing applications and other files into the Pyxis environment. The incorporation of applications and files into the Pyxis environment is what gives the product its power. To incorporate a new file into Pyxis the user selects "New - Add File" from the "File" Menu. A dialog will appear showing the files and directories of the host computer. The user navigates through the directories and selects the file to be imported into the Pyxis Environment. The user then clicks on the "Open" button and the file is imported into the environment. This means

that the file is moved from its previous location into the Pyxis folder that the user was working in. As a shortcut the user could hold down the "control" button and press "a" to get the Add File dialog box to appear without using the menus mentioned above. Figure 10 shows the dialog box created by Pyxis for importing files.

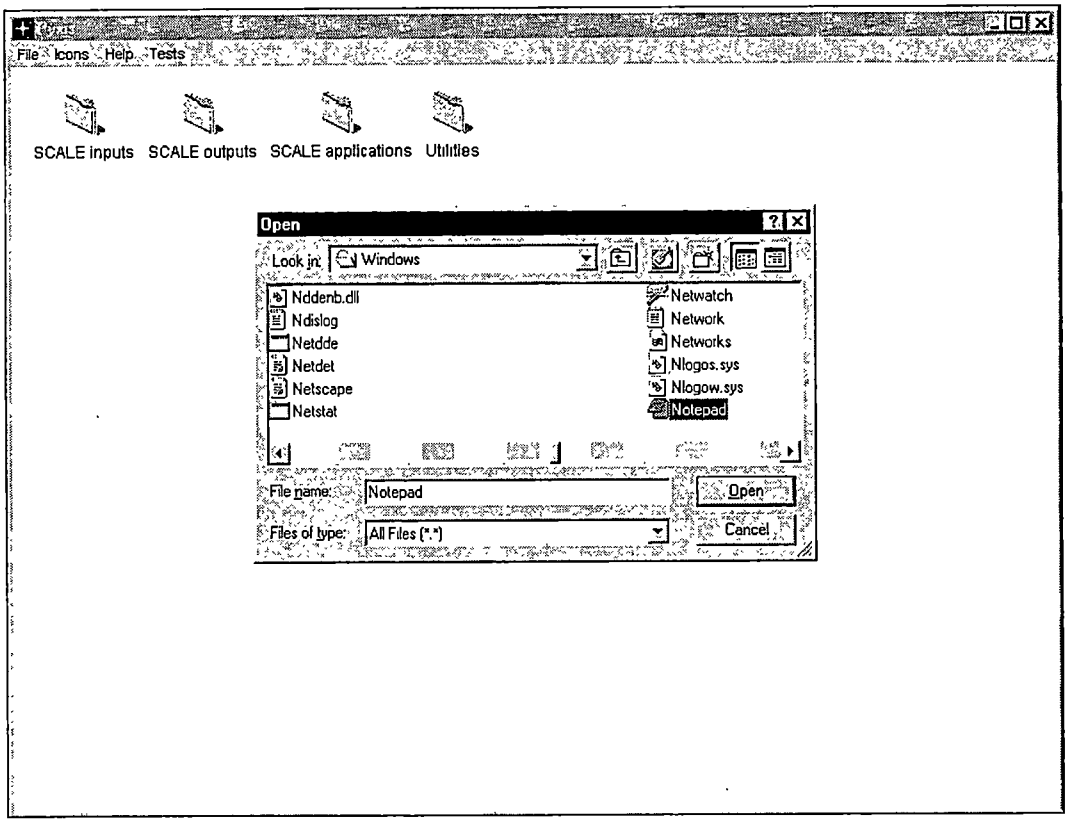


Figure 10. Incorporating Existing Applications

Creating New Files in the Pyxis Environment

After incorporation of existing applications and files into the Pyxis environment the user may want to create new files for use in various analyses. This can be done in several ways. The user can use any of the imported tools for creation of new files, or the user can use the default text editor supplied in Pyxis to create a text file. To use the

default text editor, the user selects the "New – Text Editor" menu item from the "File" menu. Then the user may create a simple text file and save it with any type of file extension. Based on the file extension the file might be used as an input file, output file, batch file, or another type of text file. Figure 11 shows the user saving an input file recently edited in the default text editor.

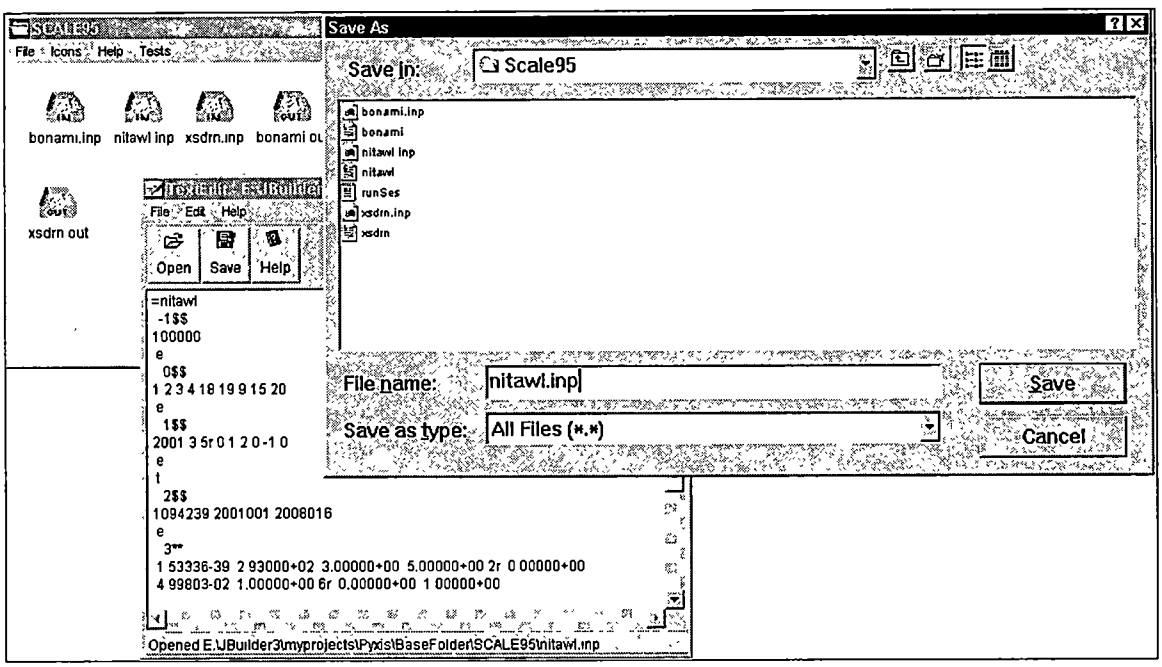


Figure 11. Creation of Input Files

Importing Items into the Session Environment

After importing and creating files for organizational use in the Pyxis environment, the user may wish to create a session for performing a calculation. This is done by selecting the "New – Session" menu item from the "File" menu of any folder window. An empty session window will appear with the title "New Session."

Items may be imported into the session two different ways. Items in Pyxis folder windows may be simply dragged into the session window. Icons representing the items

will appear in the session window and the original item will remain in the folder window. The second method for importing applications or other files into the session window is by selecting the "Add Session Element" menu item from the "Session" menu of the session window. A dialog will appear allowing the incorporation of any file on the computer into the session. Figure 12 shows an example of the dialog appearing after a user selected the "Add Session Element" menu item. Figure 12 also shows an open folder window containing several files that were previously dragged into the session window. When items are brought into a session, they are not actually moved from their original directory. The session window does not represent any directory but may contain representations of items located in many different directories.

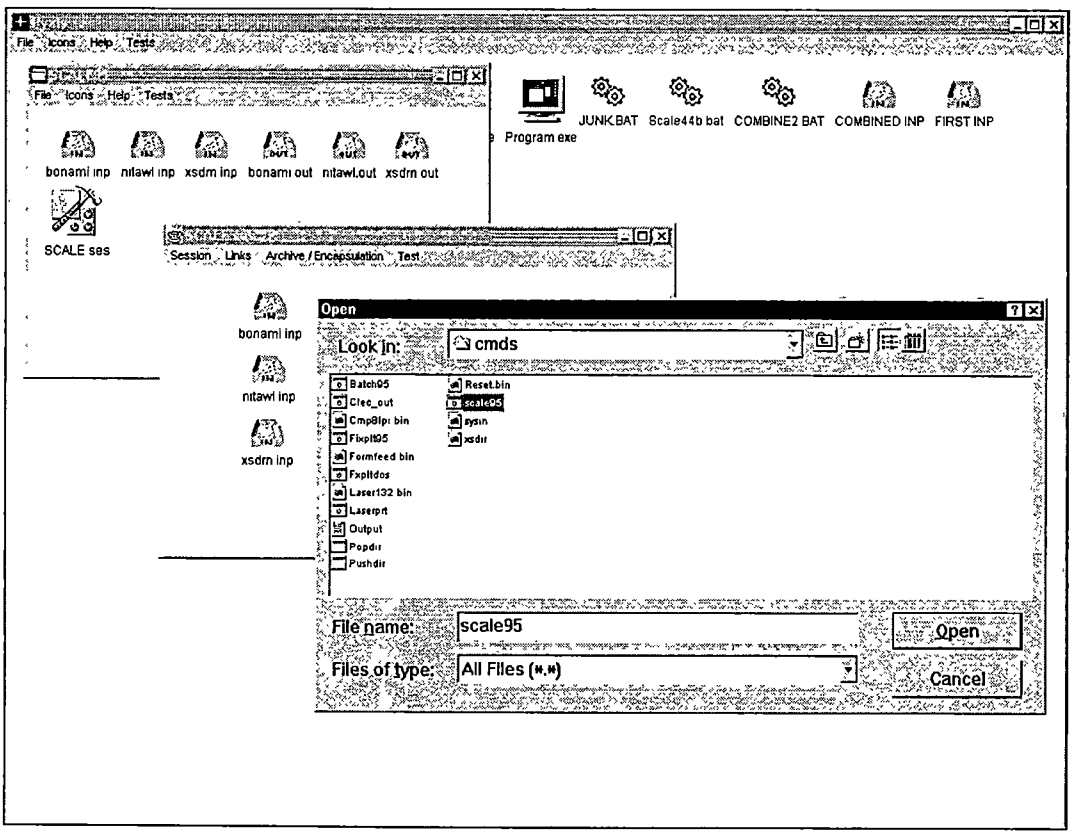


Figure 12. Importing an Application into a Session

Setting up the Calculational Series

Once all the parts for setting up an analysis have been incorporated into the session environment, the analyst can begin to set up the calculation itself. First some information must be supplied about the applications in the session window. The user selects each application and then chooses the "Set Application Properties" menu item from the "Session" menu. Figure 13 shows the "Session" menu with the "Set Application Properties" menu item selected.

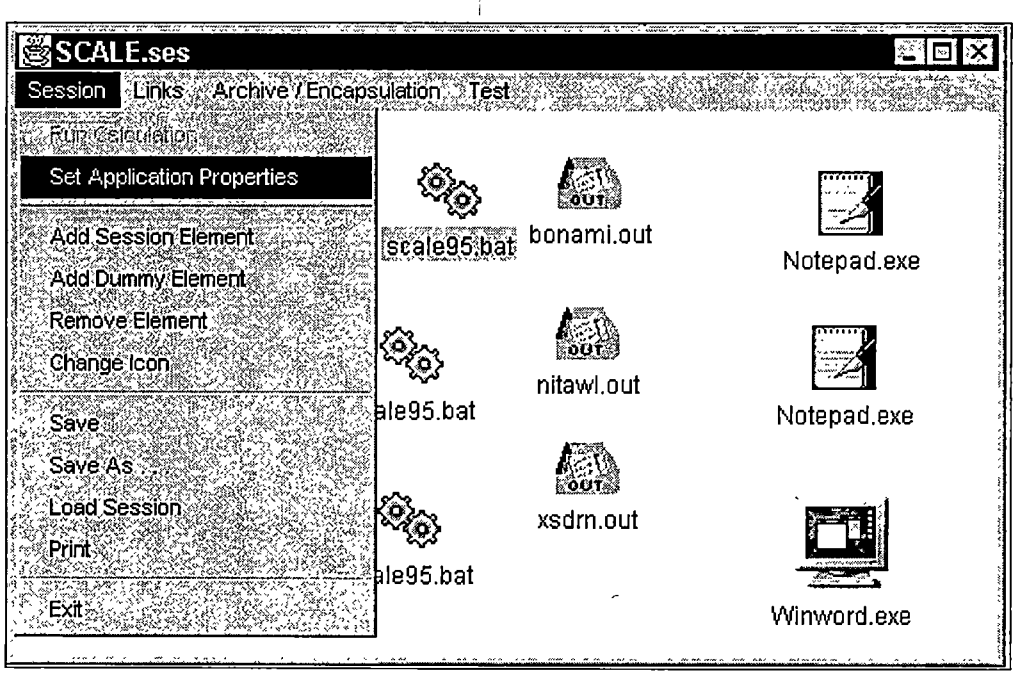


Figure 13. The Session Menu

When the user selects the "Set Application Properties" menu item a graphical user interface appears for describing the application. Pyxis must know how many inputs and outputs the application requires to function properly. The default settings for applications assume the application requires a single specified input with no specified

output. Figure 14 shows the application properties being set for the SCALE driver, which requires one specified input file and one specified output location. The interface on the lower left allows the user to describe the properties of the application. The window to the right of that interface provides a description of all properties for all input and output files, as previously set for the application. In the session window the user can tell how many inputs are required for an application by the number of green dots to the left of the application's icon. Similarly, the number of specified outputs required by the application is represented by red dots to the right of the application's icon.

Once the properties for the various applications have been defined, the analyst will begin to set up links. The user first selects the "Add Link" menu item from the "Links" menu in the session window. The "Links" menu is shown in Figure 15 with the "Add Link" menu being selected.

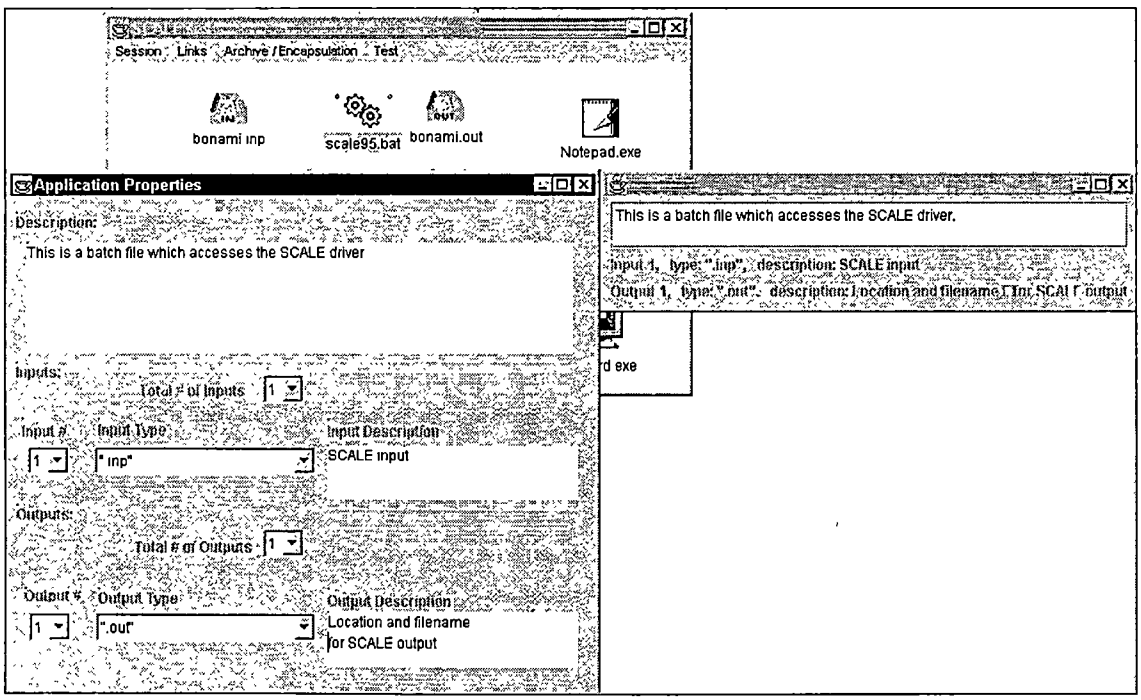


Figure 14. Setting the Application Properties

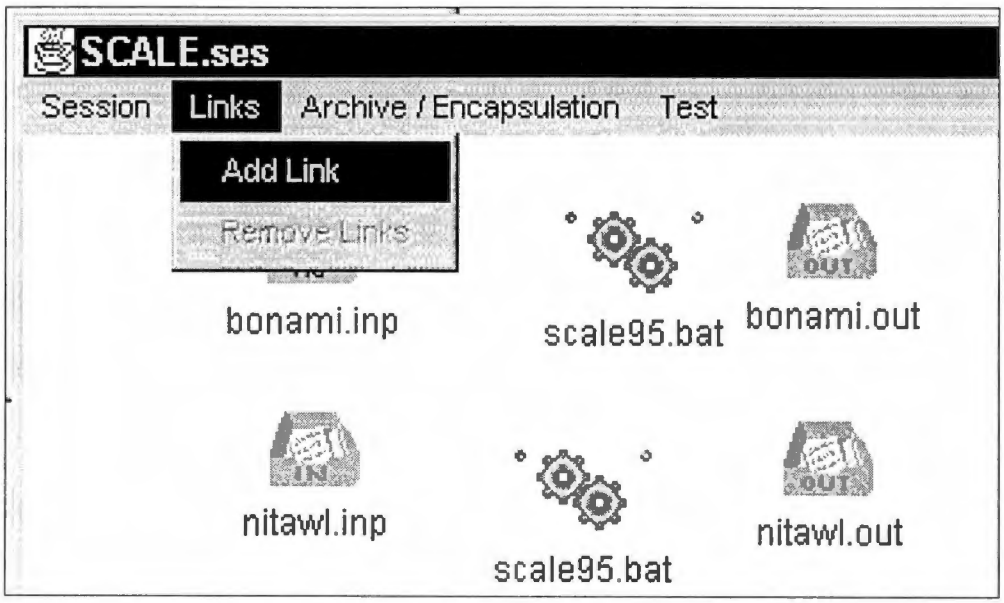


Figure 15. Add Link Menu Item

When the "Add Link" menu item is selected, the user's default cursor will be replaced with a crosshair cursor for creation of the link. The user clicks on the session element to link from then holds the mouse button down and drags the cursor to the session element to link to. The user will see a line being created as the cursor is dragged from the first element to the second. The user releases the mouse on the second session element. If the link was successfully created, the user will be notified by a beep and will see the link between the two elements with an arrow pointing toward the session element being linked to. Figure 16 shows a link being created from the "nitawl.inp" input file to the "scale95.bat" application. Figure 16 also shows several properly constructed links.

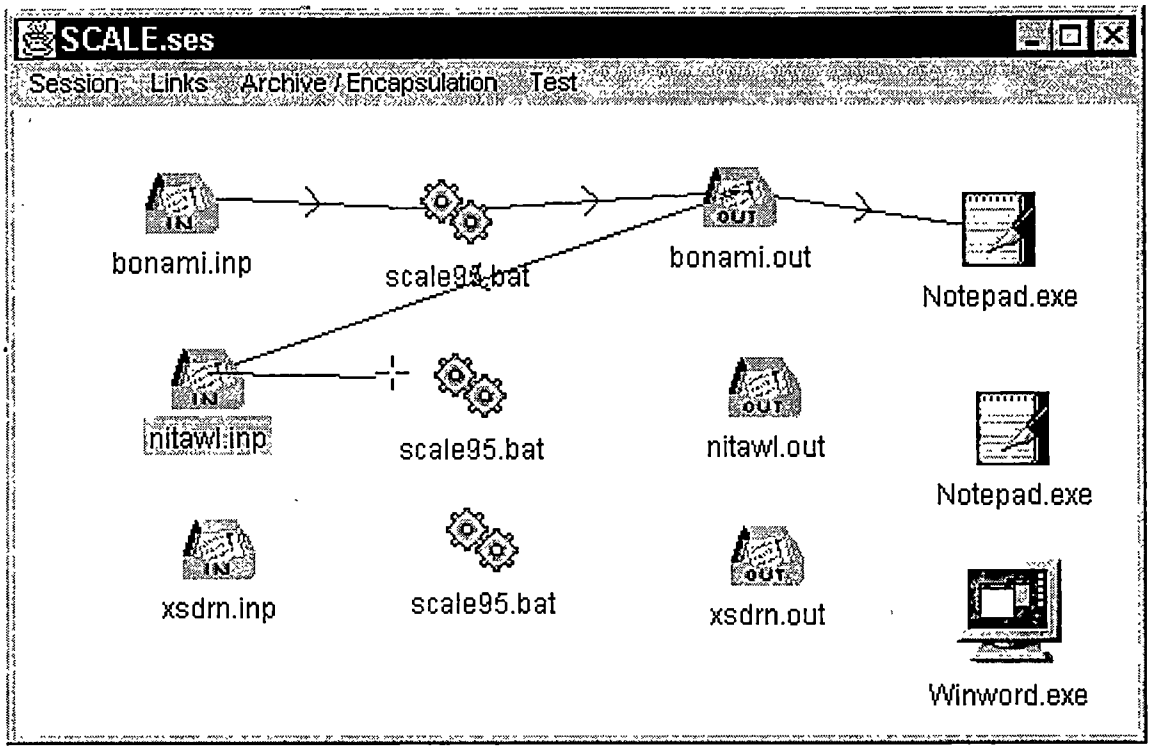


Figure 16. Creation of Links

Running the Calculations

The analyst sets up links one by one until the entire analysis is created. Figure17 shows an example of a complete analysis. In this example, the input for the BONAMI part of the sequence is run first generating an output (bonami.out) which is viewed using Notepad. Next, the NITAWL calculation is performed and the output is viewed using Notepad. Finally, the XSDRNPM calculation is performed and the results are viewed using Winword.

