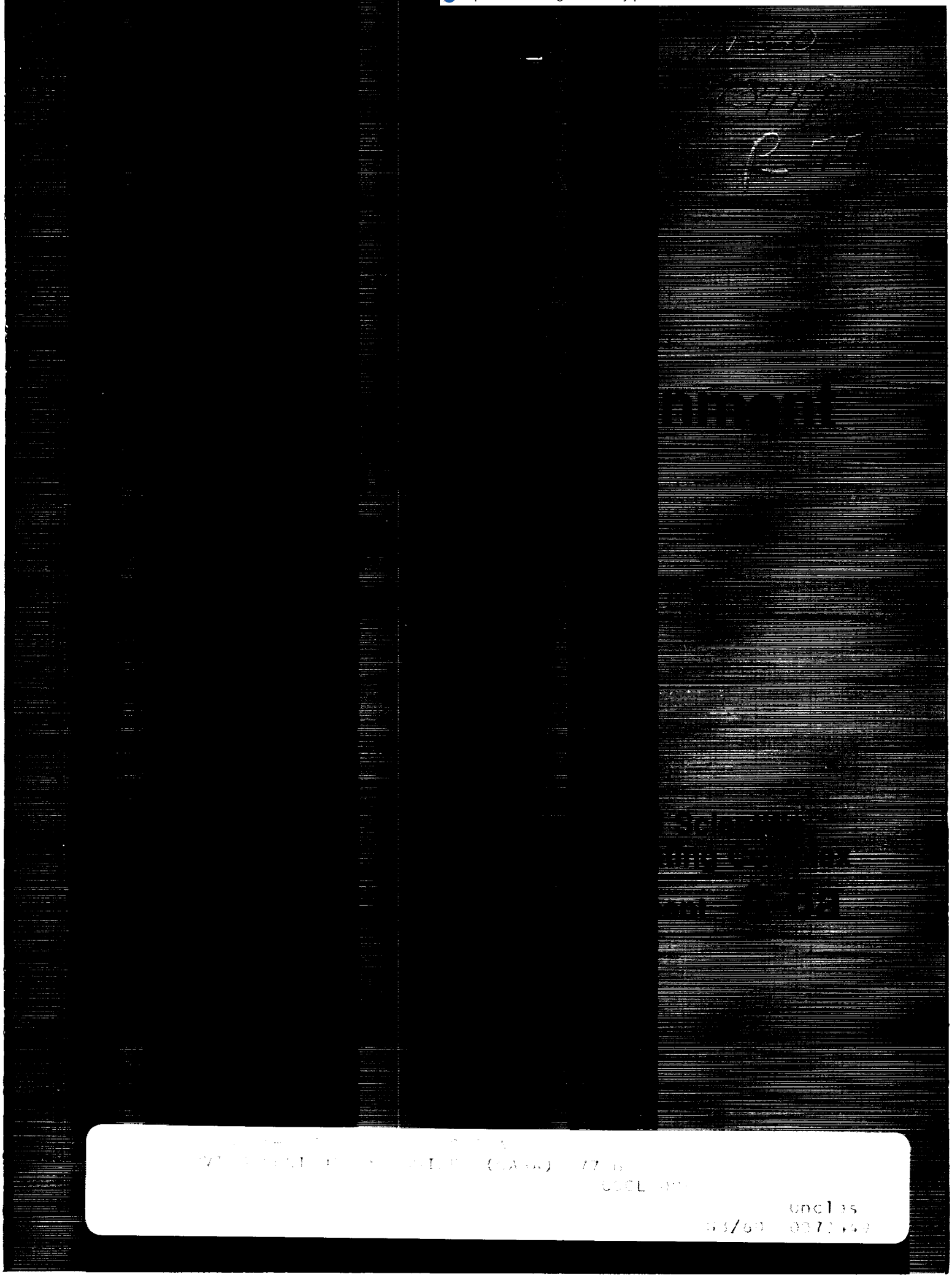

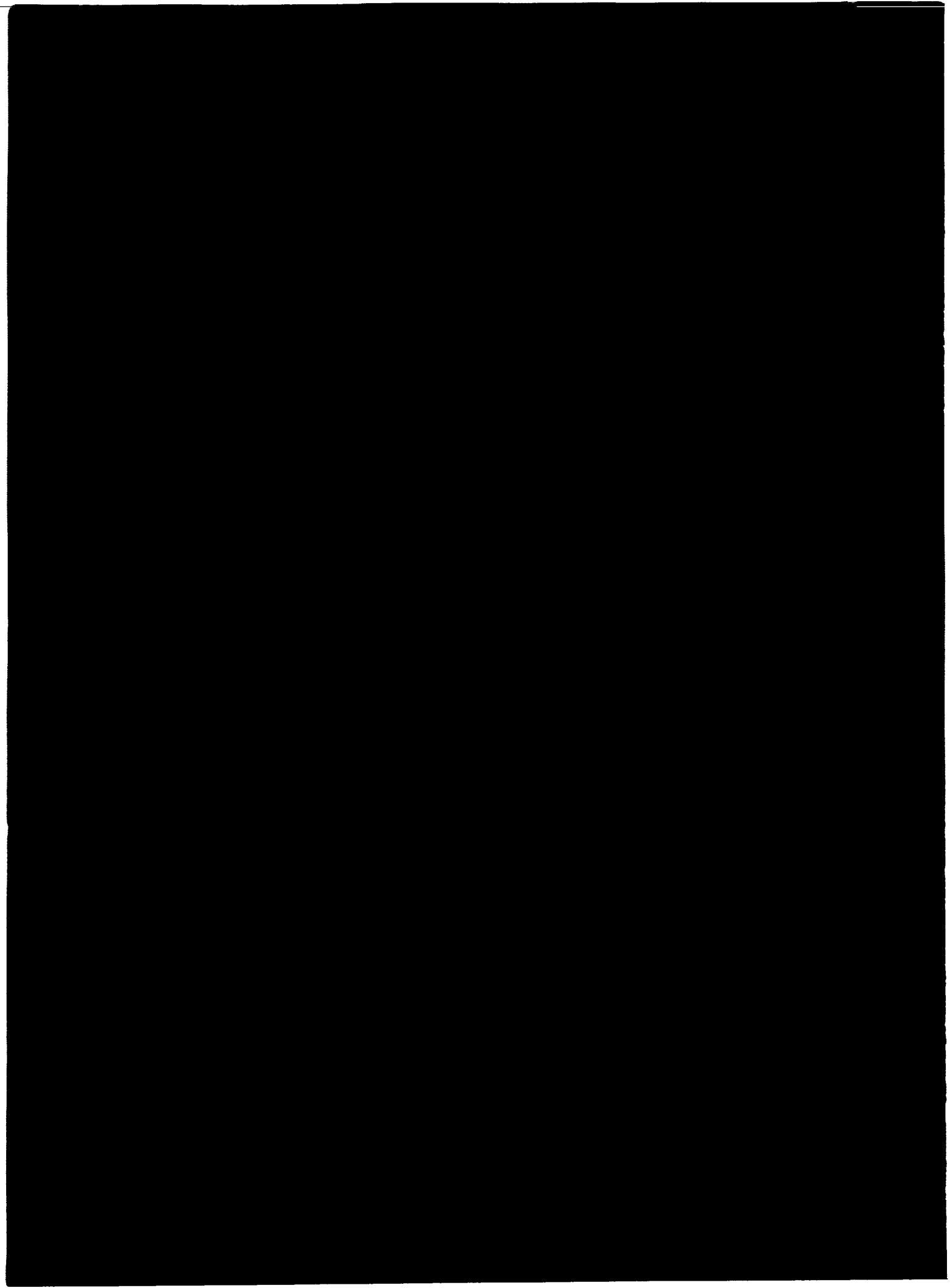

NASA Langley Research Center

Central Scientific
Computing Complex
Document A-8a

SNS Programming Environment User's Guide

February 1992

(Replaces A-8 dated February 1991)