After completely setting up the analysis the calculational series is run by selecting the "Run Calculation" menu item from the "Session" menu. The links will be executed in the order created until the calculational series is completed or input is

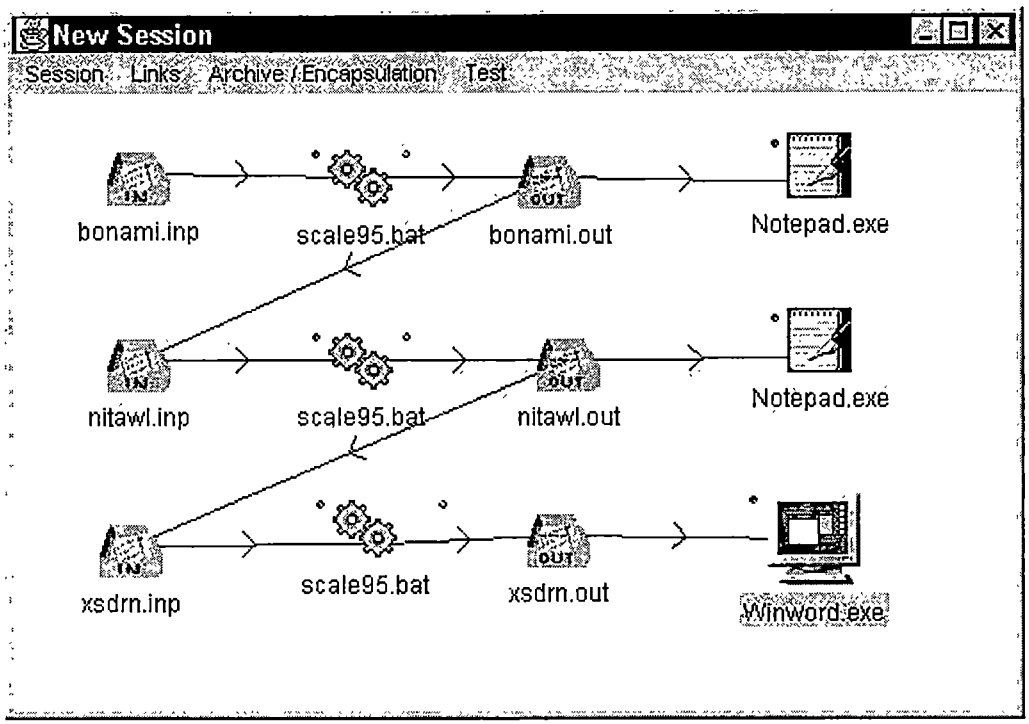


Figure 17. The Finalized Analysis

needed from the user. For this example, the final output file is displayed in the Pyxis environment by the Winword application as seen in Figure 18.

Comparison of Results

The results generated by breaking the calculation into three parts in the Pyxis environment were compared to the results obtained by running the original input file (see Figure 7) in SCALE. Figure 18 shows the section of the output as generated in Pyxis displaying the eigenvalue representing the calculated effective k value of 1.013. The results were identical to those generated by using SCALE in the traditional manner. This example demonstrates the flexibility that Pyxis provides for manipulating linked calculations similar to those processed by SCALE.

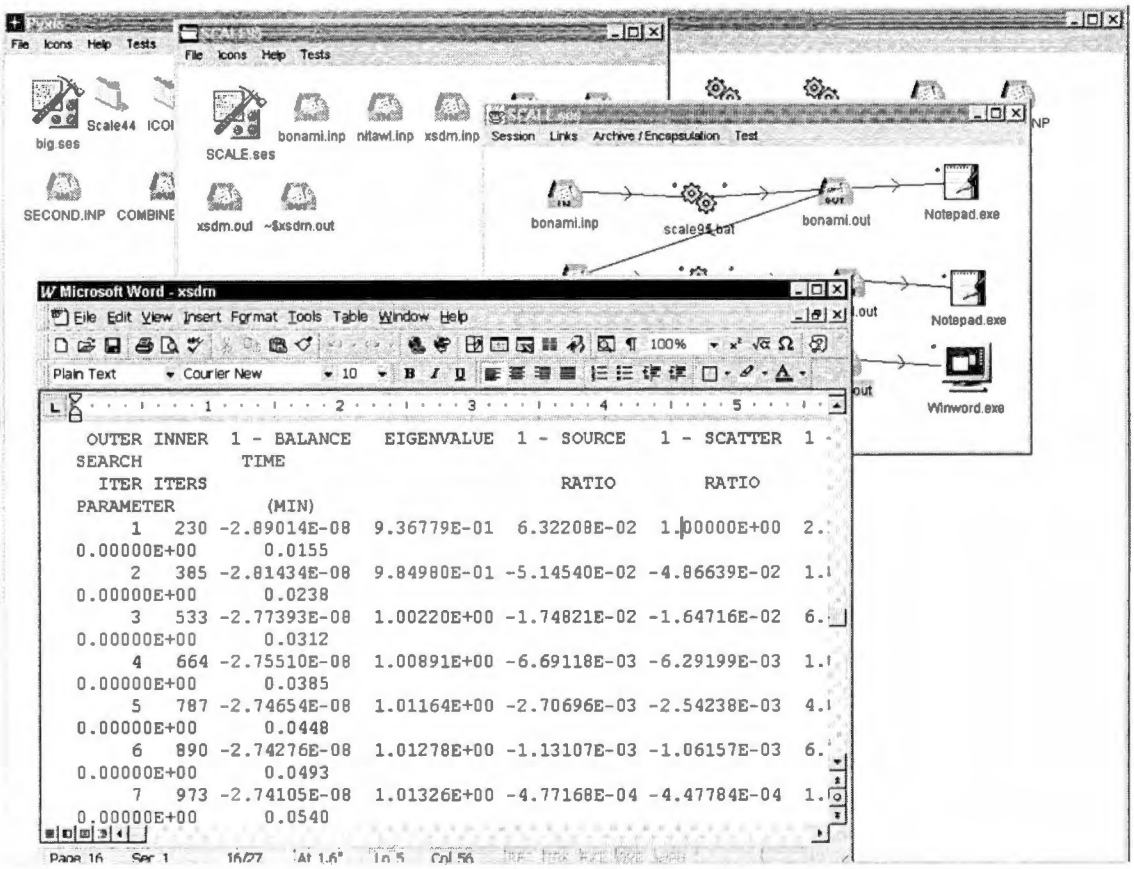


Figure 18. Examining the Results

Archiving the Calculation

The calculational series may be saved for later use or for reporting purposes in several ways. The session itself may be saved by choosing the "Save" or "Save As . . . " menu item from the "Session" menu. The session will be saved in a file with the extension "ses." Saved sessions may be reopened in Pyxis by double clicking on their icons. Selecting the "Load" menu item from the "Session" menu and selecting the appropriate file for loading can also reopen previously saved sessions. The sessions will function properly as long as none of the files represented in the session window have moved from their previous directory.

There are two other methods to save information contained in the session window. The "Archive / Encapsulation" menu provides several options for archiving a calculational series. The user may archive some or all of the files contained in the session by responding to user prompts created by Pyxis following selection of the "Archive" menu items. The user may also simply print the graphical information contained in the session window by selecting the "Print" menu item from the "Session" menu. This may be useful for communicating to another analyst information about the analysis.

Chapter Five

Importance of the Product

Applications for the Pyxis Product in its Current State

The most obvious use for this product in the short term is by nuclear engineering students at the University of Tennessee in their coursework, design projects and graduate research.

Many department upper division courses provide an intensive time-restrictive user training environment. Students are often introduced to computer codes from the nuclear engineering field for the first time in classes as examples illustrating the scientific and engineering concepts being studied. Often a significant portion of time must be devoted to explaining how to access and run the codes. This process will be greatly simplified through use of Pyxis. The students will have all the tools needed to create input files and run the calculations in a familiar user-friendly environment. More time will be available for teaching students theory and use of the codes rather than how to make them run.

The continual stream of students entering the program and graduating to engineering analysis positions creates several advantages for the development and dissemination of Pyxis. The students will serve as beta testers for future improved versions of the product, which will be made available to a larger set of users. As the students become professionals, the graphical Pyxis environment will be introduced to the workplace.

Modifications for Extended Product Use

A few modifications could make the product significantly more useful to an extended number of users. These modifications range from simple to extremely involved. I will make many of these modifications myself as I continue my work at the university. The source code of the product will also be made available to future students for continued development. Appendix C contains notes to future Pyxis developers.

One modification would be to provide more assistance for archiving calculations in the Pyxis Environment. Currently the calculations are saved as they are created in the session environment, but this is only useful to someone with the Pyxis Environment and would not be useful if applications or other files are moved from their original location. It would be more useful to be able to copy and save user specified files to a user-specified destination such as a disk or CD after performing a calculation. A visual representation of the linked calculation could also be saved (a .gif file showing the Session Window). This modification to the product would not be extremely difficult and is recommended as an early modification to the next version of the product.

The usefulness of the Pyxis environment is greatly enhanced by the incorporation of tools. When the product is given to users it could be supplemented by including tool sets for use within the product. These tools could be various input or output processors, control structures, visualization tools, etc. Graduate students could write and add tools to the environment. The graduate students could also create sample sessions showing how to best incorporate the new tools into a linked analysis. Such tools are valuable by themselves, but when incorporated into the Pyxis environment, these tools become even more powerful because they are easier to use and distribute.

Use of Pyxis Coding in Other Projects

The object oriented structure of the Java language also opens up possibilities for later use of this product. The program is very easy to modify and expand allowing for flexibility in meeting future users needs. The product could be used in part or in its entirety in another application. Because the product is object oriented, a programmer would not have to be familiar with the coding of Pyxis to use parts of the product in new applications.

Several Pyxis objects might be useful by themselves for incorporation into other codes. The FolderWindow class creates the visual folder windows tied to directories on the computer. The FolderWindow class would be useful for creating a platform-independent windows foldering system. The PFile object ties a file to an icon and position. This class provides methods for changing the filename, icon and position and could be used for visualization of files in another Java application. The TextEditFrame class creates a stand alone text editor that could be incorporated into many potential applications.

Future Applications for Later Product Versions

The Pyxis product has a tremendous range of potential applications. The product could be used in other fields, particularly fields which have many older codes or which perform linked calculations. There is nothing in the program that prevents it from incorporating applications and files of any type from any technical field.

While Pyxis may be useful in many fields, the product is designed for use in the field of nuclear engineering. The archiving and tool sharing features were developed with nuclear engineers (specifically criticality analysts) in mind. The environment is suitable

for team projects where multiple people need to have an understanding of the analysis. The session environment provided by Pyxis is ideal for assisting technical review of an analysis or to generate graphical flow charts for an analysis report. The environment can assist an analyst performing work on a problem similar to one previously investigated if the work was done in the Pyxis environment. Pyxis could assist analysts who have their work interrupted and must turn over a half-completed analysis to a new analyst. The list of potential situations for which Pyxis is well suited is virtually infinite. It is my hope that this work will benefit many nuclear engineers by making their computational work less burdensome.

Summary and Conclusions

This product is unlike most computer codes available for use in the field of nuclear engineering. Pyxis does not actually perform criticality calculations. It is not a visualization tool for interpreting input or output files. Pyxis is a workbench where tools are used and stored. It displays the relationship between various tools and allows the sharing of tools between analysts. It assists in peer review and archiving and saves the analyst time. Pyxis is designed to prevent some of the frustration that can result from performing complicated linked calculations. The intent is to improve the quality of the analyst's working environment in order to improve the quality of the analyst's work.

REFERENCES

REFERENCES

1. *Glossary of Java Related Terms.* Sun Microsystems. http://java.sun.com/docs/glossary.html.

2. *The Java Language: An Overview.* Sun Microsystems. http://java.sun.com/docs/overviews/java/java-overview-1.html.

3. Gosling, James and Henry McGilton. *The Java Language Environment, A White Paper.* May 1996.

4. Kramer, Douglas. *The Java Platform, A White Paper.* May 1996.

5. Lee, Ronald W. personal interview. October 1999.

6. Blakeman, Edward D. personal interview. October 1999.

APPENDICES

APPENDIX A

Documentation of Public Classes in the Pyxis Code

The term "public" is a keyword in the Java language that refers to classes, methods and variables that can be accessed by other classes. "Private" classes, methods and variables may only be used within the object where they are created. By studying the public classes and their public methods and variables one is able to understand how objects interact with each other and the user when the program is run. This documentation only covers the public classes, methods and variables. Documentation of all classes, methods and variables would be many times longer and is not necessary to understand the structure of the product's coding.

This documentation was generated using the JavaDoc tool provided by Sun Microsystems. JavaDoc generates documentation directly from the Java source code. JavaDoc creates HTML files that show the structure of the public classes as well as notes about these classes taken from comments within the source code.

This appendix contains two of the documents generated by JavaDoc for the Pyxis source code. The first document is the class hierarchy for the public classes of Pyxis. The class hierarchy shows the parents for the various public objects in Pyxis. The second document is an index of all public classes, methods and variables found in the Pyxis source code. The document lists the classes, methods and variables; tells where they are located; and often gives a brief description of their function.

Hierarchy For All Packages

Class Hierarchy

- class Pyxis.**PFrame**
  - class Pyxis.**BaseFrame**
    - class Pyxis.**FolderWindow**
- class java.lang.Object
  - class java.awt.Component (implements java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable)
    - class java.awt.Container
      - class java.awt.Window
        - class java.awt.Dialog
          - class Pyxis.**PasswordDialog**
          - class Pyxis.**SuperAppDialog**
  - class Pyxis.**PFile** (implements java.io.Serializable)
  - class Pyxis.**PFolder**
  - class Pyxis.**Pyxis**
  - class Pyxis.**Session** (implements java.io.Serializable)
  - class Pyxis.**SessionElement** (implements java.io.Serializable)
  - class Pyxis.**SuperApp** (implements java.io.Serializable)
- class Pyxis.**PropDescBox** (implements java.io.Serializable)
- class Pyxis.**PropertiesBox** (implements java.io.Serializable)
- class Pyxis.**PropFileBox** (implements java.io.Serializable)
- class Pyxis.**SessionWindow** (implements java.io.Serializable)

☐ class Pyxis.**TextEditFrame**

A

addElement(File) - Method in class Pyxis.PFolder

Creates a PFile from the File and adds the PFile to the vector of PFiles.

addElement(PFile) - Method in class Pyxis.PFolder

Adds a PFile to the vector of PFiles.

addElement(SessionElement) - Method in class Pyxis.Session

addPFile() - Method in class Pyxis.PFrame

This method opens a dialog to move a file into the folder associated with this frame.

addPFile() - Method in class Pyxis.SessionWindow

Allows user to add a File to the Session.

allFiles - Static variable in class Pyxis.PFile

arrangeByName() - Method in class Pyxis.PFrame

This method will arrange the PFiles alphabetically within the window when it works.

arrangeByType() - Method in class Pyxis.PFrame

This method arranges PFiles based on their file extensions.

arrangeIcons() - Method in class Pyxis.PFrame

Arranges the icons on the screen.

B

BaseFrame - class Pyxis.BaseFrame.

This creates the various members of the Frame.

BaseFrame() - Constructor for class Pyxis.BaseFrame

Construct the frame.

buttonBar_actionPerformed(ActionEvent) - Method in class Pyxis.BaseFrame

This method responds to button bar actions.

C

cancelButton_actionPerformed(ActionEvent) - Method in class Pyxis.PasswordDialog

Responds to Cancel button action by disposing of the window.

changeFileName(PFile) - Method in class Pyxis.PFrame

This method allows the user to change the filename of the selected PFile.

clear - Variable in class Pyxis.PFrame

close() - Method in class Pyxis.SessionWindow

Closes the window after prompting the user to save.

copy() - Method in class Pyxis.TextEditFrame

Copy text to the System Clipboard.

copyTo(PFile) - Method in class Pyxis.PFile

Copies the file if possible

createFolder() - Method in class Pyxis.BaseFrame

This method creates a new directory and a corresponding icon.

cut() - Method in class Pyxis.TextEditFrame

Cut = Copy selected text to the System Clipboard then delete selected text from textArea.

D

delete(PFile) - Method in class Pyxis.PFolder

Removes the given PFile from the vector of PFiles.

deleteFile(PFile) - Method in class Pyxis.PFrame

This method deletes the PFile.

description - Variable in class Pyxis.PropertiesBox

dialog - Variable in class Pyxis.SuperApp

disableMenu() - Method in class Pyxis.BaseFrame

This method disables certain menu Items when there is no selected File.

E

enableMenu() - Method in class Pyxis.BaseFrame

This method enables appropriate menu Items when there is a selected File.

enableMenu2() - Method in class Pyxis.SessionWindow

This method enables menu Items when there is a link present.

exit() - Method in class Pyxis.TextEditFrame

Exit TextEditor if it is OK.

F

fileExit_actionPerformed(ActionEvent) - Method in class Pyxis.BaseFrame

File | Exit action performed.

fileExit_actionPerformed(ActionEvent) - Method in class Pyxis.FolderWindow

File | Exit action performed.

fileExit_actionPerformed(ActionEvent) - Method in class Pyxis.TextEditFrame

File | Exit action performed.

files - Variable in class Pyxis.PFolder

FolderWindow - class Pyxis.FolderWindow.

FolderWindow(PFolder) - Constructor for class Pyxis.FolderWindow

---

G

getAddDirectory() - Method in class Pyxis.SessionWindow

Returns the base directory for opening the Add Element dialog.

getDirectory() - Method in class Pyxis.PFolder

Returns the directory file.

getElements() - Method in class Pyxis.Session

Returns the vector of SessionElements.

getFile() - Method in class Pyxis.PFile

returns the file.

getFolder() - Method in class Pyxis.PFrame

Returns the PFolder for the Frame.

getGroup() - Method in class Pyxis.SessionWindow

Returns the grouping rectangle for the session.

getHeight() - Method in class Pyxis.PFile

Returns previously set height for the visual representation of the PFile

getIcon() - Method in class Pyxis.PFile

Returns the icon image.

getItem(int) - Method in class Pyxis.PFolder

Returns the PFile at the given index of the vector of PFiles.

getItem(int) - Method in class Pyxis.Session

getLinks() - Method in class Pyxis.SessionWindow

Returns the vector of links.

getLongName() - Method in class Pyxis.SessionElement

getName() - Method in class Pyxis.PFolder

Returns a string representation of the directory name.

getName() - Method in class Pyxis.SessionElement

getPDirectory() - Method in class Pyxis.PFolder

Returns the PFile representation of the directory.

getSelectedFile() - Method in class Pyxis.PFrame

Returns the selected (highlighted) file for the frame.

getSession() - Method in class Pyxis.SessionWindow

getSize() - Method in class Pyxis.PFolder

Returns the number of PFiles in the PFolder minus one (numbering starts at 0).

getSize() - Method in class Pyxis.Session

Same as size() in class Vector.

getSuperApps() - Method in class Pyxis.Session

getType() - Method in class Pyxis.SessionElement

getWidth() - Method in class Pyxis.PFile

Returns previously set width for the visual representation of the PFile

getX() - Method in class Pyxis.PFile

Returns the x coordinate for the upper left hand corner of the icon.

getY() - Method in class Pyxis.PFile

Returns the x coordinate for the upper left hand corner of the icon.

H

height - Variable in class Pyxis.SessionElement

helpAbout_actionPerformed(ActionEvent) - Method in class Pyxis.BaseFrame

Help | About action performed.

helpAbout_actionPerformed(ActionEvent) - Method in class Pyxis.TextEditFrame

Help | About action performed.

helpAbout() - Method in class Pyxis.TextEditFrame

Coding for Help About commands.

I

icon - Variable in class Pyxis.SessionElement

iconAddress - Variable in class Pyxis.PFile

iconAddress - Variable in class Pyxis.SessionElement

inDesc - Variable in class Pyxis.PropertiesBox

inputs - Variable in class Pyxis.PropertiesBox

inType - Variable in class Pyxis.PropertiesBox

isDirectory - Variable in class Pyxis.SessionElement

isDirectory() - Method in class Pyxis.PFile

Returns true if the file is a directory or false if the file is not a directory.

J

jbInit() - Method in class Pyxis.BaseFrame

This method actually constructs the Frame.

jbInit() - Method in class Pyxis.PasswordDialog

This initialization method is called by all the constructors.

jbInit() - Method in class Pyxis.SuperAppDialog

jbInit() - Method in class Pyxis.TextEditFrame

Component initialization.

L

linkFrom - Variable in class Pyxis.SuperApp

linkTo - Variable in class Pyxis.SuperApp

load() - Method in class Pyxis.BaseFrame

This method loads the settings written to file by the save command.

M

main(String[]) - Static method in class Pyxis.Pyxis

This is the main method for this application.

N

name - Variable in class Pyxis.SessionElement

name - Variable in class Pyxis.SuperApp

nameLong - Variable in class Pyxis.SessionElement

newFolderNumber - Variable in class Pyxis.BaseFrame

O

okButton_actionPerformed(ActionEvent) - Method in class Pyxis.PasswordDialog

Responds to OK button action by checking for the correct password.

okToAbandon() - Method in class Pyxis.TextEditFrame

Checks if file needs saving before leaving TextEditor.

open - Variable in class Pyxis.PFile

originalFile - Static variable in class Pyxis.PFrame

originalFile - Static variable in class Pyxis.SessionWindow

originalWindow - Static variable in class Pyxis.PFrame

originalWindow - Static variable in class Pyxis.SessionWindow

outDesc - Variable in class Pyxis.PropertiesBox

outputs - Variable in class Pyxis.PropertiesBox

outType - Variable in class Pyxis.PropertiesBox

P

paint(Graphics) - Method in class Pyxis.PFrame

This method paints the PFiles on the Frame.

paint(Graphics) - Method in class Pyxis.SessionWindow

PasswordDialog - class Pyxis.PasswordDialog.

PasswordDialog(Frame) - Constructor for class Pyxis.PasswordDialog

PasswordDialog(Frame, String) - Constructor for class Pyxis.PasswordDialog

PasswordDialog(Frame, String, boolean) - Constructor for class Pyxis.PasswordDialog

passwordTextField_keyPressed(KeyEvent) - Method in class Pyxis.PasswordDialog

Checks for the "enter" command and treats it as an OK button action.

paste() - Method in class Pyxis.TextEditFrame

Paste Text from the System Clipboard.

pBox - Variable in class Pyxis.SessionElement

pdb - Variable in class Pyxis.PropertiesBox

pFile - Variable in class Pyxis.SessionElement

PFile - class Pyxis.PFile.

PFile(File) - Constructor for class Pyxis.PFile

PFile(PFile) - Constructor for class Pyxis.PFile

PFile(String) - Constructor for class Pyxis.PFile

pFileBox - Variable in class Pyxis.SessionElement

PFolder - class Pyxis.PFolder.

PFolder(PFile) - Constructor for class Pyxis.PFolder

PFrame - class Pyxis.PFrame.

PFrame() - Constructor for class Pyxis.PFrame

PFrame(PFolder) - Constructor for class Pyxis.PFrame

processWindowEvent(WindowEvent) - Method in class Pyxis.FolderWindow

Window closing action performed.

processWindowEvent(WindowEvent) - Method in class Pyxis.SessionWindow

processWindowEvent(WindowEvent) - Method in class Pyxis.TextEditFrame

This code deals with the exit window closing event.

PropDescBox - class Pyxis.PropDescBox.

PropDescBox() - Constructor for class Pyxis.PropDescBox

PropDescBox(PropertiesBox) - Constructor for class Pyxis.PropDescBox

PropertiesBox - class Pyxis.PropertiesBox.

PropertiesBox() - Constructor for class Pyxis.PropertiesBox

PropFileBox - class Pyxis.PropFileBox.

PropFileBox() - Constructor for class Pyxis.PropFileBox

Pyxis - class Pyxis.Pyxis.

Pyxis() - Constructor for class Pyxis.Pyxis

R

removeElement(SessionElement) - Method in class Pyxis.Session

removeElementAt(int) - Method in class Pyxis.PFolder

Removes the PFile at the given index.

removeElementAt(int) - Method in class Pyxis.Session

removeSesElement(SessionElement) - Method in class Pyxis.SessionWindow

renameTo(PFile) - Method in class Pyxis.PFile

Renames the file if possible.

reopen() - Method in class Pyxis.SuperApp

reset() - Method in class Pyxis.PropertiesBox

resetIcons() - Method in class Pyxis.PFrame

This method resets Icons to their default images.

S

save() - Method in class Pyxis.BaseFrame

Coding to write PFile information to a save file.

save() - Method in class Pyxis.SessionWindow

saveAs() - Method in class Pyxis.SessionWindow

Save the session to a session file.

sElements - Variable in class Pyxis.SuperApp

sesPFile - Variable in class Pyxis.SessionWindow

Session - class Pyxis.Session.

Session() - Constructor for class Pyxis.Session

SessionElement - class Pyxis.SessionElement.

SessionElement(PFile) - Constructor for class Pyxis.SessionElement

SessionElement(SuperApp) - Constructor for class Pyxis.SessionElement

SessionWindow - class Pyxis.SessionWindow.

SessionWindow() - Constructor for class Pyxis.SessionWindow

SessionWindow(Session) - Constructor for class Pyxis.SessionWindow

setAddDirectory(String) - Method in class Pyxis.SessionWindow

Sets the base directory for opening the Add Element dialog.

setDefaultIcon() - Method in class Pyxis.PFile

This method assigns a default icon based on the file extension.

setDirectory(File) - Method in class Pyxis.PFolder

Sets the directory file.

setFile(File) - Method in class Pyxis.PFile

Sets the file.

setFileIcon(PFile) - Method in class Pyxis.PFrame

This method allows the user to assign an icon to the selected file.

setFileIcon(SessionElement) - Method in class Pyxis.SessionWindow

Allows the user to set the icon for the selected file.

setFolder(PFolder) - Method in class Pyxis.PFrame

Assigns the PFolder for the Frame.

setHeight() - Method in class Pyxis.PFile

Default assignment of height for the visual representation of the PFile

setHeight() - Method in class Pyxis.SessionElement

setHeight(int) - Method in class Pyxis.PFile

Allows assignment of height for the visual representation of the PFile

setIcon() - Method in class Pyxis.PFile

Simply calls setDefaultIcon().

setIcon(String) - Method in class Pyxis.PFile

Sets the icon to the image stored at the given string address.

setIcon(String) - Method in class Pyxis.SessionElement

setPDirectory(PFile) - Method in class Pyxis.PFolder

Sets the pdirectory variable to the PFile which represents the directory

setPosition(int, int) - Method in class Pyxis.PFile

Assigns the x and y coordinates for the upper left hand corner of the icon.

setPosition(int, int) - Method in class Pyxis.SessionElement

setSaveDirectory(String) - Method in class Pyxis.SessionWindow

Sets the base directory for opening the Save dialog.

setSaveFile(String) - Method in class Pyxis.SessionWindow

Sets the default file for saving.

setSession(Session) - Method in class Pyxis.SessionWindow

setType(String) - Method in class Pyxis.SessionElement

setWidth() - Method in class Pyxis.PFile

Default assignment of width for the visual representation of the PFile

setWidth() - Method in class Pyxis.SessionElement

setWidth(int) - Method in class Pyxis.PFile

Allows assignment of width for the visual representation of the PFile

setX(int) - Method in class Pyxis.PFile

Assigns the x coordinate for the upper left hand corner of the icon.

setX(int) - Method in class Pyxis.SessionElement

setY(int) - Method in class Pyxis.PFile

Assigns the y coordinate for the upper left hand corner of the icon.

setY(int) - Method in class Pyxis.SessionElement

sLinks - Variable in class Pyxis.SuperApp

superApp - Variable in class Pyxis.SessionElement

SuperApp - class Pyxis.SuperApp.

SuperApp(Vector, Vector) - Constructor for class Pyxis.SuperApp

SuperAppDialog - class Pyxis.SuperAppDialog.

SuperAppDialog(SuperApp, String, boolean) - Constructor for class Pyxis.SuperAppDialog

---

T

TextEditFrame - class Pyxis.TextEditFrame.

TextEditFrame() - Constructor for class Pyxis.TextEditFrame

this_mouseEntered(MouseEvent) - Method in class Pyxis.PFrame

Handles mouse entered events.

this_mouseReleased(MouseEvent) - Method in class Pyxis.PFrame

Handles mouse release events.

this_windowActivated(WindowEvent) - Method in class Pyxis.BaseFrame

This is used to set the BaseFrame to the back and to check for new items within the directory which are not yet shown.

this_windowActivated(WindowEvent) - Method in class Pyxis.FolderWindow

This checks the window against the directory to be sure al items are shown.

this_windowClosed(WindowEvent) - Method in class Pyxis.BaseFrame

This method sets the status of the associated PFile to closed when the window is closed.

this_windowClosing(WindowEvent) - Method in class Pyxis.PasswordDialog

Diposes of the window.

tjbInit(PFolder) - Method in class Pyxis.PFrame

This is the initialization method called by the constructors

U

userPrompt1() - Method in class Pyxis.SuperApp

Prompts user for needed information upon creation of a superApp (Element to link from).

userPrompt2() - Method in class Pyxis.SuperApp

Prompts user for needed information upon creation of a superApp (Element to link to).

W

width - Variable in class Pyxis.SessionElement

X

x - Variable in class Pyxis.SessionElement

xOffset - Static variable in class Pyxis.PFrame

xOffset - Variable in class Pyxis.SessionWindow

Y

y - Variable in class Pyxis.SessionElement

yOffset - Static variable in class Pyxis.PFrame

yOffset - Variable in class Pyxis.SessionWindow

A B C D E F G H I J L M N O P R S T U W X Y

APPENDIX B

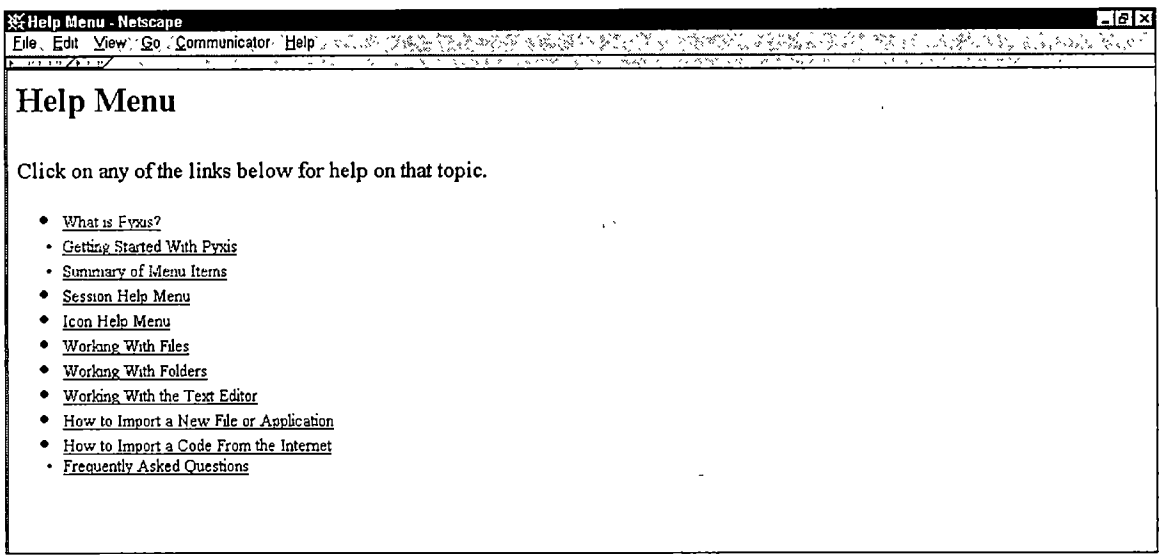
Online Help Documentation

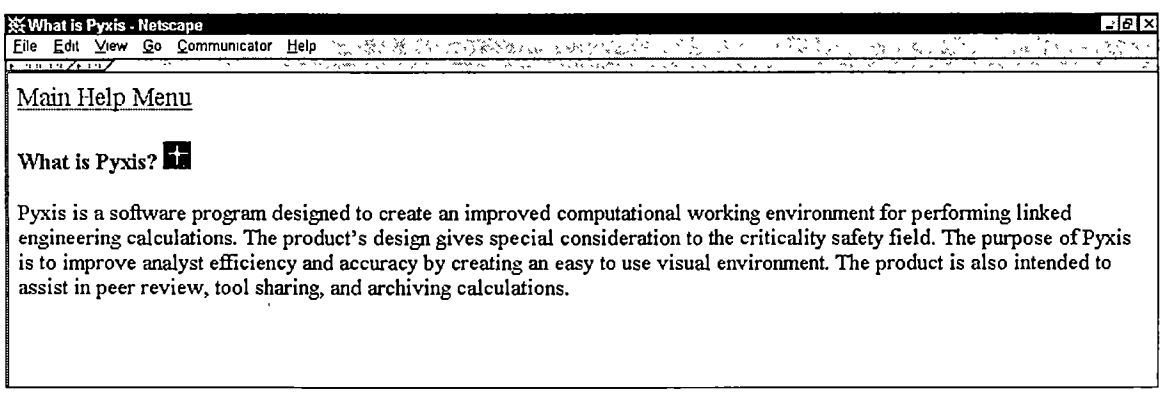
This Appendix shows the HTML help files as they appear to a Pyxis user. The help index is viewed by the default HTML viewer set for the computer running Pyxis.

APPENDIX B

Online Help Documentation

This Appendix shows the HTML help files as they appear to a Pyxis user. The help index is viewed by the default HTML viewer set for the computer running Pyxis.





Getting Started With Pyxis - Netscape

File Edit View Go Communicator Help

Main Help Menu

Getting Started With Pyxis

- First familiarize yourself with the foldering system. If the system is not intuitive to you see the working with folders help menu.
- Scroll through the various pull-down menus and test some of the features. See the Summary of Menu Items for help with any of these items.
- Next familiarize yourself with the Session Environment. Perhaps the best way to get started is to open and view some of the Sample Sessions available in the first Pyxis Window. Do this by double clicking on Sample.ses or one of the other available Session files. For help in this environment see the Session Help Menu.
- In the absence of any previously created sessions you may begin by creating your own session. See the Session Help Menu for details.

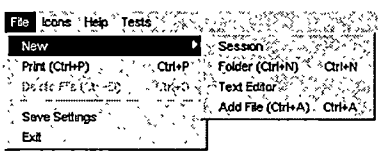
Summary of Menu Items - Netscape

File Edit View Go Communicator Help

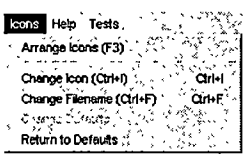
Main Help Menu

Summary of Menu Items

- File Menu Items



- Icons Menu Items



- Help Menu Items
  - Help Index
  - About Pyxis

File Menu Items - Netscape

File Edit View Go Communicator Help

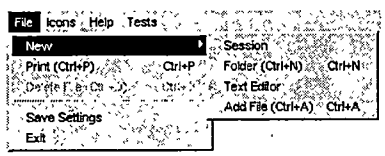
Main Help Menu

Summary of Menu Items

File Menu Items

- New
  - Session
  - Folder (Ctrl + N)
  - Text Editor
  - Add File (Ctrl + A)
- Print (Ctrl + P)
- Delete File (Ctrl + D)
- Save Settings
- Exit

Below is a picture of the File Menu Items



Session - Netscape

File Edit View Go Communicator Help

Main Help Menu

Summary of Menu Items

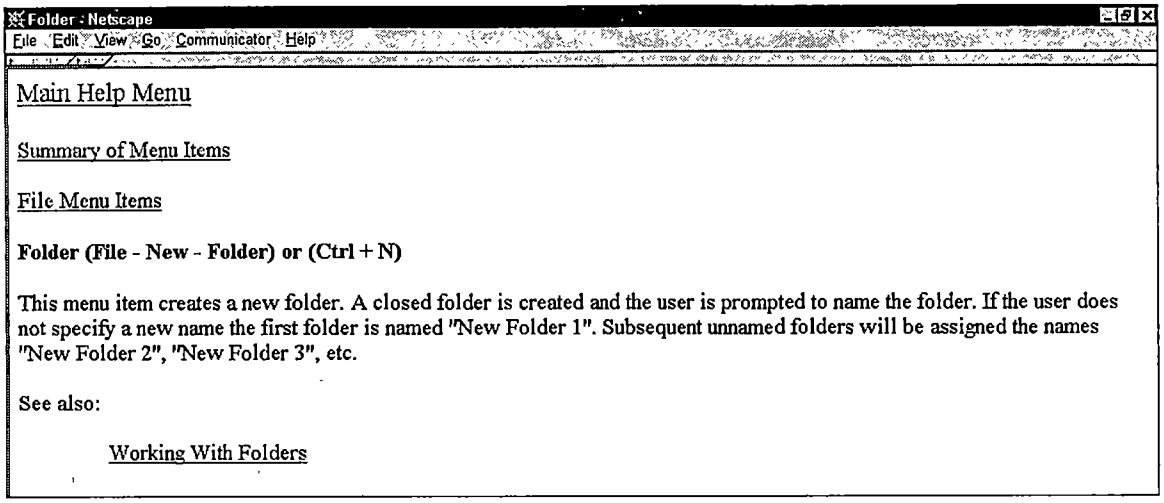
File Menu Items

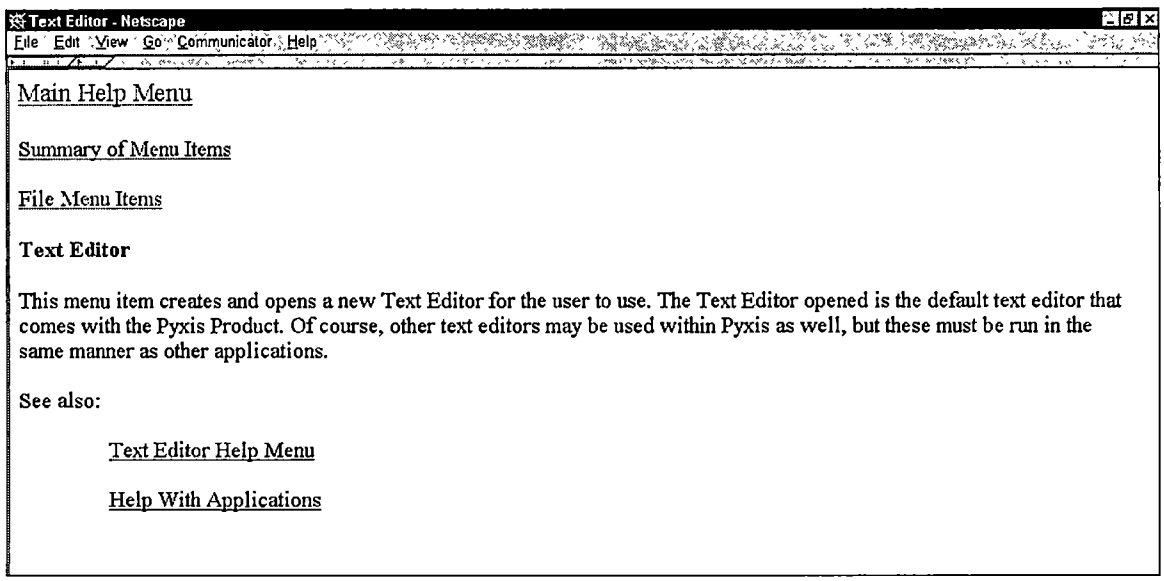
Session (File - New - Session)

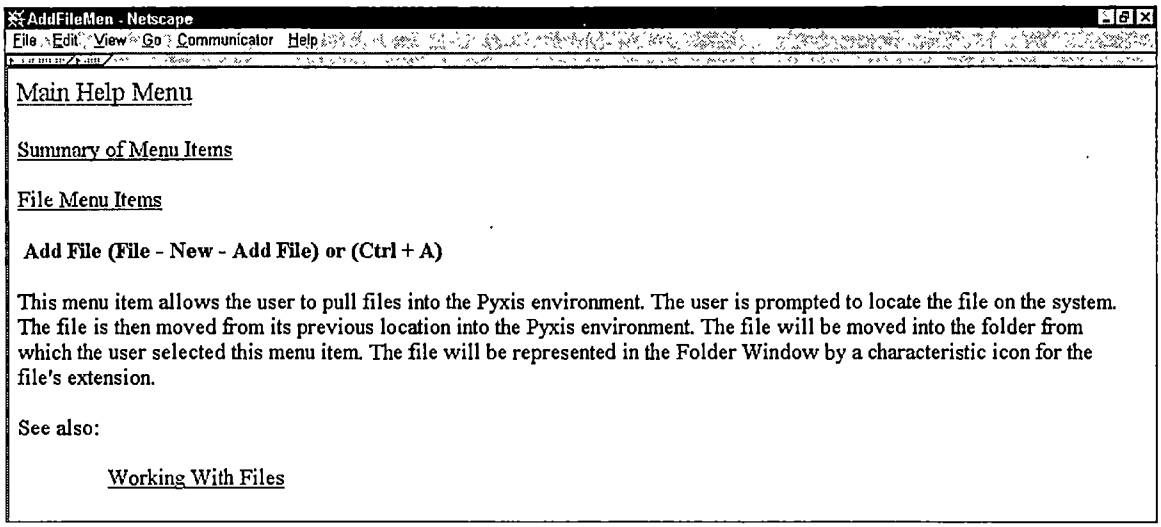
This menu item creates a new Session. The new empty Session Window will open and appear on the screen. The default name for this session is "New Session". The user may then click on the open Session Window to begin working in the new Session Window.