Geoffrey M. Tennille, Lona M. Howser, D. Creig Humes,
Catherine K. Cronin, John T. Bowen, Joseph M. Drozdowski,
Judith A. Utley, Theresa M. Flynn and Brenda A. Austin

PREFACE

This record of revision is a permanent part of the *SNS Programming Environment User's Guide*. This manual is derived from information contained in the CR-1a and CX-1c releases of the *CRAY Mini Manual* and *CONVEX Mini Manual*. In this release, some section headings have a suffix '[a]' added to indicate that the section was added or substantially changed for this revision. This method of annotation was chosen since sections in this manual are generally brief, thus a revision record can be kept with the text of the document. Grammatical, spelling and other changes that don't effect the context of the section are not marked.

DATE	REVISION	MAJOR FEATURES
February 1991	Original	The X Window System; Material from CX-1c & CR-1a incorporated.
February 1992	Revision a	Typographical corrections; Hardware and software upgrades to the computers and Mass Storage Subsystem; Changes to user consultation service; a new graphics package, PicSure

ACRONYMS

ACD	Analysis and Computation Division
ANSI	American National Standards Institute
ARC	Ames Research Center
BLAS	Basic Linear Algebra Subroutines
CAB	Computer Applications Branch
CAL	CRAY Assembly Language
CGL	Common Graphics Library
CMB	Computer Management Branch
CPU	Central Processing Unit
DCM	Division Computing Manager
DRAM	Dynamic Random-Access Memory
EARS	Explicit Archival and Retrieval System
FTP	File Transfer Protocol
GAS	Graphics Animation System
IBM	International Business Machines
IMSL	International Mathematical Statistical Library
I/O	Input/Output
ISO	International Organization for Standardization
LaRC	Langley Research Center
LaRCGOS	LaRC Graphics Output System
LaTS	LaRC Telecommunications System
LCUC	LaRC Computer Users Committee
MFLOPS	Million FLoating Operations Per Second
MOTD	Message-of-the-Day
MSS	Mass Storage Subsystem
NAS	Numerical Aerodynamic Simulator
NCAR	National Center for Atmospheric Research
NCS	NOS Computing Subsystem
NFS	Network File System
NOS	Network Operating System
NQS	Network Queuing System
NSO	Network Support Office
OCO	Operations Control Office
PVI	Precision Visuals, Inc.
RGL	Remote Graphics Library
RM/RMT	Raster Metafile/RM Translator
SAG	Supercomputing Applications Group
SNS	Supercomputing Network Subsystem
SRAM	Static Random-Access Memory
SSIG	Supercomputing Special Interest Group
TCP/IP	Transmission Control Protocol/Internet Protocol

Table of Contents

1. INTRODUCTION

The **Supercomputing Network Subsystem** (SNS) provides most of the Central Scientific Computing Complex's computational power. The **SNS** has a rapidly changing environment, however a significant portion of the software is common, or at least very similar, on all the machines. The *CRAY Mini Manual* (CR-1) and *CONVEX Mini Manual* (CX-1) are devoted to describing the hardware, software and programming issues particular to each class of machine. The next section describes this manual, *The SNS Programming Environment User's Guide,* which describes the common features of the SNS computers.

1.1. The SNS Programming Environment User's Guide [a]

This manual basically combines a description of the common features of the **SNS** computers into one manual. References to other documentation omit revision levels. Document A-5, which is accessible on-line on **Eagle** and **Mustang,** in the directory *~uchelp/Doc/A5,* contains an up-to-date listing of all CSCC documentation and other sources of user information such as the ACD Customer Services List, Division Computing Managers and Document Librarians.

This manual contains information that is less likely to change rapidly than the CRAY and CONVEX Mini Manuals, hence it will not be updated as frequently. New information is disseminated via computer bulletin and by use of the *notes* utility on the CONVEXs prior to being incorporated into A-8. This revision incorporates changes resulting from hardware upgrades to the CRAY Y-MP, CONVEX computers and the Mass Storage Subsystem (MSS). Also included are changes from software upgrades to the operating systems on all **SNS** computers. The provision of user services and consultation has also undergone a major transformation that is described in Chapter 8.

As with the Mini Manuals, comments and suggestions from users are welcomed.

1.2. SNS Configuration [a]

The production portion of **SNS** is composed of the two CONVEX computers, **Mustang** and **Eagle;** the CRAY-2S supercomputer, **Voyager;** the CRAY Y-MP supercomputer, **Sabre;** a mass storage subsystem; a high speed HYPERchannel network that ties them together; and a SUN gateway computer, **Goose,** which connects the high speed network to LaRCNET. The CONVEX machines, **Eagle** and **Mustang,** are connected to both networks. **Sabre** and **Voyager** both have two connections to the HYPERchannel, which necessitates two entries for each machine in your *.rhosts* files to insure that remote shell execution works (see section 5.2). A third entry in your *.rhosts* files for both CRAY computers is necessary to access the MSS (see section 5.5).

1.2.1. Choosing the Proper Resource [a]

The SNS computers at Langley Research Center (LaRC) and the Numerical Aerodynamic Simulator (NAS) computers at Ames Research Center (ARC) provide LaRC users with a wide spectrum of resource availability in terms of CPU power, memory size and disk space. All these computers run UNIX-based operating systems, so that portability and communication among the various components is relatively easy.

Because of the large user base at LaRC, it is in everyone's best interest that users select an appropriate computer on which to work. There are programs, that if run on one of these computers will tax that resource and greatly impact other users computing on the same machine. That same program might run comfortably on another computer available to the user.

Management of various aspects of the system, such as permanent file allocation, amount of scratch space set aside and definition of batch queues, requires that a philosophy be assumed about the use of the computers. **Table 1.1** (on the next page) contains a summary of that philosophy in the form of guidelines for the individual to use in selecting the appropriate place to compute. It is possible that experience may dictate a change in these guidelines, but users are expected to abide by them as closely as possible.

Users whose jobs dictate that they must run on **Navier,** the 256 Mwd CRAY-2D or **Reynolds** the 256 Mwd CRAY Y-MP/8 at ARC, should contact Jay Lambiotte, the LaRC NAS User Liaison, at 864-5794.