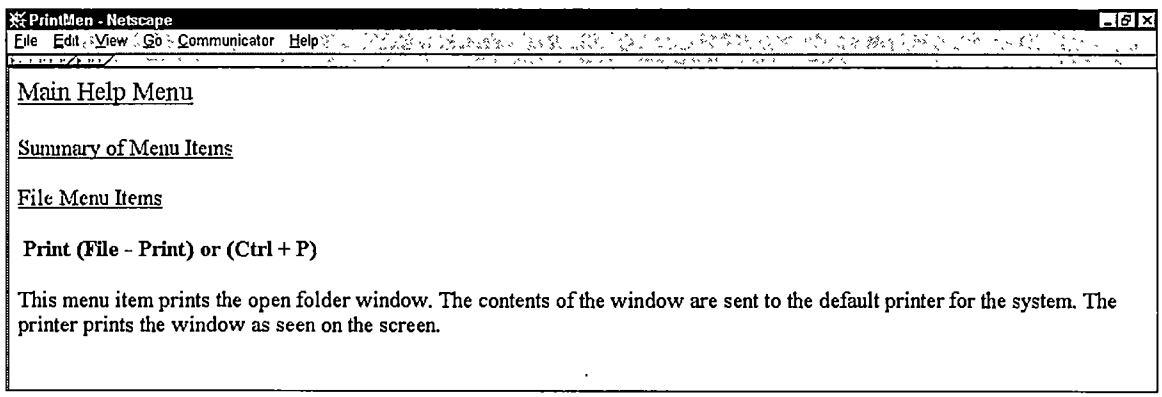
See also:

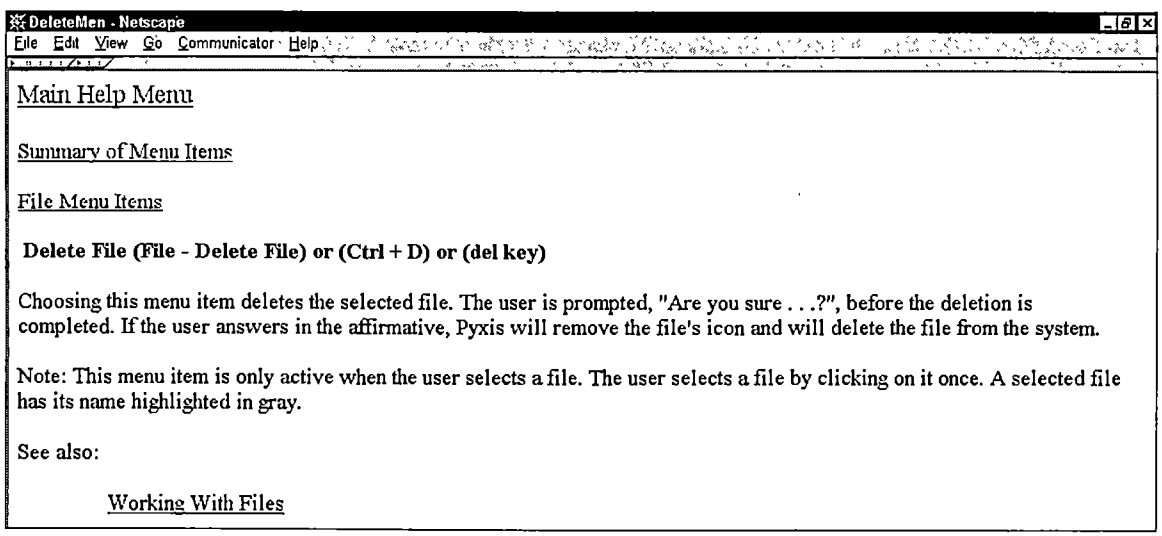
Session Help Menu

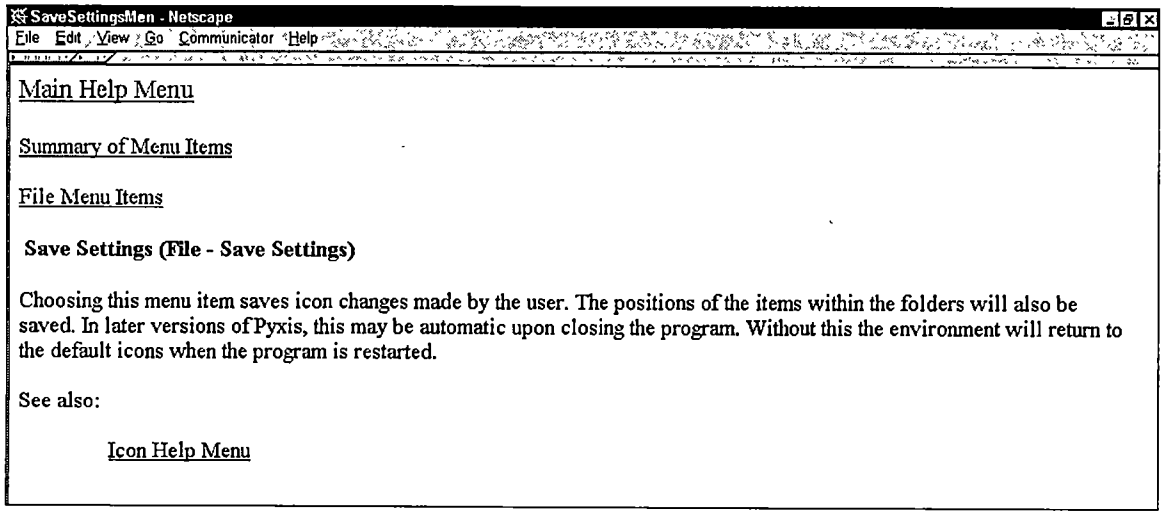

Folder - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Folder (File - New - Folder) or (Ctrl + N)
This menu item creates a new folder. A closed folder is created and the user is prompted to name the folder. If the user does not specify a new name the first folder is named "New Folder 1". Subsequent unnamed folders will be assigned the names "New Folder 2", "New Folder 3", etc.
See also:
Working With Folders

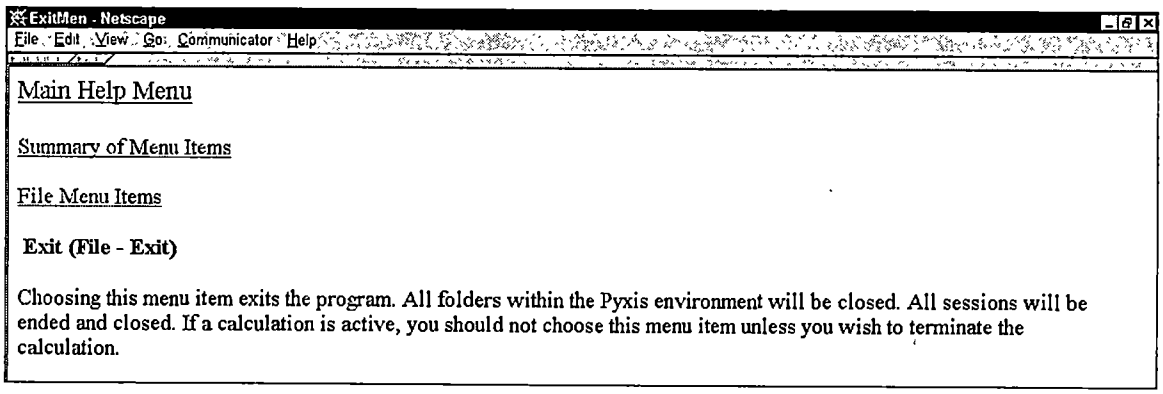

Text Editor - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Text Editor
This menu item creates and opens a new Text Editor for the user to use. The Text Editor opened is the default text editor that comes with the Pyxis Product. Of course, other text editors may be used within Pyxis as well, but these must be run in the same manner as other applications.
See also:
Text Editor Help Menu
Help With Applications

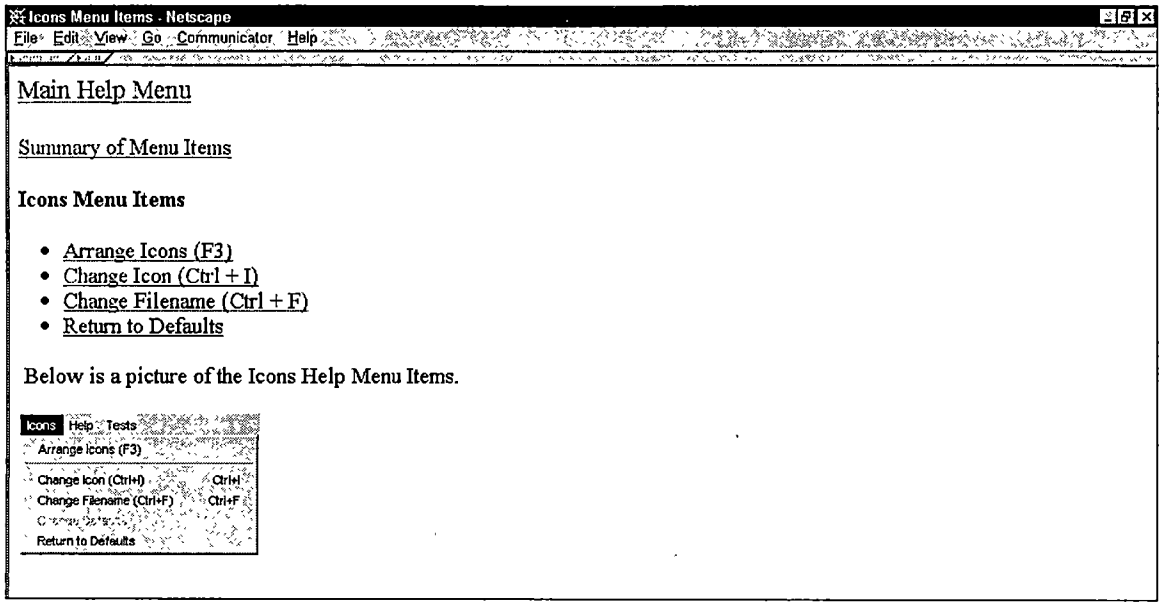

AddFileMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Add File (File - New - Add File) or (Ctrl + A)
This menu item allows the user to pull files into the Pyxis environment. The user is prompted to locate the file on the system. The file is then moved from its previous location into the Pyxis environment. The file will be moved into the folder from which the user selected this menu item. The file will be represented in the Folder Window by a characteristic icon for the file's extension.
See also:
Working With Files

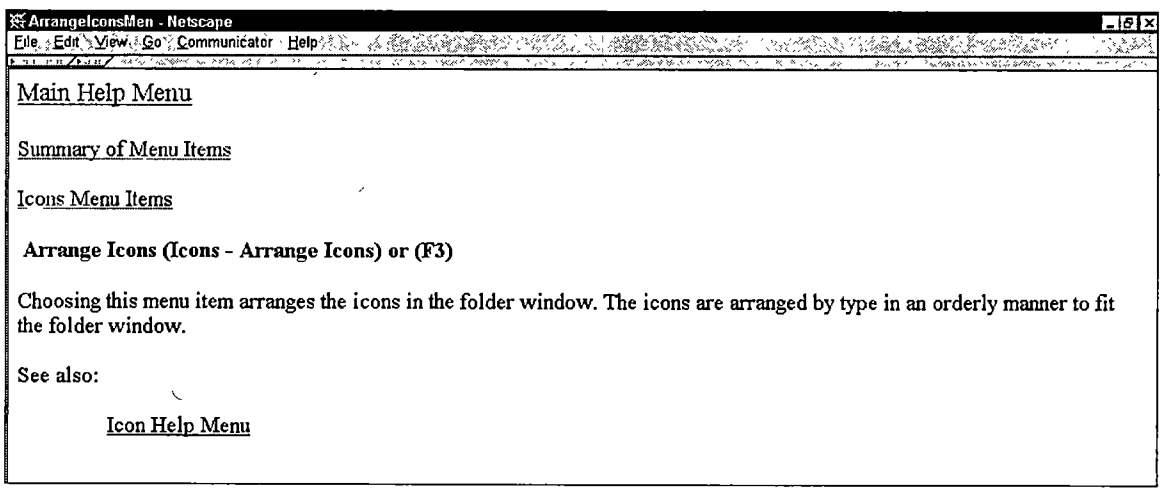

PrintMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Print (File - Print) or (Ctrl + P)
This menu item prints the open folder window. The contents of the window are sent to the default printer for the system. The printer prints the window as seen on the screen.

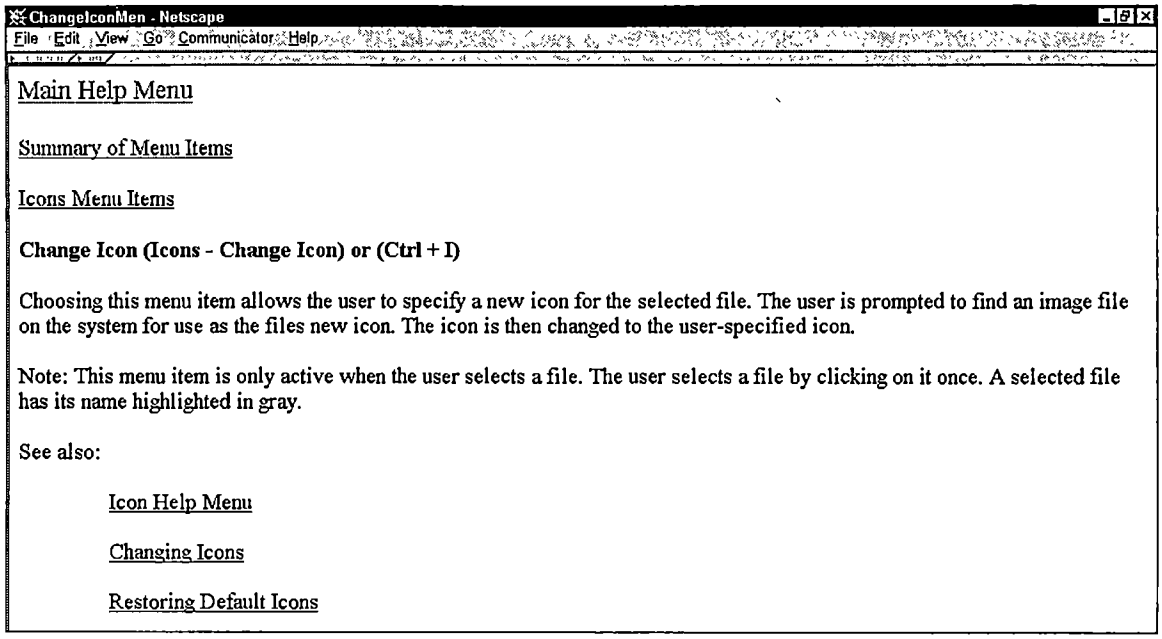

DeleteMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Delete File (File - Delete File) or (Ctrl + D) or (del key)
Choosing this menu item deletes the selected file. The user is prompted, "Are you sure . . .?", before the deletion is completed. If the user answers in the affirmative, Pyxis will remove the file's icon and will delete the file from the system.
Note: This menu item is only active when the user selects a file. The user selects a file by clicking on it once. A selected file has its name highlighted in gray.
See also:
Working With Files

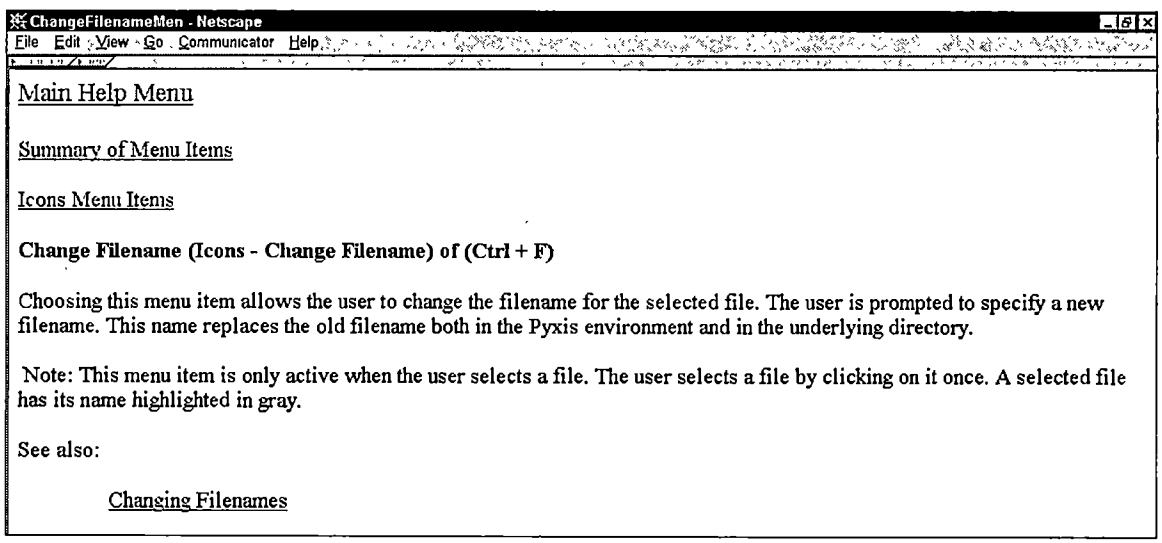

SaveSettingsMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Save Settings (File - Save Settings)
Choosing this menu item saves icon changes made by the user. The positions of the items within the folders will also be saved. In later versions of Pyxis, this may be automatic upon closing the program. Without this the environment will return to the default icons when the program is restarted.
See also:
Icon Help Menu

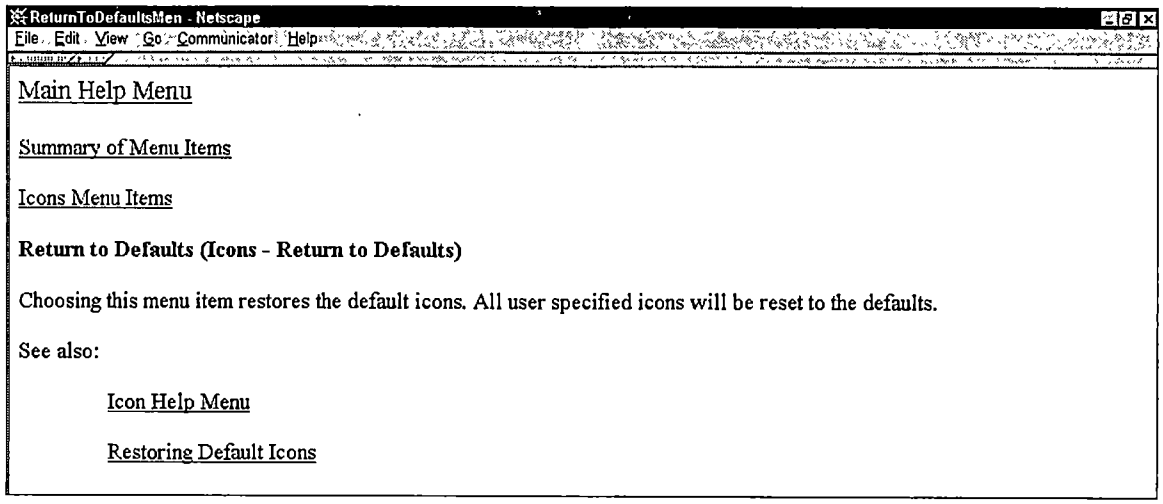

ExitMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
File Menu Items
Exit (File - Exit)
Choosing this menu item exits the program. All folders within the Pyxis environment will be closed. All sessions will be ended and closed. If a calculation is active, you should not choose this menu item unless you wish to terminate the calculation.

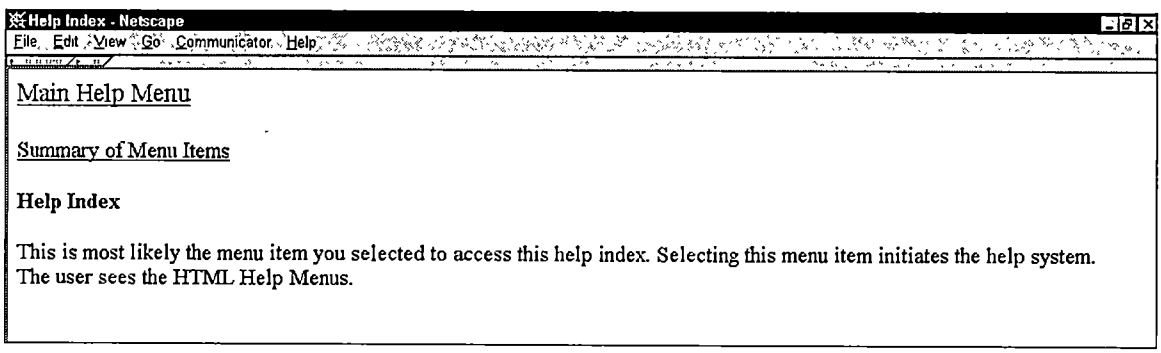

Icons Menu Items - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
Icons Menu Items
Arrange Icons (F3)
Change Icon (Ctrl + I)
Change Filename (Ctrl + F)
Return to Defaults
Below is a picture of the Icons Help Menu Items.
Icons Help Tests
Arrange Icons (F3)
Change Icon (Ctrl+I) Ctrl+I
Change Filename (Ctrl+F) Ctrl+F
Return to Defaults

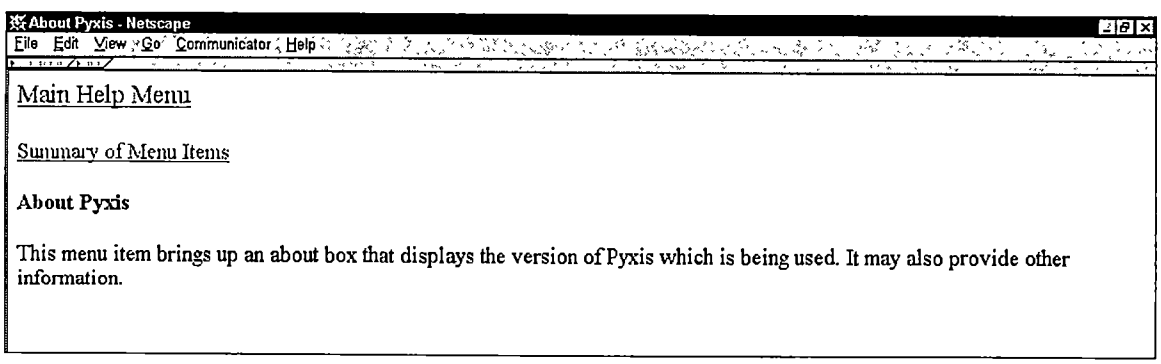

ArrangeIconsMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
Icons Menu Items
Arrange Icons (Icons - Arrange Icons) or (F3)
Choosing this menu item arranges the icons in the folder window. The icons are arranged by type in an orderly manner to fit the folder window.
See also:
Icon Help Menu


ChangeIconMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
Icons Menu Items
Change Icon (Icons - Change Icon) or (Ctrl + I)
Choosing this menu item allows the user to specify a new icon for the selected file. The user is prompted to find an image file on the system for use as the files new icon. The icon is then changed to the user-specified icon.
Note: This menu item is only active when the user selects a file. The user selects a file by clicking on it once. A selected file has its name highlighted in gray.
See also:
Icon Help Menu
Changing Icons
Restoring Default Icons


ChangeFilenameMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
Icons Menu Items
Change Filename (Icons - Change Filename) of (Ctrl + F)
Choosing this menu item allows the user to change the filename for the selected file. The user is prompted to specify a new filename. This name replaces the old filename both in the Pyxis environment and in the underlying directory.
Note: This menu item is only active when the user selects a file. The user selects a file by clicking on it once. A selected file has its name highlighted in gray.
See also:
Changing Filenames


ReturnToDefaultsMen - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
Icons Menu Items
Return to Defaults (Icons - Return to Defaults)
Choosing this menu item restores the default icons. All user specified icons will be reset to the defaults.
See also:
Icon Help Menu
Restoring Default Icons


Help Index - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
Help Index
This is most likely the menu item you selected to access this help index. Selecting this menu item initiates the help system. The user sees the HTML Help Menus.


About Pyxis - Netscape
File Edit View Go Communicator Help
Main Help Menu
Summary of Menu Items
About Pyxis
This menu item brings up an about box that displays the version of Pyxis which is being used. It may also provide other information.

Session Help Menu - Netscape

File Edit View Go Communicator Help

Main Help Menu

Session Help Menu

- What is a Session?
- Creating a Session
- Running a Calculation
- Link Help Menu
- Applications Help Menu
- Importing Files/Applications into the Session Environment
- Using the Group Box
- Working With Super Applications
- Archiving
- Creating and Using Dummy Files
- Frequently Asked Questions about the Session Environment

What is a Session - Netscape

File Edit View Go Communicator Help

Main Help Menu

Session Help Menu

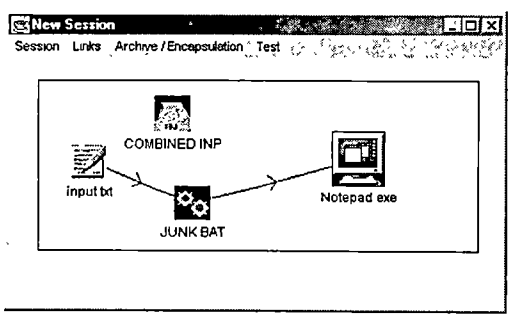
What is a Session?

Sessions are special windows that do not represent underlying directories, but instead serve to display relationships among items located in specific directories. The session environment is the one in which linked engineering calculations are performed. A user is "in the session environment" when he or she has a session window open and activated.

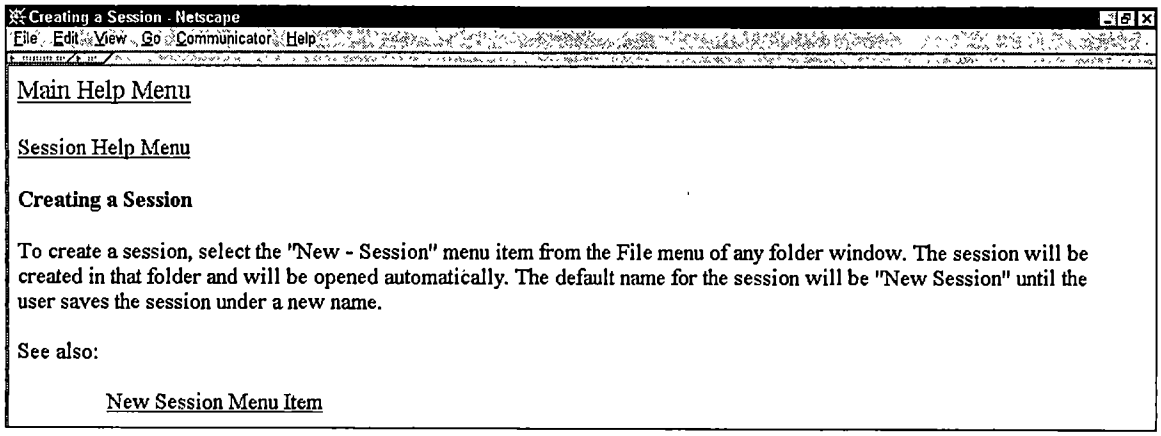
Below is an example of the Session Icon. To open a session, simply double-click on one of these session icons.

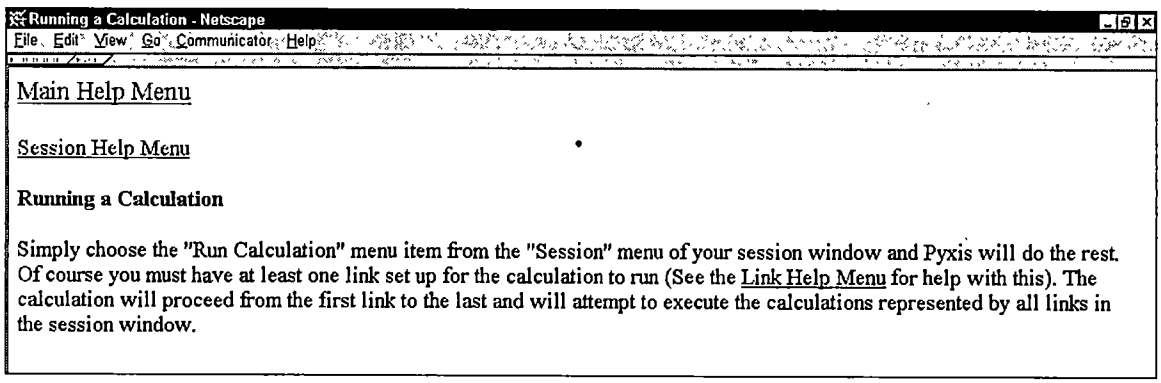


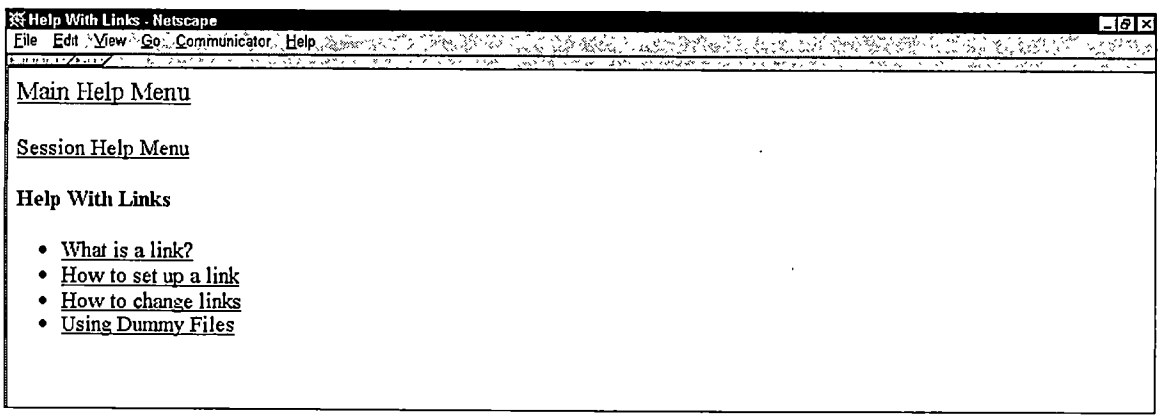
Below is an example of a Session Window.



Document Done


Creating a Session - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Creating a Session
To create a session, select the "New - Session" menu item from the File menu of any folder window. The session will be created in that folder and will be opened automatically. The default name for the session will be "New Session" until the user saves the session under a new name.
See also:
New Session Menu Item


Running a Calculation - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Running a Calculation
Simply choose the "Run Calculation" menu item from the "Session" menu of your session window and Pyxis will do the rest. Of course you must have at least one link set up for the calculation to run (See the Link Help Menu for help with this). The calculation will proceed from the first link to the last and will attempt to execute the calculations represented by all links in the session window.


Help With Links - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Help With Links
What is a link?
How to set up a link
How to change links
Using Dummy Files

What is a link - Netscape

File Edit View Go Communicator Help

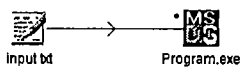
Main Help Menu

Session Help Menu

Link Help Menu

What is a link?

A link is simply a visual connection between an application and one of its input or output files. In the session window a link appears as a blue line connecting two icons. In the middle of the blue line is a blue arrow. The arrow indicates whether the file is an input for the application to use or an output coming from the application. These visual links are used to set up the command line when running a calculation. The order in which the links are set up will be the order they appear on the command line when the calculation is run. An example of a well form link appears below.



How to set up a link - Netscape

File Edit View Go Communicator Help

Main Help Menu

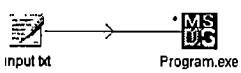
Session Help Menu

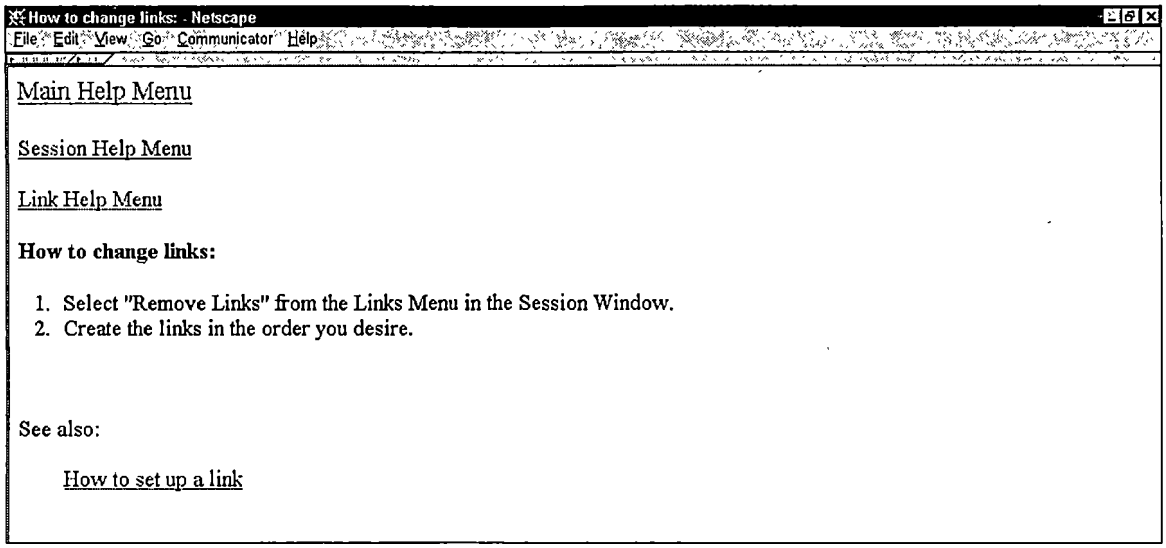
Link Help Menu

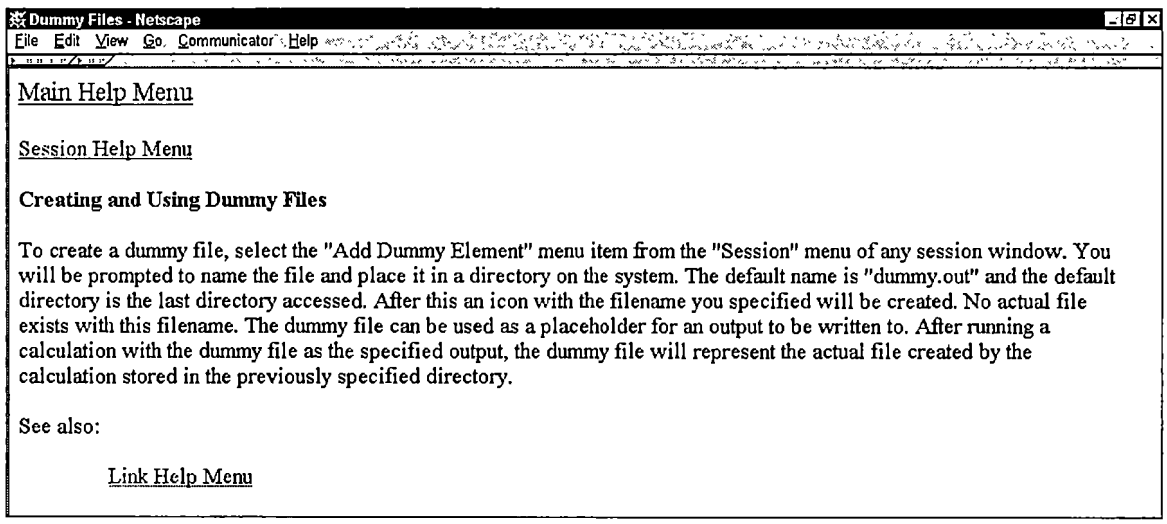
How to set up a link:

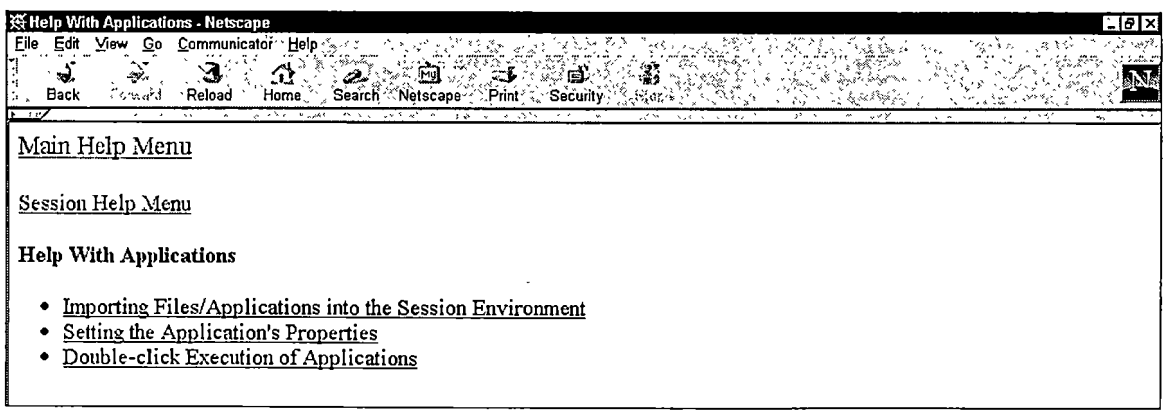
1. Select "Add Link" from the Links menu in the Session Window.
2. Move the crosshair cursor to the item in the Session Window you wish to link from.
3. Press and hold the mouse button down. Drag the crosshair cursor to the item you wish to link to.
4. Release the mouse button.

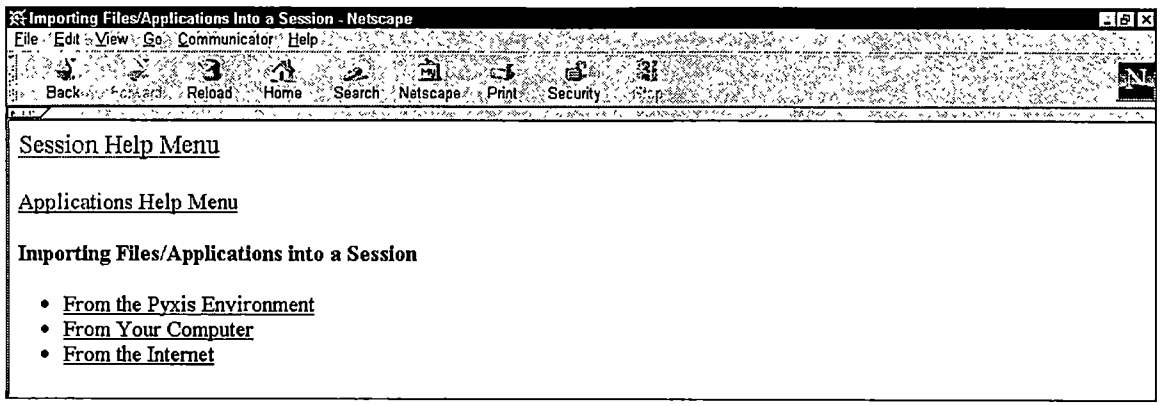
If you were successful a tone will sound and the link will be visible in the window. An example of a well-formed link appears below.