	Interactive Jobs		Batch Jobs						Machine	
			Small		Medium		Large		Description	
	Max Memory (mwds)	Max CPU (min)	Max Memory (mwds)	Max CPU (hrs)	Max Memory (mwds)	Max CPU (hrs)	Max Memory (mwds)	Max CPU (hrs)	Memory (mwds)	CPU Power
Eagle	1.	20	1.	1	2.	2	4.	2	32.	1.0
Mustang	2.	20	2.	1	4.	2	8.	2	64.	1.0
Voyager	10.	10	8.	2	20.	3	40.	4	128.	5.9 *
Sabre	5.	7	4.	3	15.	3	40.	3	128.	9.4 *
Navier (NAS)	20.	10	16.	2	40.	2	120.	2	256.	4.7 *
Reynolds (NAS)	20	10	16.	4	40.	4	120.	4	256	9.4 *

** The Power Rating is for each processor*

Table 1.1 - Guidelines for Selecting the Appropriate Computer [a]

NOTES:

1. These guidelines represent ACD's perspective of the use of these computer systems as of February 1992 and are not actual batch queues.

2. The queues described above are generic, but relevant in that system parameters are normally tuned so that job response increases as you move right to left in the job categories of **Table 1.1.** The NQS queues for **Voyager** and **Sabre** may be found in **Table 4.1** and **Table 4.2** of CR-1 and under *notes CRadmin.* Similarly the CXbatch queues for **Eagle** and **Mustang** may be found in **Table 4.1** of CX-1 and under *notes CXadmin.*

3. CPU power (per processor) is determined by comparing performance of a program that measures scalar speed and vector speed over a range of operand lengths and strides.

1.2.2. Communication with TCP/IP Computers

Since both CRAY supercomputers access LaRCNET through **Goose** (the SUN gateway), a user logged on to any other TCP/IP computer on LaRCNET (or the **SNS** high-speed network) may remotely login to either CRAY via *telnet* or use the *ftp* utility to pass files between computers. If the remote computer also runs UNIX, the analogous *rlogin* and *rcp* utilities may be used between it and the CRAY supercomputers. It should be noted that computers which are a part of the NAS configuration are also accessible as they are logically connected to LaRCNET through NASNET. The CONVEXs are directly accessible through LaRCNET, the Langley Telecommunications System (LaTS) and SURAnet.

Although the CRAY supercomputers are not directly accessible through LaTS, you may use *CALL LARCNET* to access the **cisco** box. The system responds:

Welcome to the LaRCNET Terminal Server
CISCO1>

Then by typing *voyager* you connect to **Voyager.** Any other computer on LaRCNET can be accessed in the same manner.

1.2.3. Communication with NCS Computers

Files can be passed to and from NOS Computing Subsystem (NCS) computers by using the *ftp* utility (See section 5.3). The NCS machines are still called **a, y** and **z.** Using *ftp* from the **SNS** computers to the **y** NCS computer works as described in section 5.3; however, using *ftp* from a NCS computer to a **SNS** computer is somewhat different. See *CDCNET TCP/IP Usage,* Document N2-46, for details).

For example, using *ftp* from a NCS computer to *get* a file from any **SNS** computer brings the file over to NOS as a local file. You must explicitly *SAVE* the file on NOS. The *put* command works as you might expect on NCS to **SNS** transfers.

It is also possible to use *telnet* to login to one of LaRC's central NCS computers from any of the **SNS** computers (See section 5.1). To use *telnet* from a NCS computer to any **SNS** computer requires the *%crec* command (See N2-46).

1.3. SNS Validation [a]

Each user of the computers in **SNS** must be specifically validated for **SNS** by the Computer Management Branch (CMB) of ACD. To receive access to **SNS,** a user must have a valid login name (at most eight characters starting with a letter) and be authorized on a valid **SNS** account (any account beginning with the letter "a"). If you need to be validated as an **SNS** user or wish to change existing validation, then you should complete the **LaRC Computer User Authorization Form,** which is available from your Division Computing Manager (DCM). If you do not know who your DCM is, call OCO at 864-6562. They have a current list of DCM's arranged by organization code. A current list of DCM's may be found on-line in Document A-5 (see section 1.1) as well.

If you are not already validated for an **SNS** account, the **LaRC Computer Account Authorization Form,** also available from your DCM, can be used to establish a new **SNS** account or to add a user to an existing account. When validated, you will receive a password that you must change the first time you login to each of the **SNS** computers (See section 4.1.1). All newly validated **SNS** users receive copies of this manual (A-8), CR-1 (the *CRAY Mini Manual*) and CX-1 (the *CONVEX Mini Manual*). Section 7.1 of both Mini Manuals lists additional documentation, some of which is sent to all Document Librarians. You may request copies of any of these user manuals from OCO by calling 864-6562 or by sending electronic mail to:

oco@eagle

1.3.1. Charging, Accounts and Groups

CRU charging for **SNS** computers is currently for CPU time only. The relative charges for the **SNS** computers reflect the relative power ratings in **Table 1.2.** The entire charging procedure is periodically reviewed and changes are made, if warranted. At login, you are informed of the account to which your CPU time is being charged. If you are authorized for more than one account, it is simple to change to another, as illustrated later in this section.

On the **SNS** computers, account names and group names are the same. They all begin with the letter "a". An individual validated for multiple accounts (and hence for multiple groups) should be aware of the differences between accounts and groups.

Accounts are used to charge for CPU time. You have a default account that is in effect whenever you log into any **SNS** computer. The account that is active when a process starts is the account that is charged for that process. If you have multiple accounts, the *charge* command can be used to change the active account for the interactive session or to change the default account for subsequent sessions.

Groups are a UNIX feature designed to facilitate file access and sharing. By default a file created in a directory belongs to the group to which that directory (its parent directory) belongs. This ownership can be displayed by typing

ls -lg

which lists all your files in a long format with group identifiers (See section 2.2.7). You may change the group ownership of a file with the *chgrp* command, which is illustrated later in this discussion. In particular, the group ownership of your home directory is the same default name as your default account at login. The group ownership of your home directory is changed whenever you use the *charge* command to change your default account.

For example, if you have two accounts a00273 and a00373, with a00273 being your default, then the command

charge a00373

changes your account for the current interactive session to a00373. All subsequent charges for CPU time during the current interactive session are made to a00373. There is no change to your default account with this use of *charge* and hence the file ownership of your home directory is still a00273. All files created in your home directory, while charging to a00373, thus belong to a00273. However, if you use the command

charge -b a00373

then both your current account and your default account are changed. Your home directory ownership is also changed to a00373, so that all new files created under your home directory belong to a00373. There are other options for the *charge* command that can be read in the *man* page for *charge*.

If you create a file, *file_name,* with ownership a00273, but you want it to belong to a00373, then the command

chgrp a00373 file_name

changes the group ownership of *file_name* to a00373 (See section 2.2.8).

1.3.2. Permanent File Quotas [a]

Each user is given some amount of permanent allocated file space on **Eagle, Voyager** and **Sabre.** There are no user home directories on **Mustang.** If you are unable to operate without additional space, on any of these computers you may petition for a larger allocation by using the **LaRC Resource Authorization Form** available from OCO at 864-6562.

You have both "soft" and a "hard" quota limits. If you exceed your "soft" limit, the operating system sends you a warning message. However, the system does not permit you to exceed your "hard" quota limit. On the CRAY's, the "soft" limit is 85% of your "hard" limit. On **Eagle,** the "hard" limit is 125% of your "soft" limit.

To check on your file quota on the CRAY's, use the *quota* command without any parameters. The utility *quotamon* can be used in your *.login* file (See section 4.1.4) to give you a periodic warning when you have exceeded your "soft" quota. The example in section 4.1.4 of CR-1 uses

quotamon -s 600

to set the periodicity to 600 seconds.