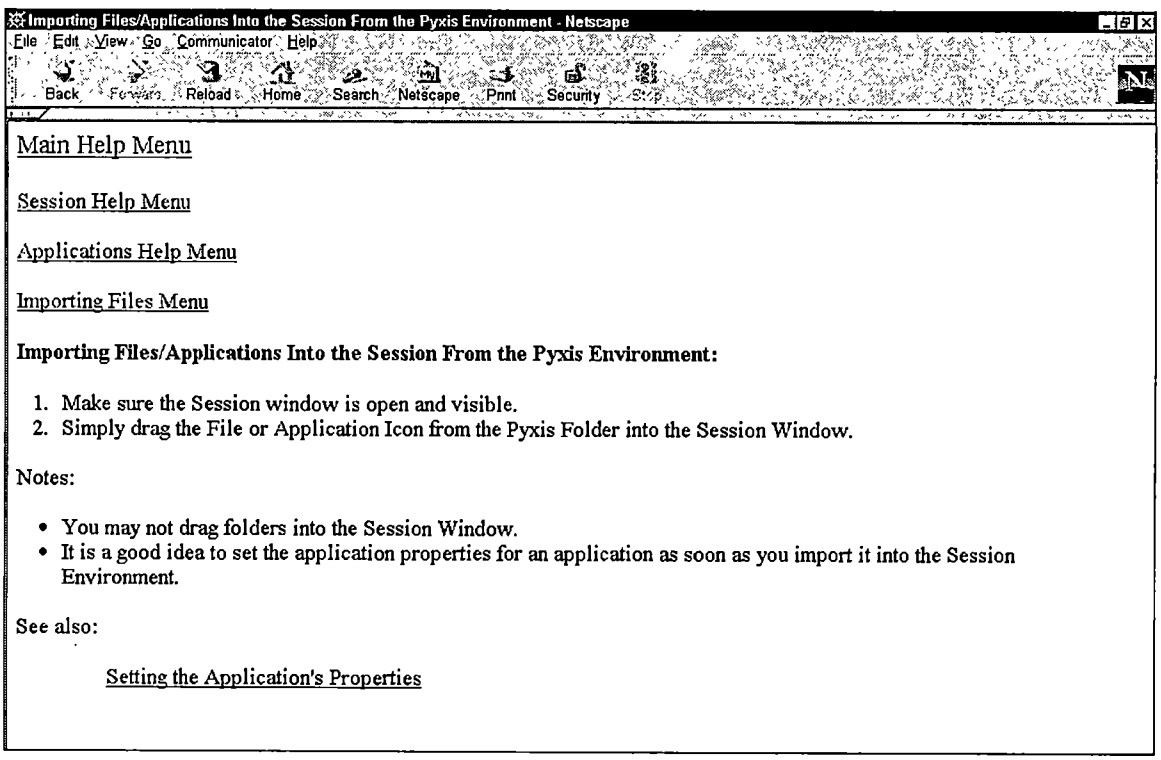


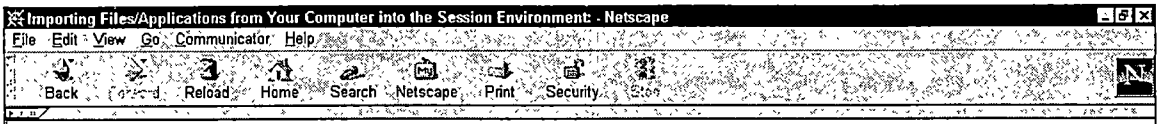

How to change links: - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Link Help Menu
How to change links:
1. Select "Remove Links" from the Links Menu in the Session Window.
2. Create the links in the order you desire.
See also:
How to set up a link


Dummy Files - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Creating and Using Dummy Files
To create a dummy file, select the "Add Dummy Element" menu item from the "Session" menu of any session window. You will be prompted to name the file and place it in a directory on the system. The default name is "dummy.out" and the default directory is the last directory accessed. After this an icon with the filename you specified will be created. No actual file exists with this filename. The dummy file can be used as a placeholder for an output to be written to. After running a calculation with the dummy file as the specified output, the dummy file will represent the actual file created by the calculation stored in the previously specified directory.
See also:
Link Help Menu


Help With Applications - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security
Main Help Menu
Session Help Menu
Help With Applications
Importing Files/Applications into the Session Environment
Setting the Application's Properties
Double-click Execution of Applications


Importing Files/Applications Into a Session - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security
Session Help Menu
Applications Help Menu
Importing Files/Applications into a Session
From the Pyxis Environment
From Your Computer
From the Internet


Importing Files/Applications Into the Session From the Pyxis Environment - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security
Main Help Menu
Session Help Menu
Applications Help Menu
Importing Files Menu
Importing Files/Applications Into the Session From the Pyxis Environment:
1. Make sure the Session window is open and visible.
2. Simply drag the File or Application Icon from the Pyxis Folder into the Session Window.
Notes:
You may not drag folders into the Session Window.
It is a good idea to set the application properties for an application as soon as you import it into the Session Environment.
See also:
Setting the Application's Properties



Main Help Menu

Session Help Menu

Applications Help Menu

Importing Files Help Menu

Importing Files/Applications from Your Computer into the Session Environment:

1. Select "Add Session Element" from the Session Menu in the Session Window.
2. Select the File you wish to import and click on the "Open" button.

Notes:

- You may not import an item by selecting a shortcut to that item (for Windows Users). You must go to the directory where the file or application resides.
- It is a good practice to set the application's properties immediately after importing it into the Session Environment.

See also:

Setting the Application's Properties



Main Help Menu

Session Help Menu

Applications Help Menu

Setting the Application's Properties

To set the applications properties, first click once on the application in the session window to select it. The filename below the icon should be highlighted in gray.

Next, choose the "Set Application Properties" menu item from the "Session" menu in the session window.

You will see and interface which prompts you for information about the number and type of inputs and outputs. It is important to list only the inputs and outputs which must be specified in order for the application to run. In other words, outputs that would not be specified on the command line, but would be generated by default, should not be specified in this dialog. Forcing the application to look for inputs or outputs that it does not require may cause the calculation to fail.

Note:

This process is key to successful use of the session environment and should be done by someone with experience with the product.

Double-click Execution - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security

Session Help Menu

Applications Help Menu

Double-click Execution of Executable Files

Files with the extensions ".exe" or ".bat" may be executed by double-clicking on them if the file with that extension is an executable. Double-click execution is available in both the folder windows and session windows.

Note: If this method for running an application is unsuccessful attempt to run the application after linking it to something (perhaps simply a dummy file) in a session window using the "Run Calculation" menu item. See Running a Calculation.

See also:

Working With Files

Using the Group Box - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security

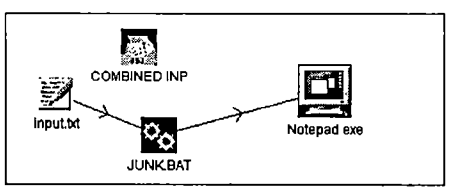
Main Help Menu

Session Help Menu

Using the Group Box

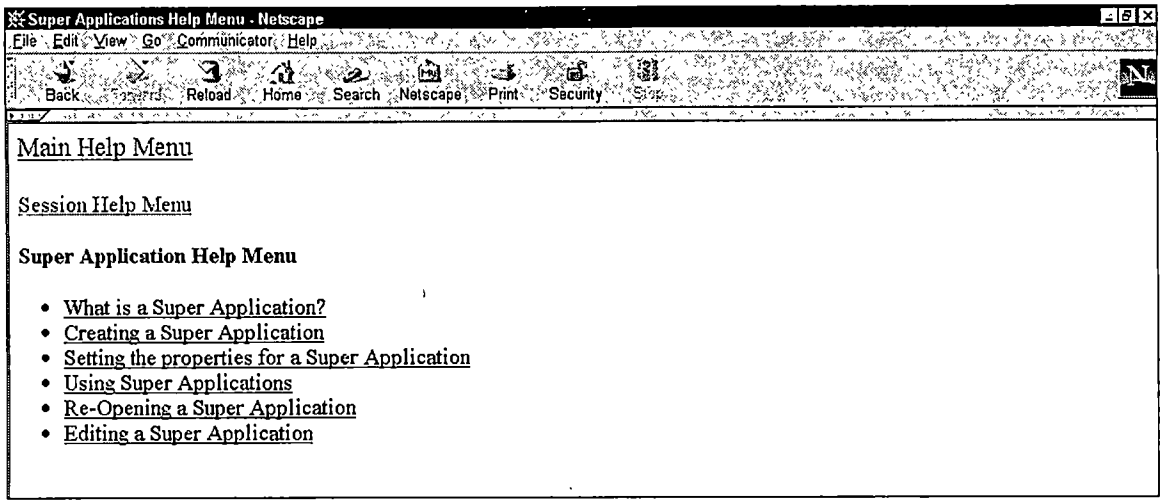
The group box has two uses. It may be used in archiving, and it may be used for creating Super Applications. To create a group box choose "Create Group Box" from the "Archive/Encapsulation" menu in the session window. A Crosshair cursor will appear. Put the cursor where you want the upper left corner of the group box. Click and hold the mouse button down. Drag the cursor to the point for the lower right corner of the group box and release the mouse button. You may only have one group box in use in a given session window. To alter the group box simply repeat this process. The newly created group box will replace the old one, which will be automatically removed.

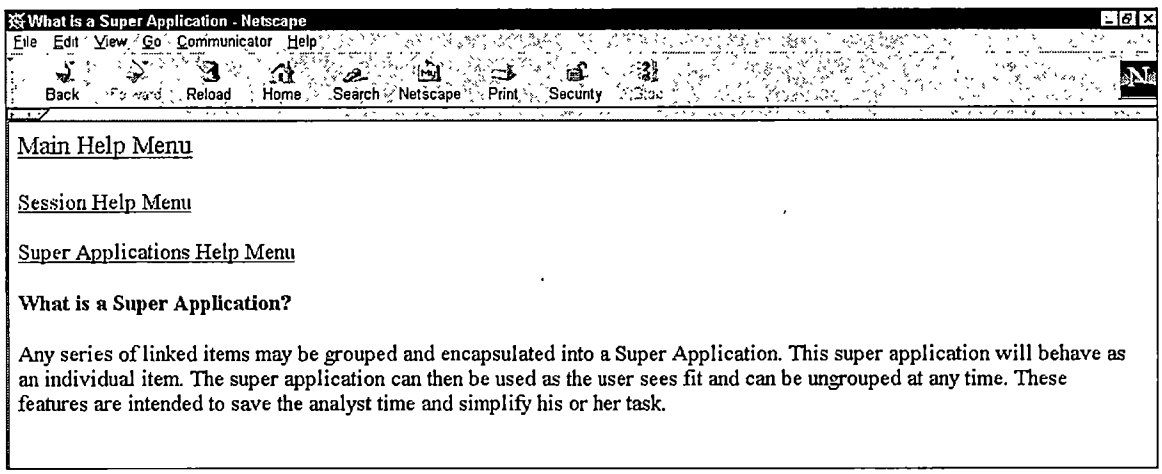
A well-formed group box is shown below.

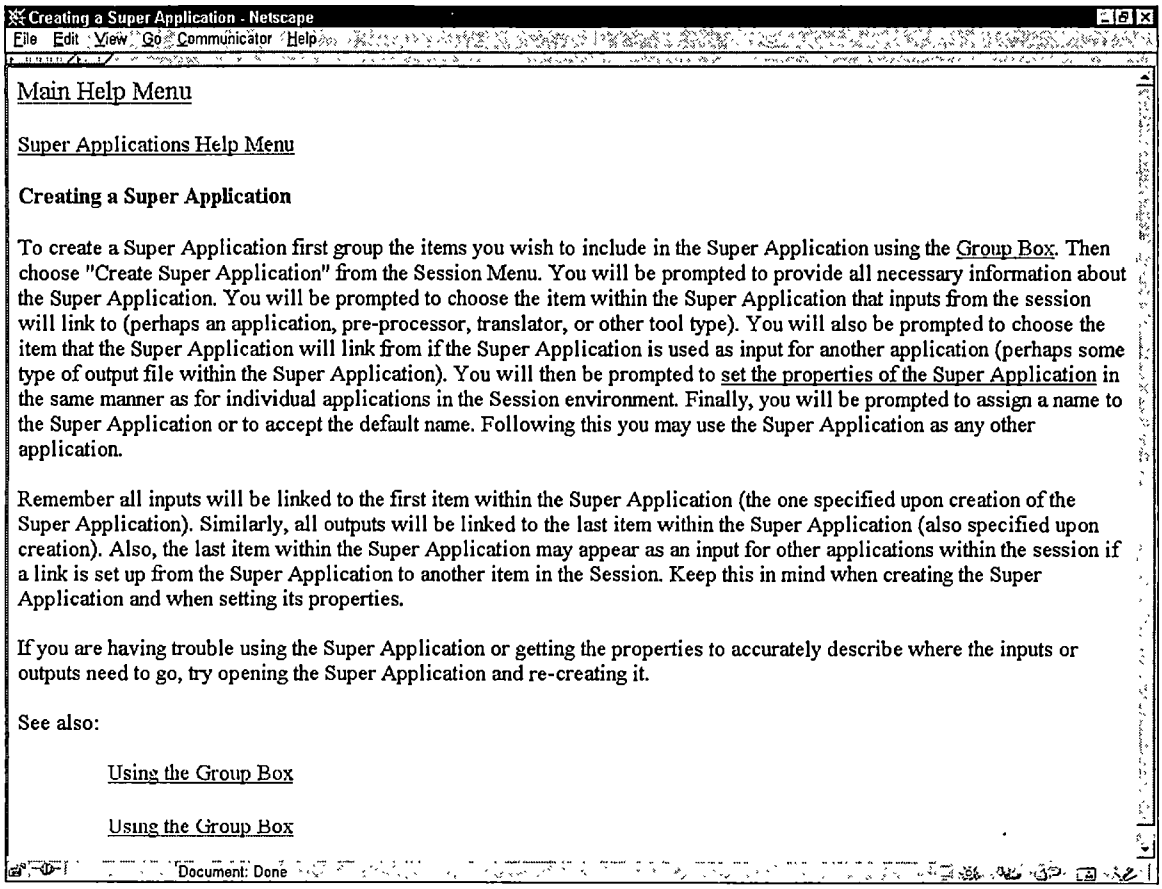


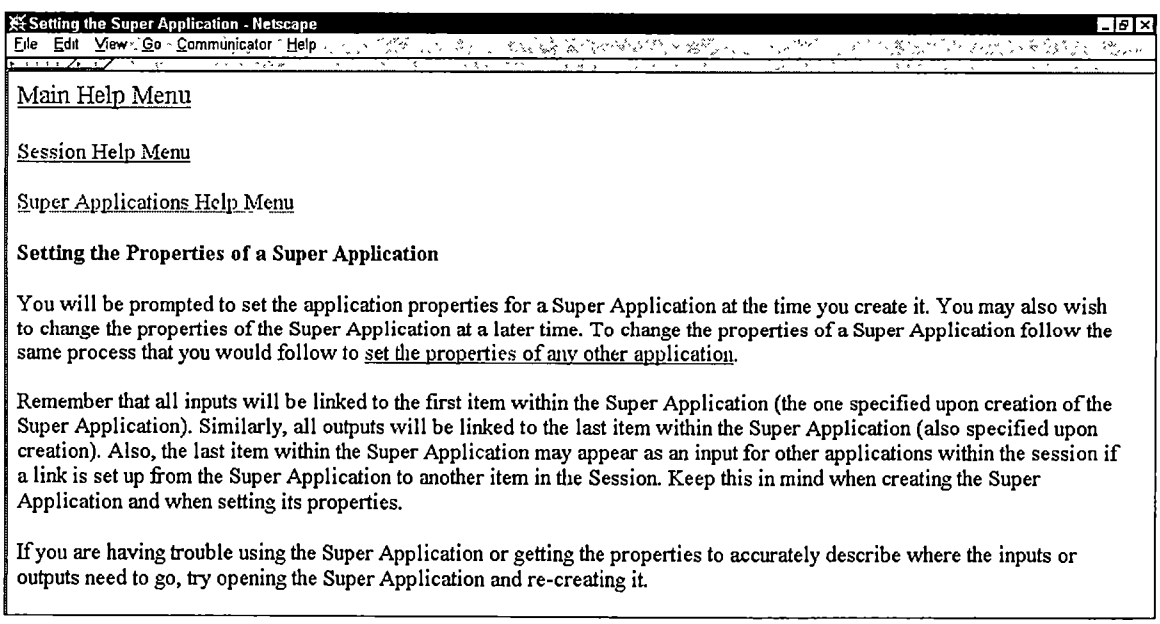
To remove the group box select the "Remove Group Box" menu item from the "Archive/Encapsulation" menu in the session window.

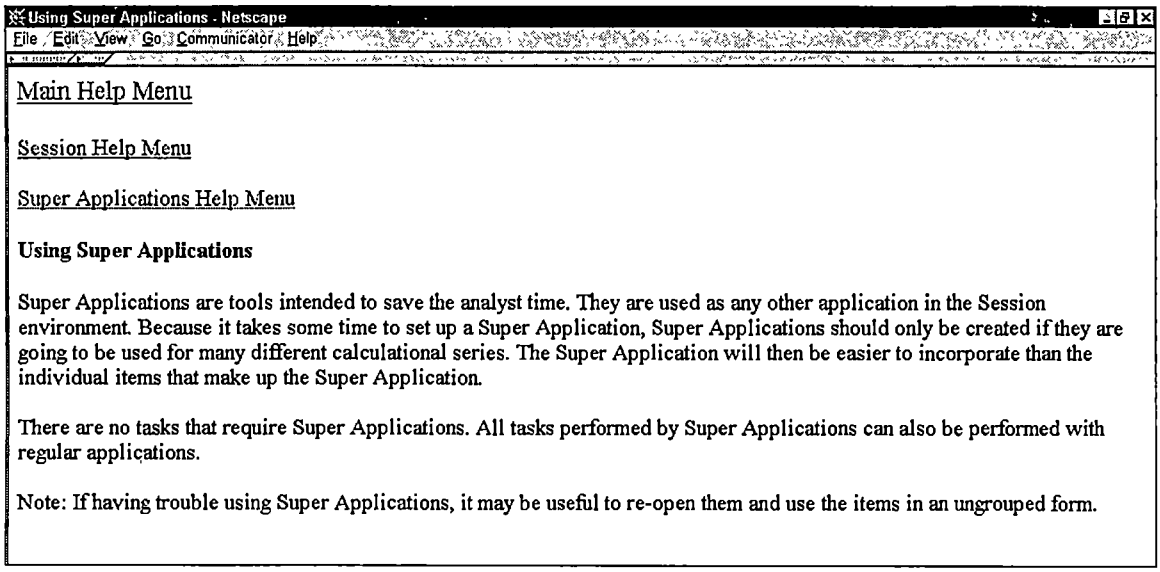
Document: Done

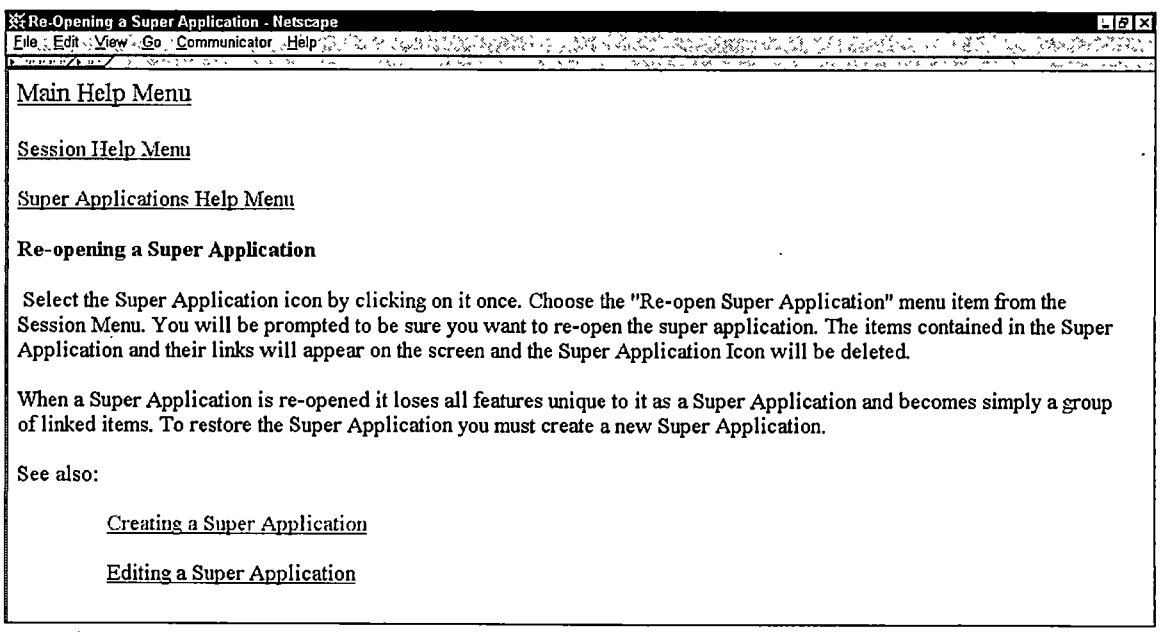

Super Applications Help Menu - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security
Main Help Menu
Session Help Menu
Super Application Help Menu
What is a Super Application?
Creating a Super Application
Setting the properties for a Super Application
Using Super Applications
Re-Opening a Super Application
Editing a Super Application