For more information on *quota* and *quotamon,* check the *man* pages. They are not yet documented in CR-6. In addition, you share a large scratch space on each **SNS** computer with other users, which is not regulated by quotas. The use of permanent and scratch file space is discussed in section 4.1.

You may check your limits and usage on **Eagle** with the *quota -v* command. It displays your disk usage and limits in units of kilobyte blocks. On **Mustang,** *quota -v* only prints a header, without any other information.

You may also check your usage for each subdirectory with:

du -s

It displays your disk usage in units of 512 word (4096 bytes) blocks for each directory from your current working directory downward. There is generally a discrepancy between the number of blocks that *du* returns for your disk usage and the number indicated by *quota* on the CRAY's but not on **Eagle.** At the time this manual was printed, files sized less than 8 blocks (32,768 bytes) on the CRAY's are allocated one block at a time. Files greater than 8 blocks in size are allocated in chunks of 48 blocks. FORTRAN source codes of 2,000 to 4,000 lines typically use 10 to 20 blocks according to *du,* however they consume 48 blocks of your quota.

1.3.3. Default Home Directory Access Permissions

Each user is installed on **SNS** with a set of system provided file access permissions (See section 2.2.8) for his home directory. These permissions are read, write and execute for the user, while members of his group and the world are granted only read and execute permissions. Read access for the world on your home directory allows any other user to list your files' names with the *ls* command, while execute permission allows other users to use the *cp* or *cat* commands on any of your files that have read permission turned on for the world. These accesses can be very useful in file sharing and with consultation issues. However, you should be careful to remove permissions for any file containing sensitive information, such as your *.rhosts* or *.netrc* files (See section 5). Home directory permissions can be modified if desired.

2. UNIX ON THE SNS COMPUTERS

UNICOS is derived from AT&T System V UNIX, a standard UNIX operating system and the CONVEX OS is an enhancement of U.C. Berkeley UNIX 4.2. UNIX is the basic user interface for all **SNS** computers.

2.1. UNIX Overview

UNIX and its derivatives, such as UNICOS and CONVEX OS, have become popular operating systems for a variety of reasons, chief among them being that UNIX is portable, since it is written in C. UNIX is being used generically in the discussions that follow. Only when a feature is directly related to UNICOS or CONVEX UNIX is a distinction made. Thus each vendor does not have to invest the time and money in developing an operating system, but must write only device drivers and a C compiler. It is also a powerful and flexible system, with each command being a separate entity but able to be linked with other commands to form more powerful constructs. There are several good books on UNIX available from the Center library. CRAY and CONVEX have provided a large set of documentation (See section 7.1 of the Mini Manuals). Additionally, training is offered for both UNIX and **SNS** specific issues (See section 8.4).

2.1.1. UNIX Features

There are numerous important features of the UNIX operating system that differ from other non-UNIX operating systems with which you may be familiar. A few of these features are:

1. Upper case and lower case characters are significant.

2. Commands and utilities can be executed interactively (foreground process) or as a "batch" job (background process) (See section 4.2.1).

3. True batch processing, as opposed to executing a background process is available with the Network Queuing System (NQS) on the CRAY and with CXbatch on the CONVEX (See section 4.3.1 of CR-1 and CX-1 respectively).

4. There are different user interfaces, called shells, available under UNIX. Each has different attributes and capabilities. The default shell is the C-shell.

5. It is possible to create utilities called shell scripts. A shell script is a file containing commands and utilities to be executed (See section 2.1.3)

6. Most foreground processes can be terminated with ^C (Control-C).

UNIX has commands and utilities that can be referenced through the on-line documentation using the *man* pages (See section 8.1.1) or in Documents CR-6, the *CRAY UNICOS Commands Reference Manual* and CX-6 the *CONVEX OS Man Pages for Users.*

2.1.2. UNIX Command Syntax

A UNIX command is entered in lower case as follows

command [options] [expression] [file_name]

where only *command* is required, the other three arguments are optional and dependent on the particular command (See section 2.1.4 of CR-2). Throughout this manual command arguments enclosed in square brackets are optional. Usually *options* are preceded by a minus sign (-). Options modify the command or give specification as to how it is to be executed. A character string used as input to a command is referred to as an *expression.* Finally, *file_name* is the ordinary or directory file upon which the command is to perform. The examples listed next illustrate the use of each type of argument:

ls -al
echo 'You have a meeting in ten minutes.'
cat myscript

These three commands give a long listing of all files in your current working directory (See section 2.2.7.1); cause the message **You have a meeting in ten minutes.** to be output to your screen; and list your script file *myscript* to the screen.

2.1.3. Shell Scripts

The default interactive command interpreter for UNIX at LaRC is the C-shell, although the Bourne shell is available. Chapters 4 and 5 of CR-2 have detailed information on writing shell scripts in the Bourne and C shells respectively. However, by default scripts execute in the Bourne shell. To execute in the C-shell, scripts must begin with either a pound sign (#) in the first column of the first line or with *#!/bin/csh* as the first line. To explicitly invoke the Bourne shell use *#!/bin/sh* as the first line of the script.

To create a shell script, use any editor to put the commands you want executed repeatedly into a file. Any line beginning with the pound sign is interpreted as a comment, so that the script can be made self explanatory. Once the file is created, it must be given the appropriate execute permissions (See section 2.2.8). It can then be executed as a regular command.

2.2. Files and Directories

An important attribute of the UNIX operating system is the hierarchical file structure. User files are of three types: ordinary files, directories and hidden files. The hidden files *.login* and *.cshrc* are discussed in sections 4.1.4 and 4.1.5 of both CR-1 and CX-1. The hidden files *.exrc, .forward, .mailrc, .rhosts* and *.netrc* are discussed in sections 2.3.1, 3.1.3, 3.1.4, 5.2 and 5.3 (of this manual) respectively. Chapter 2 of CR-2 and chapter 6 of CX-1 have diagrams of some typical file system structures.

2.2.1. Ordinary Files

A file under UNIX is like a file under any other operating system; it is simply the residence of some set of data. Ordinary files under UNIX contain text, ascii data, binary data, source code and the like. UNIX file names have fewer restrictions than file names under other operating systems. These restrictions are:

1. UNICOS is restricted to having file names of fourteen (14) characters or less, while CONVEX UNIX file names may have up to two-hundred-fifty-five (255) characters. For compatibility between **Voyager, Sabre, Mustang** and **Eagle** you should limit file names to fourteen (14) characters or less.

2. Any alphanumeric character may be used. These include letters, digits, the period and the underscore character.

3. Upper case and lower case letters are significant. The files TAPE10 and tape10 are different to UNIX.

4. File names beginning with the period (.) are hidden files and are not listed by the basic *ls* command.

5. In order to be recognized by the compilers, FORTRAN, Pascal, Ada and C source code files must end with *.f, .p, .a* and *.c* respectively.

2.2.2. Directories

A directory in the UNIX environment is a file because it contains data; however the data that a directory contains is information about other files and directories. Any directory that resides within another directory is called a subdirectory. This concept is the basis for the hierarchical nature of the file system. A handy naming convention that provides a distinction between ordinary files and directories is to use lower case letters for the names, except for the first character of the directory name. Then a basic *ls* command shows which of your files are ordinary files and which are directories.

2.2.3. Home Directory

When you receive your login name and account for **SNS,** you are assigned to a specific directories in the respective file systems for each machine. These directories, whose path names end with your login name, are your home directories. Within them you can create or remove files and directories as you please up to your cumulative limit. Every time that you login to any **SNS** computer, you are placed in your home directory to begin your interactive session.

2.2.4. Current Working Directory

Although you are automatically placed into your home directory when you login, you may change to any directory for which you have access permission. However, you can only be in one directory at a time. The directory in which you are located is called your current working directory. If you are not sure of the directory in which you are currently working, the command *pwd* will print your current working directory.

2.2.5. Scratch Directory

The UNIX operating system is file prolific. A FORTRAN compilation, load and execute may generate several large files for which you will have no need once the program completes. For this reason, it is often a good idea to run your large applications in a scratch directory (See section 4.1), to avoid cluttering your home directory. File space in a scratch directory is not backed up and is only temporary; however it does not count against the space allocated for your home directory. The management of scratch directories is dynamic; as usage warrants it may change.

2.2.6. Path Names

To specify where a file or directory exists within the UNIX file structure, it is necessary to use path names. Path names may be either full or relative. A full path name is the complete specification of where that file resides in the file system beginning at the root. A relative path name specifies the location of the file with respect to the current working directory or some home directory.

Path names consist of directory names, slashes (/) and a file name (if a file is required by the command). Full path names begin at the root, designated by a slash (/), subdirectories are delineated by slashes and the file name is the last name specified. UNIX always assumes that the files that you reference are in your current working directory, unless you specify a path name to the file.

Absolute path names can become rather long and cumbersome, however UNIX has several features that can be used to shorten the reference to either your own files or someone else's files. Using any shorthand notation for a path name yields a relative rather than an absolute path name. Your home directory may need to be moved because of space limitations on a particular disk. If this happens, the structure of your file space won't change, however the path to your home directory from the root of the file system will change. Thus use of a shorthand notation will minimize the changes necessary for commands and batch scripts to execute correctly.

The tilde (~) shorthand notation is perhaps the easiest method. If your login name is *john,* then using the notation *~john* specifies the full path to your home directory from the root of the file system, without having several directory names and slashes (/) preceding your file name. This notation is also an efficient means to access another user's files when you don't know the complete path. Your shell expands the tilde into the complete path. Alternatively, you can set a variable within a script to the path name of the directory where the files you wish to access reside. For example,

set hispath = /cr/et/dca/lyh/john/Progs

allows you to reference the file *filex* belonging to *john* in his directory *Progs,* with the notation *$hispath/filex,* rather than the full path name:

/cr/et/dca/lyh/john/Progs/filex.

The *ln* command (See section 2.2.7.10) can be used both to establish a shorthand notation to reference a file belonging to you or another person, and more importantly, as an efficient mechanism to allow users to share the same file without having duplicate copies. The command

ln filey /cr/et/dca/lyh/john/Progs/filex

establishes an entry *filey* in your current working directory which is a link to *john's* file *Progs/filex.* The name *filey* appears in your directory when you use the *ls* command, even though it actually resides in the directory *john/Progs.*

Two other frequently used shorthand notations used in relative path names are the period (.) and double period (..). They represent your current working directory and it's parent respectively. For example, the file *../Dir1/file1* is in the directory *Dir1* which has the same parent directory as your current working directory. In particular, if your path variable does not include the directory ".", (See section 4.1.4) then you will be unable to execute scripts or programs within your current working directory. UNIX will tell you that the command was not found.

2.2.7. Frequently Used UNIX File and Directory Commands

This section briefly describes some of the more frequently used file and directory manipulation commands. More information on all of them may be obtained with the *man* command or in Documents CR-6 and CX-6.

2.2.7.1 ls

The *ls* command lists the names of files and subdirectories in the current working directory or directory specified on the *ls* command. A common option combination is

ls -al

which lists all files, including hidden files in a long format that includes lengths, date of modification, access permissions and other information.

2.2.7.2 cp

The *cp* command copies the contents of the old file into the new file. The syntax is:

cp old_file new_file

Both old_file and new_file exist after the *cp* command, unlike the *mv* command discussed in section 2.2.7.5.

2.2.7.3. cat

The *cat* command concatenates the contents of one or more files and displays the result to the screen. Examples of this command are:

cat file1 file2 | more

cat file1 >> file2

In the first example, the symbol " | " is the pipe symbol and *more* is a filter that only allows one screen of information to be displayed before taking a pause (See section 3.2). In the second, *file1* is *file2*. The *more* command can also be used to display one or more files in the following manner:

more file1 file2

2.2.7.4. rm

The *rm* command removes (purges) a file from the system. If you use the form

rm -i file_name

then the system will prompt you to insure that you really wanted to remove the file called *file_name*.

2.2.7.5. mv

The *mv* command moves the specified file to a new location and may optionally change its name. For example, the command

mv message ./New/msg

would move the file *message* from your current working directory to the subdirectory *New* and rename it *msg*. The file *msg* would have the same group ownership as its parent directory *New* (See section 1.3.1). If the optional name *msg* is not specified, then the file retains the name *message,* under the subdirectory *New*.

2.2.7.6. pwd

The *pwd* command echos your current working directory to the screen. It has no arguments.

2.2.7.7. cd

The *cd* command changes your current working directory. The syntax is:

cd [path_name]

Without an argument it changes back to your home directory. If the argument is two periods (..) you are changed into the directory immediately above your current working directory in the file structure. If you use a period (.) as the argument, you remain in your current working directory.

2.2.7.8. mkdir

The *mkdir* command creates a directory. So if you type

mkdir Program

you create a subdirectory called *Program* under your current working directory.

2.2.7.9. rmdir

The *rmdir* command deletes a directory. The directory to be deleted should be empty before trying to delete it. Once you have removed all files under your subdirectory *Program,* then if you type

rmdir Program

the subdirectory, *Program,* will be removed.

2.2.7.10. ln

The *ln* (link) command allows you to specify a name by which to reference another file usually in a different directory. This may be helpful for users who share files, since the file physically resides in only one directory, but may be shared by many users. To use the *ln* command, the *target_file* must be in the same file system as your home directory. The syntax is:

ln file_name target_file

After the link is created, the file called *file_name* appears in your directory when you execute the *ls* command, but does not take up any space as reported by the *quota* command.

If someone creates a hard link to a file in one of your directories, then you must exercise caution when removing the file. If you remove a file that has a link to it, only your link to the file is destroyed. The file still exists and is charged against your quotas. Even though the file will only appear in someone else's directory, you are still the owner, however you cannot remove the file.

2.2.8. File Permissions

When you type *ls -l* the display on your screen will look something like this:

```
drwxr-xr-x    1   john     1024   Jan   17   7:46   Program
-rwxr-xr-x    1   john    12634   Jan   26   9:18   prog.f
-rwxrwxrwx    1   john      512   Jan   17   8:31   run_it
```

The first field tells whether the file is a directory or not, by having a "d" or "-" as the first character and gives the file access permissions for each file. The second through fourth characters give your permissions; the fifth through seventh give your group permissions; and the eighth through tenth give permission for anyone else who can login to the machine. See section 2.5.3 of CR-2 for additional information on modifying file permissions. The other fields show number of links to the file, owner, date and time of last modification and file name, respectively.