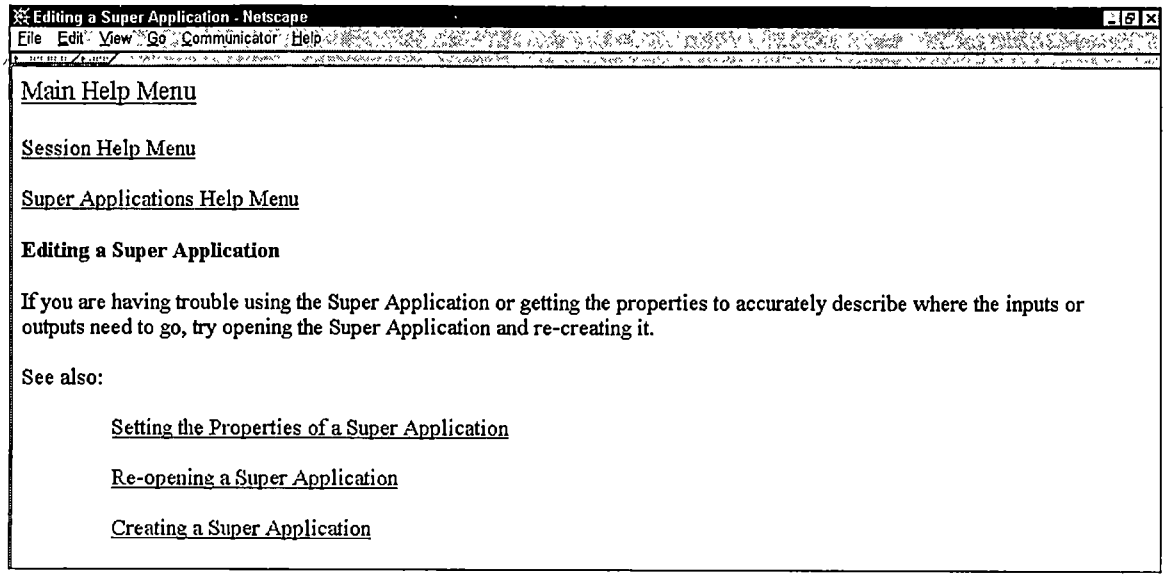

What is a Super Application - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security
Main Help Menu
Session Help Menu
Super Applications Help Menu
What is a Super Application?
Any series of linked items may be grouped and encapsulated into a Super Application. This super application will behave as an individual item. The super application can then be used as the user sees fit and can be ungrouped at any time. These features are intended to save the analyst time and simplify his or her task.

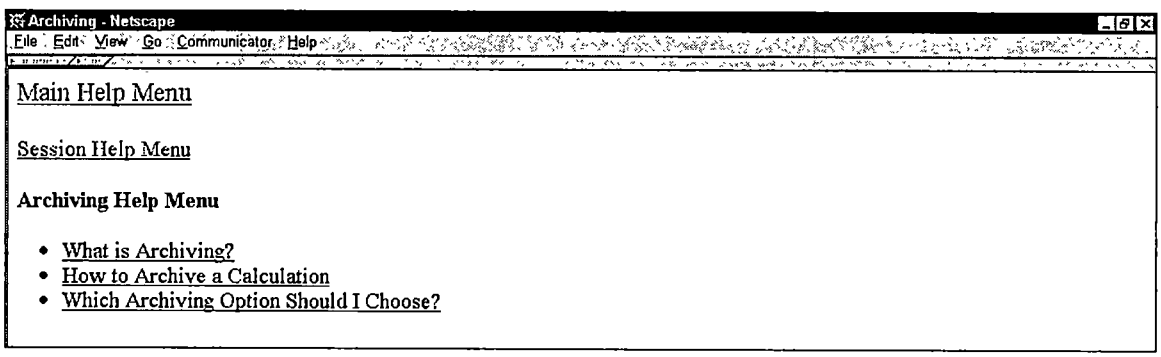

Creating a Super Application - Netscape
File Edit View Go Communicator Help
Main Help Menu
Super Applications Help Menu
Creating a Super Application
To create a Super Application first group the items you wish to include in the Super Application using the Group Box. Then choose "Create Super Application" from the Session Menu. You will be prompted to provide all necessary information about the Super Application. You will be prompted to choose the item within the Super Application that inputs from the session will link to (perhaps an application, pre-processor, translator, or other tool type). You will also be prompted to choose the item that the Super Application will link from if the Super Application is used as input for another application (perhaps some type of output file within the Super Application). You will then be prompted to set the properties of the Super Application in the same manner as for individual applications in the Session environment. Finally, you will be prompted to assign a name to the Super Application or to accept the default name. Following this you may use the Super Application as any other application.
Remember all inputs will be linked to the first item within the Super Application (the one specified upon creation of the Super Application). Similarly, all outputs will be linked to the last item within the Super Application (also specified upon creation). Also, the last item within the Super Application may appear as an input for other applications within the session if a link is set up from the Super Application to another item in the Session. Keep this in mind when creating the Super Application and when setting its properties.
If you are having trouble using the Super Application or getting the properties to accurately describe where the inputs or outputs need to go, try opening the Super Application and re-creating it.
See also:
Using the Group Box
Using the Group Box
Document: Done

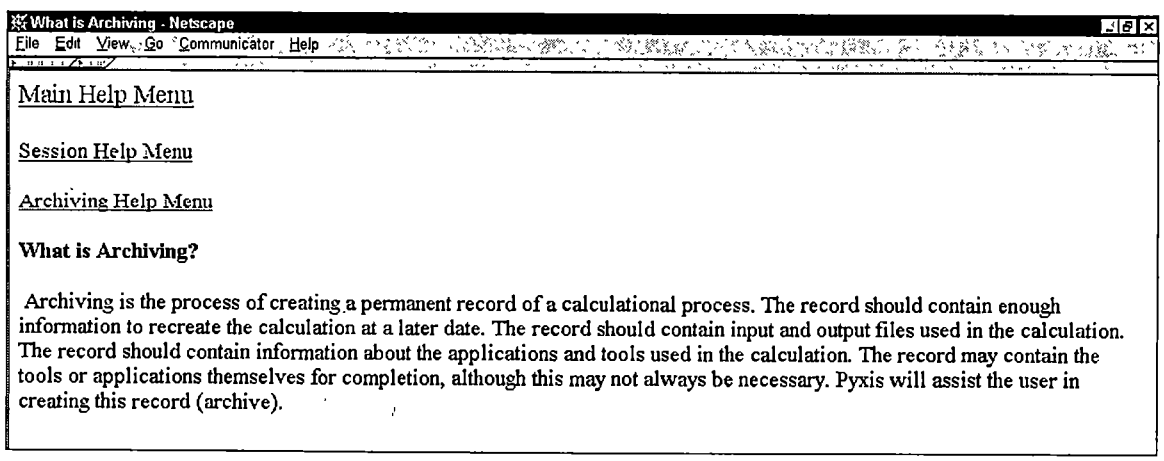

Setting the Super Application - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Super Applications Help Menu
Setting the Properties of a Super Application
You will be prompted to set the application properties for a Super Application at the time you create it. You may also wish to change the properties of the Super Application at a later time. To change the properties of a Super Application follow the same process that you would follow to set the properties of any other application.
Remember that all inputs will be linked to the first item within the Super Application (the one specified upon creation of the Super Application). Similarly, all outputs will be linked to the last item within the Super Application (also specified upon creation). Also, the last item within the Super Application may appear as an input for other applications within the session if a link is set up from the Super Application to another item in the Session. Keep this in mind when creating the Super Application and when setting its properties.
If you are having trouble using the Super Application or getting the properties to accurately describe where the inputs or outputs need to go, try opening the Super Application and re-creating it.

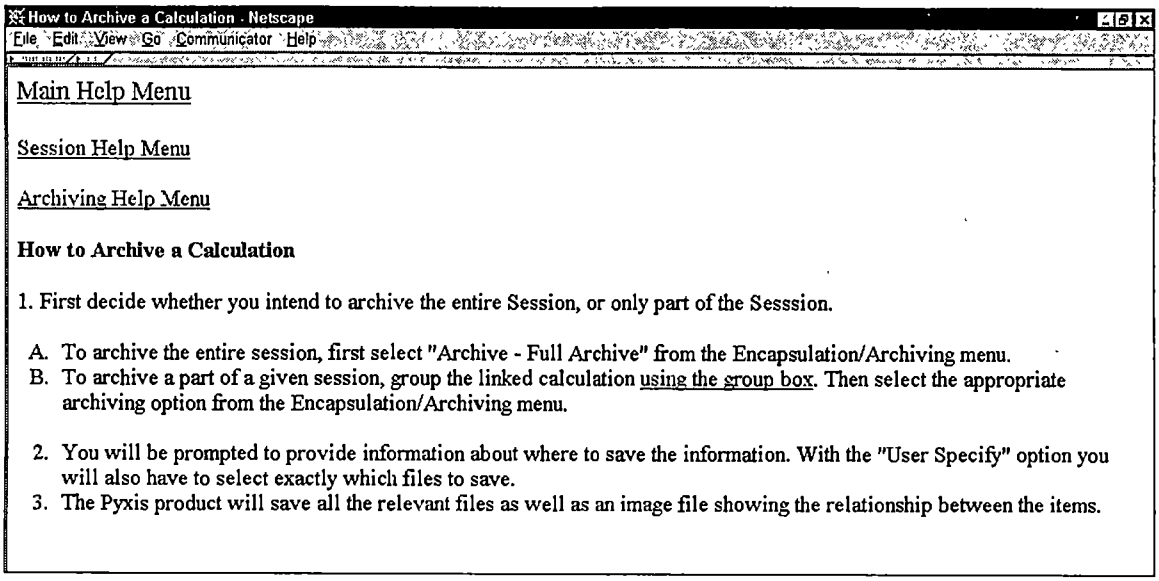

Using Super Applications - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Super Applications Help Menu
Using Super Applications
Super Applications are tools intended to save the analyst time. They are used as any other application in the Session environment. Because it takes some time to set up a Super Application, Super Applications should only be created if they are going to be used for many different calculational series. The Super Application will then be easier to incorporate than the individual items that make up the Super Application.
There are no tasks that require Super Applications. All tasks performed by Super Applications can also be performed with regular applications.
Note: If having trouble using Super Applications, it may be useful to re-open them and use the items in an ungrouped form.

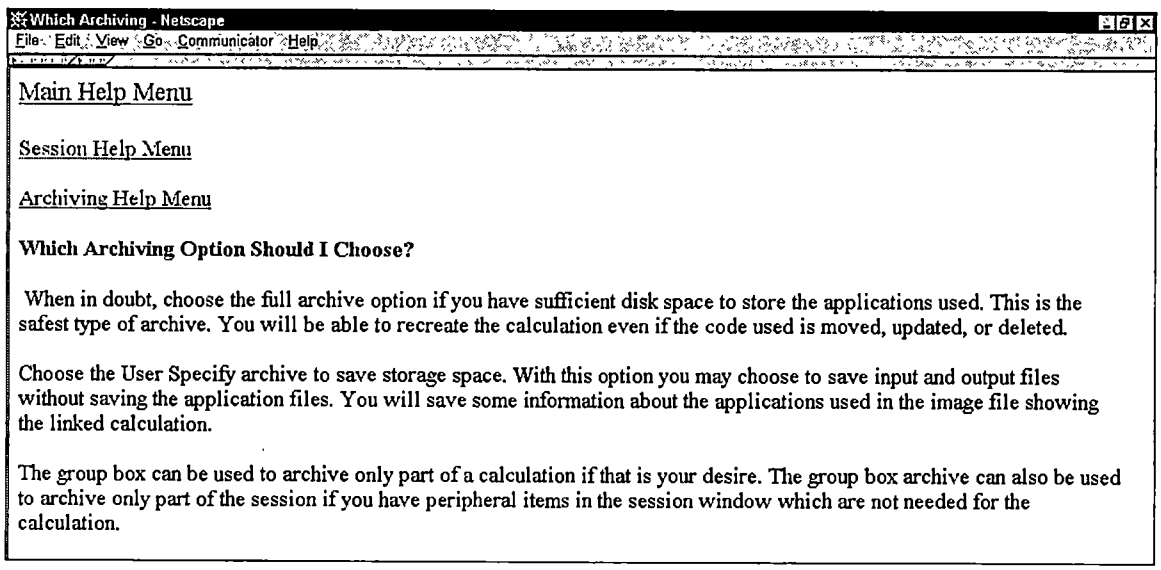

Re-Opening a Super Application - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Super Applications Help Menu
Re-opening a Super Application
Select the Super Application icon by clicking on it once. Choose the "Re-open Super Application" menu item from the Session Menu. You will be prompted to be sure you want to re-open the super application. The items contained in the Super Application and their links will appear on the screen and the Super Application Icon will be deleted.
When a Super Application is re-opened it loses all features unique to it as a Super Application and becomes simply a group of linked items. To restore the Super Application you must create a new Super Application.
See also:
Creating a Super Application
Editing a Super Application

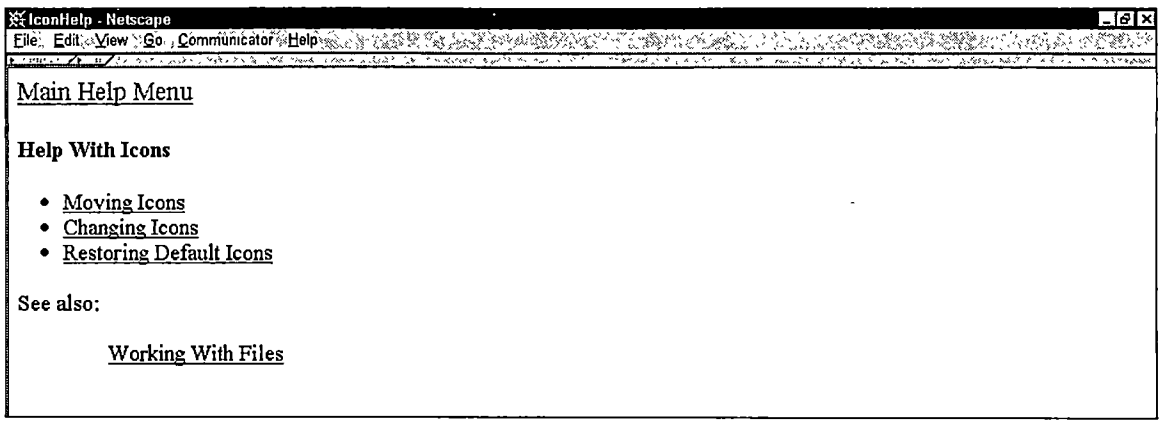

Editing a Super Application - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Super Applications Help Menu
Editing a Super Application
If you are having trouble using the Super Application or getting the properties to accurately describe where the inputs or outputs need to go, try opening the Super Application and re-creating it.
See also:
Setting the Properties of a Super Application
Re-opening a Super Application
Creating a Super Application

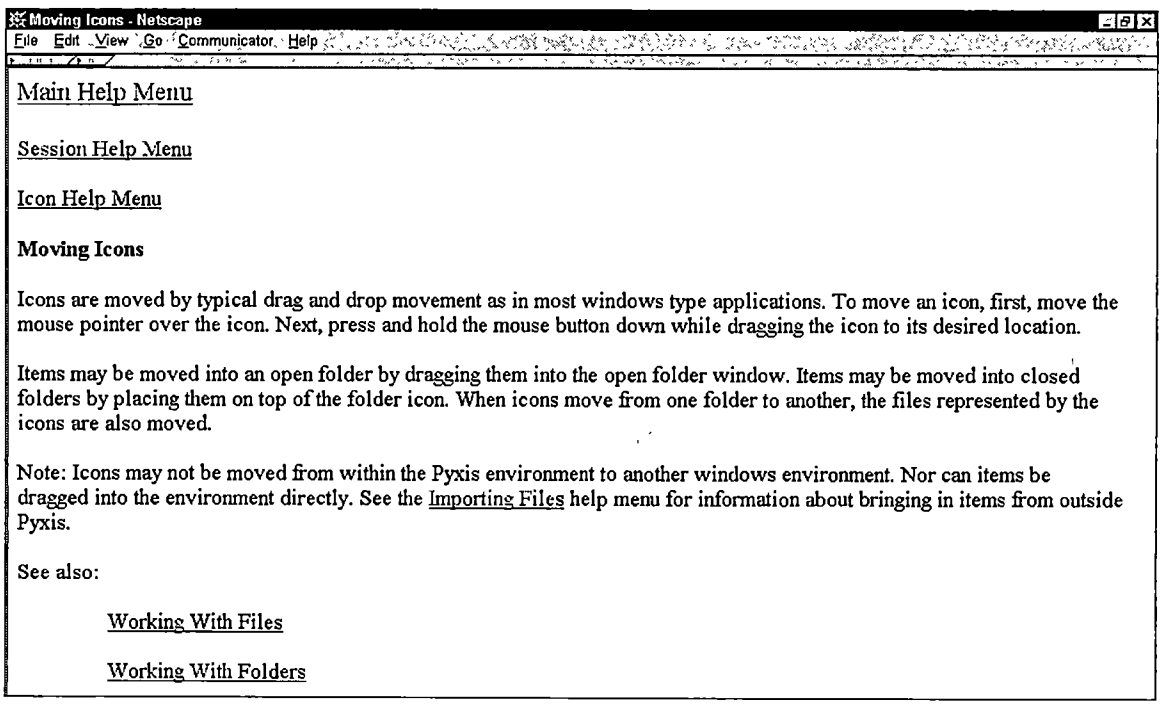

Archiving - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Archiving Help Menu
What is Archiving?
How to Archive a Calculation
Which Archiving Option Should I Choose?


What is Archiving - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Archiving Help Menu
What is Archiving?
Archiving is the process of creating a permanent record of a calculational process. The record should contain enough information to recreate the calculation at a later date. The record should contain input and output files used in the calculation. The record should contain information about the applications and tools used in the calculation. The record may contain the tools or applications themselves for completion, although this may not always be necessary. Pyxis will assist the user in creating this record (archive).


How to Archive a Calculation - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Archiving Help Menu
How to Archive a Calculation
1. First decide whether you intend to archive the entire Session, or only part of the Sesssion.
A. To archive the entire session, first select "Archive - Full Archive" from the Encapsulation/Archiving menu.
B. To archive a part of a given session, group the linked calculation using the group box. Then select the appropriate archiving option from the Encapsulation/Archiving menu.
2. You will be prompted to provide information about where to save the information. With the "User Specify" option you will also have to select exactly which files to save.
3. The Pyxis product will save all the relevant files as well as an image file showing the relationship between the items.


Which Archiving - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Archiving Help Menu
Which Archiving Option Should I Choose?
When in doubt, choose the full archive option if you have sufficient disk space to store the applications used. This is the safest type of archive. You will be able to recreate the calculation even if the code used is moved, updated, or deleted.
Choose the User Specify archive to save storage space. With this option you may choose to save input and output files without saving the application files. You will save some information about the applications used in the image file showing the linked calculation.
The group box can be used to archive only part of a calculation if that is your desire. The group box archive can also be used to archive only part of the session if you have peripheral items in the session window which are not needed for the calculation.