The "rwx" indicates read, write and execute permission exist, while a dash indicates that the permission is turned off. The user has control over these permissions. They can be changed or set in several ways, such as having default permissions eliminated in your *.login* file. The *chmod* command gives you another way of changing the permission. The *chmod* command has two forms. One easy way to understand *chmod* is to think of each group of three permissions, rwx, as a binary representation of an octal number. A one bit adds the permission and a zero bit takes the permission away. If you wished to change the permissions on your file *run_it* to allow other members of your group just to read and execute the file; and for the rest of the world to just read your file, then you would type

chmod 754 run_it

and the permission field would change to -rwxr-xr-- which you would see when you did your next *ls -l*. Note that using bits to represent the permission field "rwxr-xr--" yields the binary representation 111 101 100, which are the octal digits 7, 5 and 4 respectively.

The other form of *chmod* has the syntax:

chmod [ugo] [+][-][rwx] file_name

You either grant "+" or deny "-" specific file accesses ("r", "w" or "x") for a given file to some class of users ("u", "g" or "o"), where *ugo* refers to user, group and others. If none of these classes is specified the *chmod* command executes as if all were specified. If the file *run_it* had the permissions *rw-r--r--*, then to make the permissions *rwxr-xr--*, the following form of the *chmod* command may be used:

chmod ug+x run_it

2.3. Editors

There are four commonly used editors in the UNIX environment: *vi, ex, ed* and *emacs*. The *emacs* editor is available only on the CONVEX computers. Each of the editors is invoked by typing the editor name, followed by a file name, such as:

vi newfile

When invoking the *ed* editor for a new file, you will get a warning message that the file can't be opened, however it is created for you. The following sections describe the basic features of each editor.

2.3.1. vi

The *vi* editor is a full screen display editor. More detailed information may be obtained by using the *man* command or by referencing the *CRAY UNICOS Editors Primer* (CR-7) or the *CONVEX Text Editor's User's Guide* (CX-7). Major features of *vi* include:

1. It uses an edit buffer. What you see displayed on the screen is a window into this buffer.

2. It has three modes of display: open, visual and *ex* command modes.

3. It has two modes to enter text: text insertion and command modes. The escape key (Esc on most keyboards) exits from text insertion mode.

4. Commands for the *ex* editor may be executed when you are not in insert mode by typing a colon (:) to precede the command. To exit *vi* without saving changes, type *:quit!*. To quit and save changes type *:wq*.

5. It is very important to have your terminal characteristics set properly when using *vi,* especially your terminal definition (See section 4.1.2). In particular, if your cursor keys do not function properly, you may need to create a file called *.exrc,* that will be invoked every time you enter the *vi* editor. In this file, the *h, j, k* and *l* keys are mapped into the corresponding cursor keys as shown below:

   :map ^v[up arrow] k
   :map ^v[down arrow] j
   :map ^v[right arrow] l
   :map ^v[left arrow] h

6. There is no prompt in *vi*.

2.3.2. ex and ed

The *ex* and *ed* editors are general purpose interactive line editors. Neither editor automatically displays the file which you are editing. The *ex* editor is a subset of the *vi* full screen editor. More detailed information may be obtained by using the *man* command or by referencing Documents CR-7 and CX-7 or section 2.2.2 of CR-2. Major features of *ex* and *ed* include:

1. There is no prompt when using *ed*. The *ex* editor has no prompt in text input mode, but command mode has a colon (:) as prompt. The question mark (?) indicates that the editor has received invalid input. To exit input mode with either editor, type a period as the first character of a line, followed by a carriage return.

2. Line numbering is implicit; when lines are added or deleted, all lines are automatically renumbered.

3. Lines in a file may be accessed by line number, position relative to the current line or by a string of characters.

4 To exit either editor and save your changes, type *w* <*cr*> (<*cr*> stands for a carriage return) followed by *q* <*cr*>. If you type only *q* <*cr*>, the editor queries you with a question mark (?) to see if you really intended to exit without saving your changes. If that is your intention, type *q* <*cr*> again.

2.3.3. emacs

The GNU emacs editor is available on the CONVEX computers. It is described in Document CX-21, CONVEX GNU Emacs Manual. The GNU Emacs editor is also installed on **UXV.** Major features of the *emacs* editor include:

1. It is extendible and can be dynamically changed to suit your needs.

2. It provides a windowing capability.

3. It has two on-line assistance commands *help* and *learn*.

4. It may be used to compile programs and correct errors as they are found.

2.4. File Redirection

The following are defaults for standard input, output and error for the CRAY and CONVEX FORTRAN compilers:

standard input	(FORTRAN unit number 5)	keyboard or stdin.
standard output	(FORTRAN unit number 6)	screen or stdout.
standard error	(FORTRAN unit number 0)	screen or stderr.

It is possible to redirect these defaults (See section 3.1 of CR-2). Within the C-shell, the easiest way is to use the left (<) and right (>) chevrons with specified files at execution time. The file name following the left chevron (<) will be the file read by FORTRAN unit 5. The file name following the right chevron (>) will be the file written to by FORTRAN unit 6 and PRINT statements. The file name following the right chevron and an ampersand (>&) will be the file written to by FORTRAN unit 6, PRINT statements and any run time errors. **If unit 6 appears in an OPEN statement within the FORTRAN program, then output written to unit 6 will be on the file specified in the OPEN statement. It cannot be changed by file redirection.**

The syntax to run executable code is :

prog [<inputfn] [>outputfn]

where *prog* is the executable file generated by the compiler. FORTRAN reads from unit number 5 will be read from file *inputfn*. FORTRAN writes to unit 6 and PRINT will be written to file *outputfn*.

prog < dataruna

FORTRAN unit 5 is read from file *dataruna*.

FORTRAN unit 6, PRINT statements and run time errors are written to the terminal screen or stdout.

prog > outrunc

FORTRAN unit 6 and PRINT statements are written to file *outrunc*.

FORTRAN unit 5 is read from the terminal keyboard or stdin. Run time errors are written to the terminal screen or stdout.

prog < datarunx > & outrunx

FORTRAN unit 5 is read from file *datarunx*.

FORTRAN unit 6, PRINT and run time errors are written to file *outrunx*.

3. SPECIAL PURPOSE UTILITIES

UNICOS has many utilities that you will discover as you become more familiar with the system. Novices find that electronic mail and pipes and filters are useful from the very start, so they are introduced in the next two sections.

3.1. Electronic Mail

The electronic mail utilities, *mail* and *mailx,* allow you to send and receive messages from other users on your machine as well as users of any other machine accessible by the local networks (See section 3.5.1 of CR-2). The *mailx* utility is a UNICOS implementation that is similar to the utility *mail* under CONVEX UNIX. UNICOS also has a *mail* utility, but it doesn't have as many useful features as *mailx*. The sample *.cshrc* file in section 4.1.5 of CR-1 aliases *mailx* to *mail* to insure that the more versatile version of mail is always used. The file *.mailrc* (see section 3.1.4) is a start-up file for *mailx*. It is used to initialize your environment variables for electronic mail. For the remainder of this chapter, *mail* is used for both *mailx* and *mail.*

3.1.1. Reading Mail

First of all, how do you know that you have any mail ? Depending on how your environment is defined, the system may send you the message

you have mail

when you login. To read mail, type the command *mail.* Your mail is displayed one message at a time. Only one screen full of long messages is displayed, until you press the carriage return key. At this point you are in the command mode for *mail.* After each message, the prompt, *?,* appears and the systems waits for your response. If you try to read mail and have none, the response

no mail for login

appears, where **login** stands for your login name. Some useful *mail* commands are:

q	Quit, preserving all messages in the file *mbox.*
d	Delete the message just read and read the next message if *autoprint* (see section 3.1.4) is set.
h	List headers for all mail messages.
s file_name	Save messages in file called *file_name.*
R	Send a response to the author of the message.
r	Send a response to all addressees of the message.

3.1.2. Sending Mail

To send mail type

mail login@host

where *login* is the login identifier of the individual(s) who will get your mail message and *host* is the identifier of the machine where that login name is valid, if different from the machine that you are using. While sending electronic mail you are in the input mode of *mail.*

The *mail* utility prompts you for a subject before you type your message. By default, you are not prompted for carbon copy addressees after you finish typing your message, unless you have set the environment variable *askcc* (see section 3.1.4). When you are finished typing your message, type a period (.) in the first column of the new line to exit *mail.* To abort a mail message use two ^C (Control C) signals.

There are commands called tilde (~) commands that can be used only while sending mail. Some useful tilde commands are:

~c names	Add names to the carbon copy (Cc) list.
~v	Invoke the full screen editor specified by the VISUAL environment variable (see section 4.1.4).
~r file	Send the file as the message, or add the file to the message.
~?	Help for the tilde " ~ " commands.

3.1.3. Forwarding Mail

To have your mail forwarded to another machine, create a file called *.forward* in your home directory. If your login name is *john* and you want your mail forwarded to **Eagle** then your *.forward* file contains the following line:

john@eagle

3.1.4. The .mailrc File and Aliases for Mail

Electronic mail aliases are names that specify a group of individuals as recipients of mail. Using *alias* is easier than typing all the names in the group that you frequently need to contact. For example, if you want to set up an *alias* called *fab4* for the members of your group, who have login names *john, paul, george* and *ringo* then you need to add the following line to your *.mailrc* file to establish the alias:

alias fab4 "john paul george ringo"

Thereafter, you will be able to send mail to the group by typing:

mail fab4

Your *.mailrc* file can contain other commands and may be used to set environment variables for *mail* as shown in this sample *.mailrc* file:

```
set askcc                          # Ask for carbon copy prompt.
set autoprint                      # Automatically read next message
                                   #  after deleting a message.
set crt=25                         # Set the number of lines per screen.
                                   #  Longer messages will be piped
                                   #  through pg or more.
alias sag "howser overman tennille"  # Establish an alias
```

3.1.5. Reading Saved Mail

If you exit the *mail* utility by typing "q", then your mail messages that you did not explicitly delete will be saved in a file called *mbox*. To read saved mail type

mail -f

and the system responds with the path name to your *mbox* file and the headers of messages it contains. To read any particular message, type the message number. If you have no saved messages, the system replies that it can't find your *mbox* file. If you have saved mail in a file called *mymail,* then the command

mail -f mymail

will list the message headers for all mail saved in *mymail,* and you can read any message by typing it's number.

3.2. Pipes and Filters

In UNIX, it is often convenient to have the standard output of one command become the standard input of another command. One way of doing this is to use file redirection and temporary files, however this method is not efficient. To solve this problem, UNIX supports the concept of pipes and filters. The pipe symbol is " | " (it appears in the upper right hand corner on most keyboards as two short vertical lines). The ampersand (&) can be used with the pipe symbol to send standard error on to the next command, just as it is with file redirection. Any command which accepts its input from standard input and produces its output on standard output is called a filter. The *more* and *grep* filters are described in the next two sections.

3.2.1. more

The *more* command takes the standard output of another command and outputs one screen full of information at a time. Otherwise the output may scroll by too fast to read it. Once you have a number of files, one use of *more* might be

ls -al | more

to see detailed information on all your files. The *more* command may also be used like the *cat* command (see section 2.2.7.3), except that the file is listed only one screen full at a time.

3.2.2. grep

The *grep* command (See chapter 2 of CR-2) searches the standard input file for a specified character string. For example, if you wanted to see if *john* was logged on to **Voyager,** you could type *who | more* and search for *john* yourself. However, it would be more efficient to let the system do the search for you by typing:

who | grep john

Then if *john* is on the system, you only have one line of output at which to look. If *john* is not on the system there is no output, and the system prompt reappears. On the CRAYs, for example, you could search for your jobs in one of the batch NQS queues with

qstat queue_name | grep login

where *queue_name* is one of the NQS queues defined in section 4.3.1 of CR-1 and *login* is your login identifier.

3.3. make Utility

The *make* utility provides a mechanism for maintaining up-to-date versions of large programs without the necessity of recompiling the entire program if just a few routines are modified. More than one compiler may be used to generate the *.o* files for input to the loader. The *make* utility uses a descriptive file, called *Makefile* or *makefile,* to direct the compilation, assembly and loading of a program from its various components.

Using *make* requires that each subroutine (or group of closely related routines) be maintained as separate files under a directory. If your entire source is on one *.f* file, use the *fsplit* utility to split the large source code into separate *.f* files, one for each subroutine. This may seem cumbersome if you are not familiar with UNIX; however, in addition to facilitating the use of *make,* this practice also makes the use of the various editors more efficient. The *make* utility checks the date of modification of each *.f* file with the date of creation of the corresponding *.o* file. If the *.f* file has not been modified since the *.o* file was created, then the routine is not recompiled. This feature can save you significant CPU time with the global optimizing *cft77* and *fc* compilers.

The *make* utility is documented on-line, in Document CX-22, in chapter 2 of CR-9, *UNICOS Support Tools Guide,* and in CR-6.

The syntax of the *make* utility is

make [-f make_file] [option] [targetfile]

where the *-f make_file* parameter is used to specify a name for the *makefile* other than *makefile* or *Makefile*. The *targetfile* is the file that *make* generates. See the *man* pages for *make* for a description of the various parameters that can be specified by *option*. The next page contains an example of a simple *makefile* to be executed on **Voyager** or **Sabre.**

UNICOS has a command *fmgen,* which splits a file containing FORTRAN source code into individual files and creates the makefile to compile and load the program automatically. You can select the compiler to be used, but an editor must be used to change the loader to *segldr* rather than *ldr*. The syntax of *fmgen* is:

fmgen [-m makefile] [-c compiler] [-f flags] [-o command] source.f

The *fmgen* utility allows you to name the *makefile,* choose the compiler, select compiler options and choose a name for the executable code. The file *source.f* contains the original FORTRAN source code.

Example:

Assume that your FORTRAN source code contains the modules main, sub1, sub2 and fn1 on file *source.f*. To create your *makefile* without using *fmgen*, use

fsplit source.f

to create the files *main.f, sub1.f, sub2.f* and *fn1.f* in a directory by themselves. Then create a file called *makefile* in the same directory that has the following entries:

```
OBJECTS = main.o sub1.o sub2.o fn1.o
CFTFLAGS = -es
#
runit: $(OBJECTS)
 <Tab> cf77 $(CFTFLAGS) -o runit $(OBJECTS)
.f.o:
 <Tab> cft77 $(CFTFLAGS) $*.f
clean:
 <Tab> rm -f $(OBJECTS)
```

The *make* utility runs in the shell specified by the *SHELL* environment variable. The default for most users is the C-shell. The string " *<Tab>* " in this example represents a tab character that must be present. It is not sufficient to space over a few characters. If the list of object modules (specified by *OBJECTS* =) needs to be extended over more than one line, end the line with a back slash (\). Once the *makefile* is created, the command

make -n

outputs but does not execute the commands *make* would execute. The command

make runit

using the example above would generate the executable file *runit*. The *make* utility must be executed from within the directory containing the *makefile* and all the *.f* source code files.