IconHelp - Netscape
File Edit View Go Communicator Help
Main Help Menu
Help With Icons
Moving Icons
Changing Icons
Restoring Default Icons
See also:
Working With Files


Moving Icons - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Icon Help Menu
Moving Icons
Icons are moved by typical drag and drop movement as in most windows type applications. To move an icon, first, move the mouse pointer over the icon. Next, press and hold the mouse button down while dragging the icon to its desired location.
Items may be moved into an open folder by dragging them into the open folder window. Items may be moved into closed folders by placing them on top of the folder icon. When icons move from one folder to another, the files represented by the icons are also moved.
Note: Icons may not be moved from within the Pyxis environment to another windows environment. Nor can items be dragged into the environment directly. See the Importing Files help menu for information about bringing in items from outside Pyxis.
See also:
Working With Files
Working With Folders

Change Icons - Netscape

File Edit View Go Communicator Help

Main Help Menu

Session Help Menu

Icon Help Menu

How to Change the Icon for an Item:

1. Select the item by clicking once on its icon. The item is selected if the text box with the item's name is shaded.
2. Select "Change Icon" from the Icons Menu.
3. Select the image file (".gif" ".ico" ".jpeg" ".bmp") for the icon. Click on the open button in the dialog box. The image chosen will become the icon for the selected item.

See also:

Restoring Default Icons

Change Icon Menu Item

Restoring Default Icons: - Netscape

File Edit View Go Communicator Help

Icon Help Menu

Restoring Default Icons:

This option restores the programs default icons to the Pyxis environment. All user-specified icons will be replaced with the program's default icons.

To restore the default icons simply select the "Restore Default Icons" option from the Icons Menu.

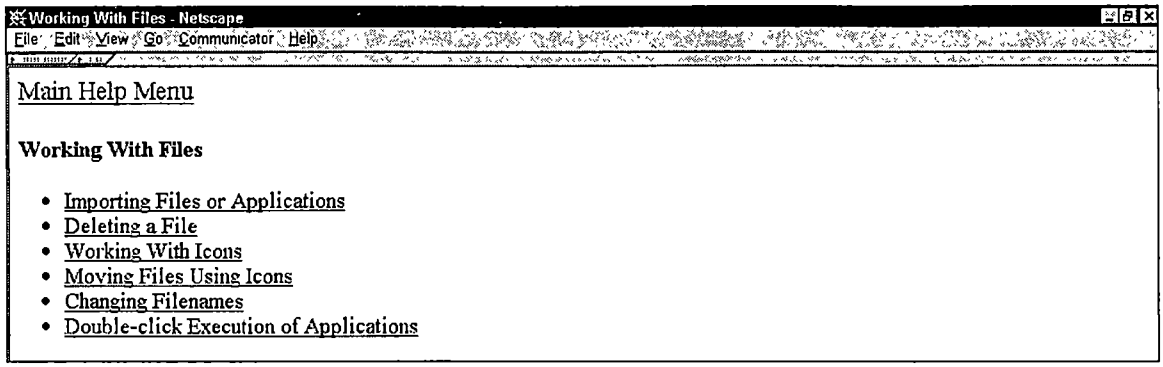
Several of the default icons are shown below:

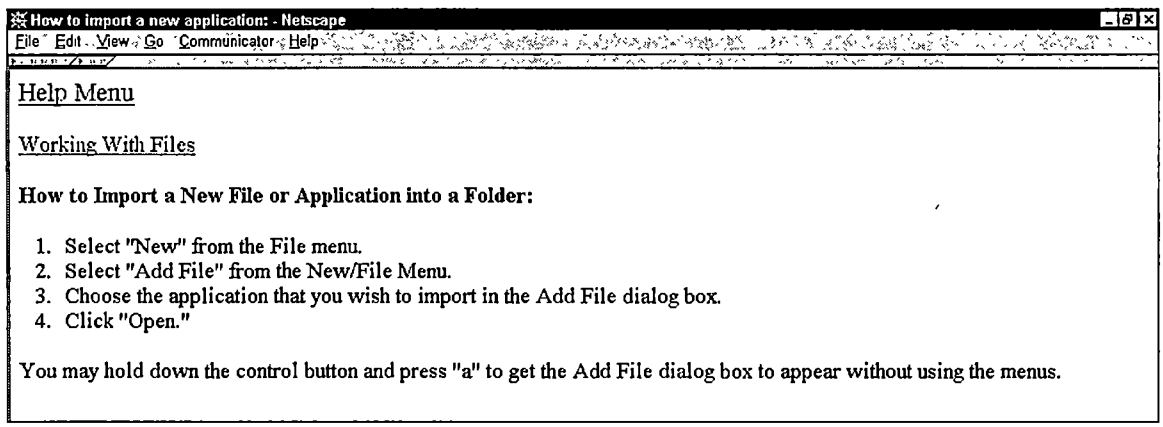
- ".ses" = 
- ".bat" = 
- ".exe" = 
- ".doc" = 
- ".in" = 
- ".out" = 
- ".txt" = 
- other = 

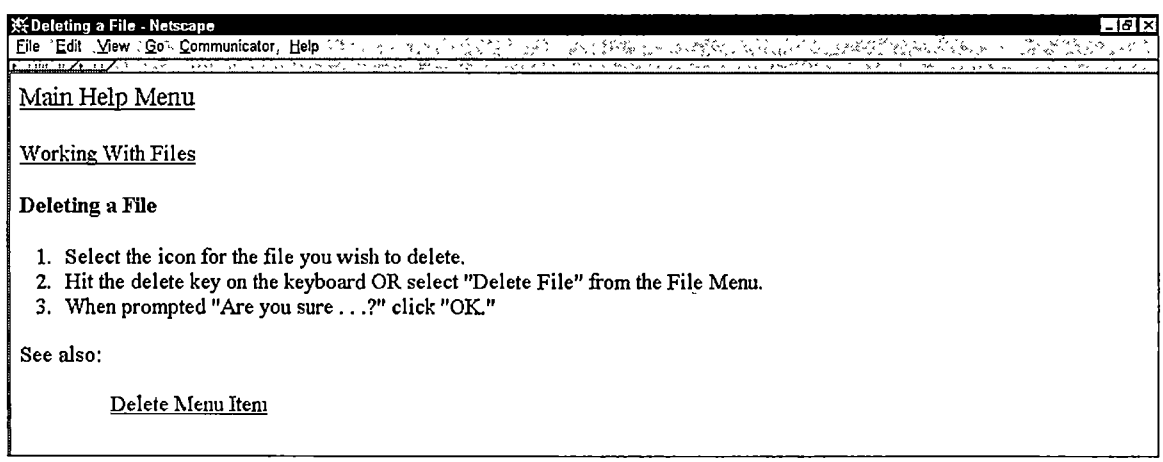
See also:

Changing Icons

Document: Done


Working With Files - Netscape
File Edit View Go Communicator Help
Main Help Menu
Working With Files
Importing Files or Applications
Deleting a File
Working With Icons
Moving Files Using Icons
Changing Filenames
Double-click Execution of Applications


How to import a new application: - Netscape
File Edit View Go Communicator Help
Help Menu
Working With Files
How to Import a New File or Application into a Folder:
1. Select "New" from the File menu.
2. Select "Add File" from the New/File Menu.
3. Choose the application that you wish to import in the Add File dialog box.
4. Click "Open."
You may hold down the control button and press "a" to get the Add File dialog box to appear without using the menus.


Deleting a File - Netscape
File Edit View Go Communicator Help
Main Help Menu
Working With Files
Deleting a File
1. Select the icon for the file you wish to delete.
2. Hit the delete key on the keyboard OR select "Delete File" from the File Menu.
3. When prompted "Are you sure . . .?" click "OK."
See also:
Delete Menu Item

Changing Filenames - Netscape

File Edit View Go Communicator Help

Main Help Menu

Working With Files

Changing Filenames

1. Select the icon of the file you wish to change the name of.
2. Select "Change Filename" from the Icons Menu.
3. Enter the new name for the file in the dialog box that appears.
4. If the name is acceptable for your operating system (not used in that directory, appropriate length for your system, etc.), the name will be changed. This will change the name in the Pyxis environment as well as in the directory itself.

See also:

Change Filename Menu Item

Working With Folders - Netscape

File Edit View Go Communicator Help

Main Help Menu

Working With Folders

- What is a Folder?
- Creating a New Folder
- Opening a Folder
- Moving Items Between Folders

What is a Folder - Netscape

File Edit View Go Communicator Help

Main Help Menu

Working With Folders

What is a Folder?

A folder is a visual representation of a directory on your computer. Files residing in the directory are represented by icons with the filenames written below the icon. A typical folder can be seen below.



Creating a New Folder - Netscape

File Edit View Go Communicator Help

Main Help Menu

Working With Folders

Creating a New Folder

1. From the File Menu, scroll down to "New" then select "New Folder" from the File/New Menu.
2. You will be prompted to give the folder a name. You may assign the folder a name or accept the default name. Blank names or names of existing folders are not acceptable.
3. A new directory will be created and represented by a folder with the same name.

Opening a Folder - Netscape

File Edit View Go Communicator Help

Main Help Menu

Working With Folders

Opening a Folder

Simply double-click on the folder icon (as seen here ) and the folder will open into a folder window. If the folder window is already open, you will receive an error message. Only one folder window may be open for a given folder.

Text Editor Help Menu - Netscape

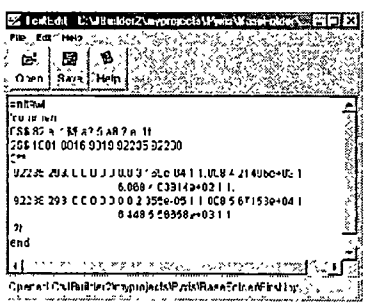
File Edit View Go Communicator Help

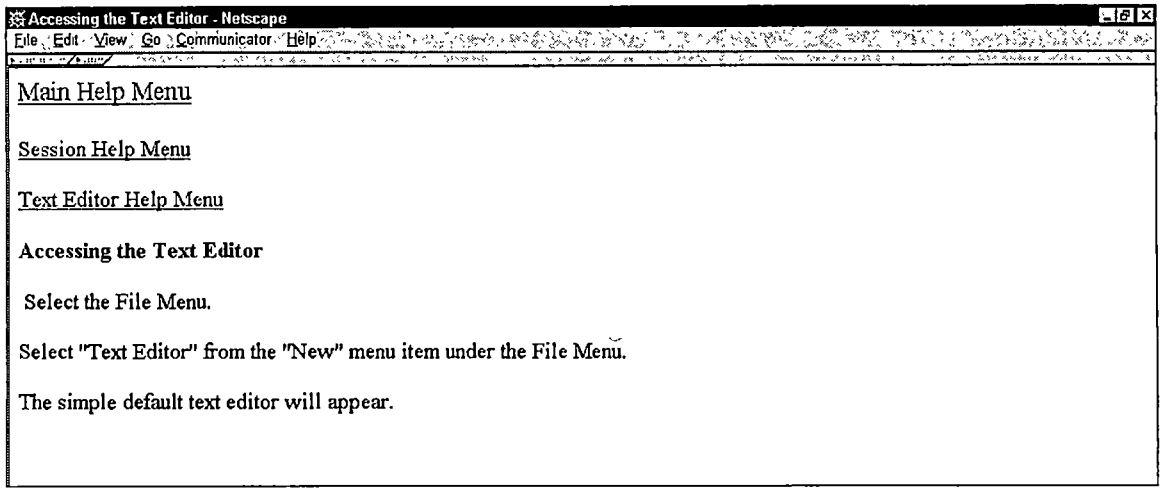
Main Help Menu

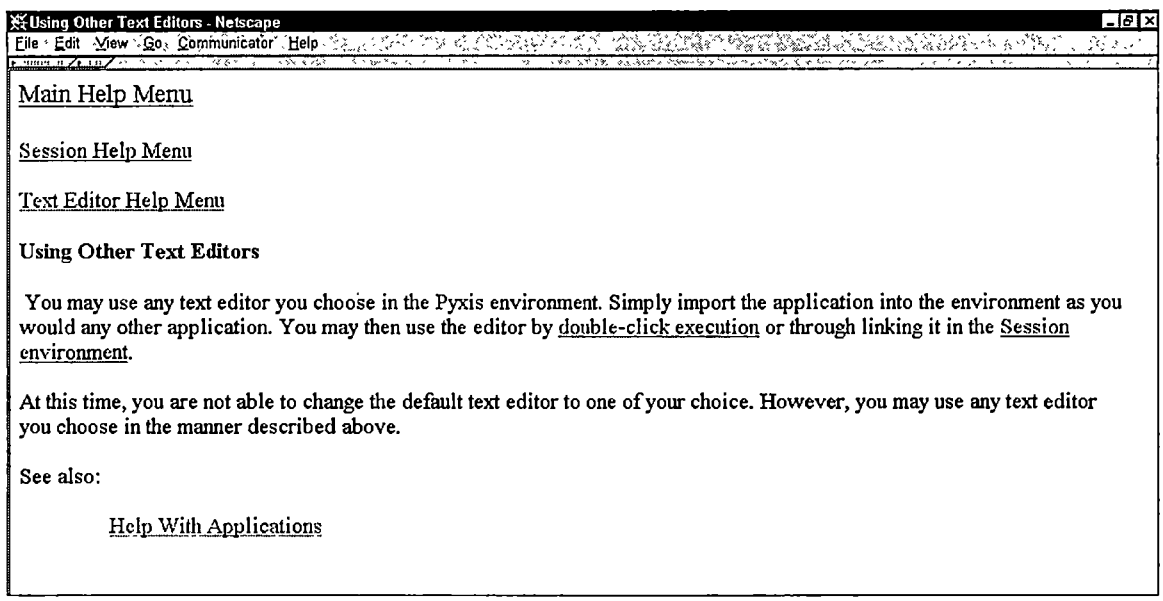
Text Editor Help Menu

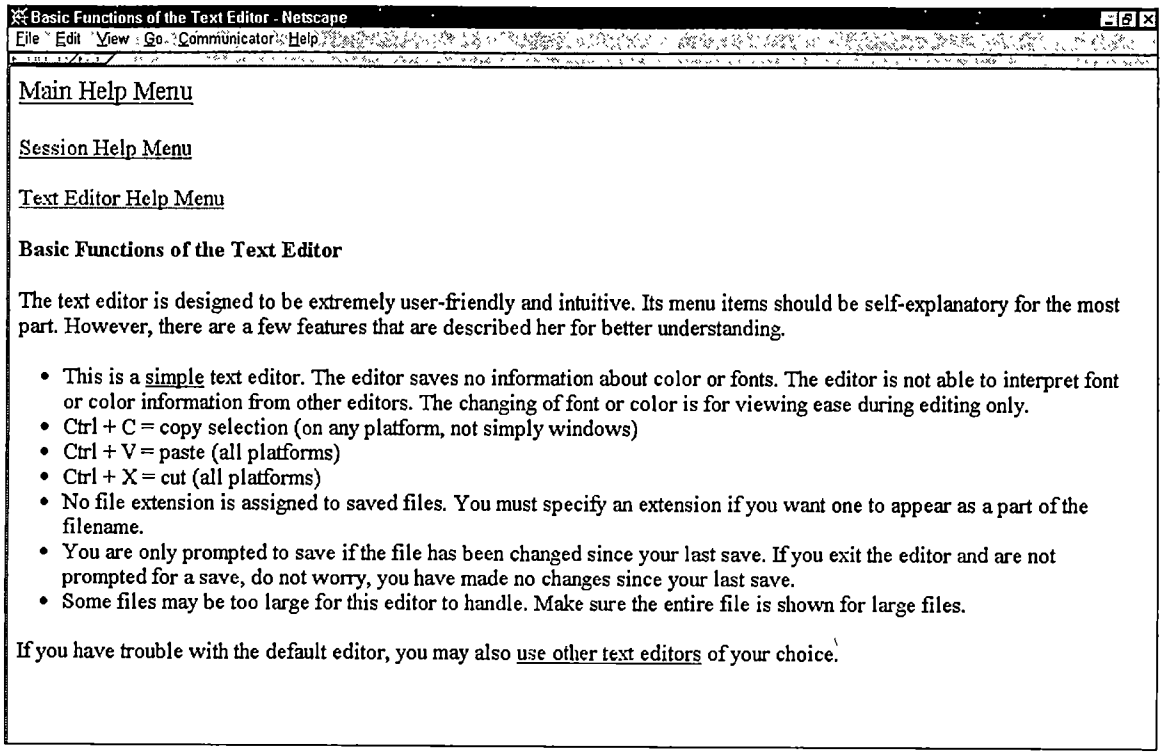
- Accessing the Text Editor
- Using Other Text Editors
- Basic Functions of the Text Editor
- Text Editor FAQ

Below is a Sample of the Text Editor




Accessing the Text Editor - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Text Editor Help Menu
Accessing the Text Editor
Select the File Menu.
Select "Text Editor" from the "New" menu item under the File Menu.
The simple default text editor will appear.


Using Other Text Editors - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Text Editor Help Menu
Using Other Text Editors
You may use any text editor you choose in the Pyxis environment. Simply import the application into the environment as you would any other application. You may then use the editor by double-click execution or through linking it in the Session environment.
At this time, you are not able to change the default text editor to one of your choice. However, you may use any text editor you choose in the manner described above.
See also:
Help With Applications


Basic Functions of the Text Editor - Netscape
File Edit View Go Communicator Help
Main Help Menu
Session Help Menu
Text Editor Help Menu
Basic Functions of the Text Editor
The text editor is designed to be extremely user-friendly and intuitive. Its menu items should be self-explanatory for the most part. However, there are a few features that are described her for better understanding.
This is a simple text editor. The editor saves no information about color or fonts. The editor is not able to interpret font or color information from other editors. The changing of font or color is for viewing ease during editing only.
Ctrl + C = copy selection (on any platform, not simply windows)
Ctrl + V = paste (all platforms)
Ctrl + X = cut (all platforms)
No file extension is assigned to saved files. You must specify an extension if you want one to appear as a part of the filename.
You are only prompted to save if the file has been changed since your last save. If you exit the editor and are not prompted for a save, do not worry, you have made no changes since your last save.
Some files may be too large for this editor to handle. Make sure the entire file is shown for large files.
If you have trouble with the default editor, you may also use other text editors of your choice.

APPENDIX C

Notes to Future Pyxis Developers

1. Java programs require a great deal of memory to run smoothly. If the program is not running well, check the available memory when the program is in use. Free up memory by closing other applications. If working in a Windows environment, make sure there is significant hard drive space remaining because Windows sometimes uses available hard drive space for short-term memory.
2. Java serialization is not very good for complicated objects such as the SessionWindow in the Pyxis code. This problem was the most difficult one that I encountered when creating Pyxis. The best way to resolve this is to serialize only the information needed to reconstruct the object, instead of the object itself. It is usually possible to save all needed information in the form of strings and integers, which serialize well. Then after deserializing the relevant information about the object, recreate it. The "get" and "set" methods are ideal for retrieving the needed paramaters and recreating the object after deserialization of the relevant parameters.
3. Complete platform independence is a work in progress with Java technology. In order to make sure Pyxis runs well on all platforms, certain parts of the code often needed alteration. Pay particularly close attention to the conventions used by different operating systems to structure directories. Unix systems use forward slashes between directories and will not accept backslashes. Most modern Windows and DOS systems accept either the forward or backward slash, although some require the backslash. Edit the parts of the code which contain the slashes to fit the

operating system used. This does not take long once you realize what you are looking for.

4. Some applications are difficult to use in the Pyxis environment without first creating a batch file for them. This is true if the command line is not in the form application.exe (or ".bat") followed by a space and then needed input or output files (if any). If there are switches which need to be entered on the command line, Pyxis will not provide an opportunity to do this. The easiest solution I have found is to create a batch file (or some equivalent file) which specifies the switches and is in the form mentioned above. Pyxis could also be modified to accommodate switches in the PropertiesBox class.
5. As Java evolves I have found it useful to immediately replace depracated methods and classes. This allows the product to run smoothly on the most recent Java version, which is likely to be the one found when Pyxis is deployed.

VITA

Daniel Carlyle Evans was born in Atlanta, Georgia on July 23, 1975. He attended public scools in Huntsville, Alabama and Milton Massachusetts before moving to Brentwood and later nearby Franklin, Tennessee. He completed middle school, then graduated from Brentwood High School in May 1993. He entered the University of Tennessee, Knoxville in August 1993 and received the Bachelor of Science Degree in Nuclear Engineering in May 1997. He began work on his Masters Degree in Nuclear Engineering in August 1997. He was married to Amelia J. Brown, a University of Tennessee graduate student in Food Science and Technology, in March 1998. He spent the summer months of 1998 at Oak Ridge National Laboratory working on his Masters thesis. He hopes to receive his Masters Degree in December 1999 and is presently pursuing a Doctorate of Philosophy in Nuclear Engineering at the University of Tennessee, Knoxville.