3.4 The X Window System [a]

The X Window System is a network-transparent graphical window-based software system that was developed at MIT in 1984. The architecture of the X Window System is based on a client-server model. Display servers are programs residing on workstations, X terminals, and PCs, which provide display capabilities and manage user input devices (keyboard, mouse, etc.). Clients are application programs that perform specific tasks and may be run on workstations, PCs or **SNS** computers. The discussion presented here covers the client side of the X Window System as it pertains to the **SNS** computers. For further information, please consult *The Definitive Guides to the X Window System* Volumes 0-7, published by O'Reilly and Associates. In particular, Volume 3 is a user's guide to the X Window System. These manuals are available in the NASA Technical Library.

The X Window System provided on the **SNS** computers is Version 11, Release 4 (X11R4). On the CONVEX computers, the X Window product is called CXwindows. It consists of client programs, include files and libraries. The client programs reside in the */usr/bin/X11* directory; the include files reside in the */usr/include/X11* directory; and the libraries reside in the */usr/lib/X11* directory. The *man* pages for the client programs are available interactively on the system. The topic *notes Xinfo* provides additional information. The libraries available are as follows:

libX11.a	*Xlib library*
libXaw.a	*Athena widget set library*
libXt.a	*Xt Intrinsics library*
libXmu.a	*Miscellaneous utilities library*

To begin using the X Window System on your workstation, you must first start the X server, at least one *xterm* emulator, and a window manager, such as *uwm*. Some of this setup may be done automatically for you. See your workstation's System Administrator for details.

Before using an application with an X Window interface such as *cdbx* on **Voyager** or **Sabre,** or the graphics package *unimap* on **Eagle,** you must identify the host to the server because the application resides on a host remote to your workstation. Host identification is accomplished by invoking the *xhost* client program on your workstation, as illustrated:

xhost +voyager

This allows any X application running on **Voyager** to open a window on your display. You may also remove a host from the host list in the following manner:

xhost -eagle

To customize your X environment, you may choose to use the X initialization script called *xinitrc,* which is illustrated in the *man* page for the X client *xinit.* A commonly used client program that is usually initiated from with the *xinitrc* script is the clock program, *xclock,* which displays a clock in a small window that you can place anywhere on your display screen.

There are two other minor setup duties that you should perform to use the X Window System on the **SNS** computers. These setup procedures are not mandatory, but can be helpful. First, in order to execute the client programs without explicitly typing the entire path to the command, you need to put the directory, */usr/bin/X11* into your default path. See Section 2.2.6 of this manual and Section 4.1.4 of CX-1 and CR-1 for information on adding directories to your current path. Second, you need to define where the windows will be displayed (i.e. identify the server). This is done by setting an environment variable, *DISPLAY,* to the network name (or internet address) of your display monitor, as shown below.

setenv DISPLAY myxterm:0.0

The display name is of the form:

host:display.screen

where

host	is the name of the machine to which the display is physically connected.
display	is the display number (usually 0), where display usually refers to a collection of displays that share a common keyboard and mouse.
screen	is the screen number (also usually 0). Multiple screens may also share a single keyboard and mouse.

See Sections 4.1.2 and 4.1.3 of this manual for more information on setting environment variables.

Setting of these variables is not necessary and can be accomplished in other ways. First, you may explicitly define the full path every time you execute an X client program. Second, you may execute an X client program using the *-display* option to define where the output should be displayed. For example, if you would like to run an *xterm* (character-based terminal emulator) on **Voyager** and display the window on the display named *myxterm,* you would type the following command:

/usr/bin/X11/xterm -display myxterm:0.0

If you have set your path to include the */usr/bin/X11* directory, you may eliminate the full path specification to the command as follows:

xterm -display myxterm:0.0

If you have also set your environment variable, *DISPLAY,* you may eliminate the use of the *-display* option as follows:

xterm

Refer to the O'Reilly manuals for a complete discussion on the various client programs.

There are specialized client programs called window managers which manage the display space and allow you to manipulate windows (by moving, resizing, or iconifying). One window manager, *uwm,* is supported on **Voyager** and **Sabre.** Two window managers, *twm* and *mwm,* are supported on **Eagle** and **Mustang.** These window managers are located in the same directory as other client programs, */usr/bin/X11* and are executed in the same way. However, it is in your best interest to run the window manager from your own workstation, rather than on one of the **SNS** computers. CRU's accumulate with every mouse movement!

4. FILE MANAGEMENT

Historically, it has been true that the user community desires, and could in fact make use of, more disk storage than system managers can feasibly provide. This is the case for all the **SNS** computers. Thus, one of the biggest challenges for both **SNS** users and managers is to effectively manage the disk space that is available. Most users should include communication with the Mass Storage Subsystem (See section 5.5) into their overall file management philosophy.

The management of the disk resource is the responsibility of the Computer Management Branch (CMB) of ACD. CMB manages the overall disk resource with the goals of providing users with adequate space to perform their research and backup security for permanent files. The current procedures used in system file management can be reviewed with the *notes* utility (See section 8.3) by typing *notes CRadmin* or *notes CXadmin* on **Eagle** or **Mustang.**

4.1. Use of Permanent and Scratch File Space

You may work from within your own home directory or from within a shared scratch directory. The Network File System, NFS, is not utilized between **Voyager** and **Sabre** as it is between **Eagle** and **Mustang.** At login you are placed in your home directory. All files created in or copied into that directory or any of its subdirectories are permanent and backed up by the system on a periodic basis. Each user has been allocated a fixed amount of permanent file space. That "hard" quota (See section 1.3.2) cannot be exceeded. The scratch directory, designated */scr,* is a large, global area of disk available to and shared by all users of the system. It has no individual quotas. Files existing in this directory are not backed up and have a limited lifetime. If a file has not been accessed in an established period of time, it is removed by the system. At the printing of this manual the period is set to 72 hours. Use the commands *notes CRadmin* and *notes CXadmin* to obtain current information about this and other system management procedures.

Because UNIX commands and utilities frequently create files in the directory from which they were executed, and because many user program and data files are large due to the types of problems we solve at Langley, it is not possible to allocate you enough file space to allow you to execute normally from within your home directory. You are expected to work from within the larger, but less permanent, */scr* directory. It is suggested that you create for yourself a subdirectory within */scr* as illustrated for user *john:*

mkdir /scr/john

User *john* changes to */scr/john* to perform normal activities. Your home directory should be used for files such as source, script, and small data files. Files created in the scratch directory that must be kept may be copied into the home directory if space is available or moved to mass storage or another system with sufficient file space.

The */scr* directories are independent among all the computers, including **Mustang** and **Eagle.** They are not crossmounted with NFS.

The scratch directory is shared by everyone using the system and, although large, it benefits everyone if you remove files when you no longer need them. This lessens the probability of a job aborting due to insufficient scratch space.

4.2. Disposition of Print Files

The **SNS** configuration includes a path on LaRCNET to the central site high speed laser printer, which has the default name **ibmlaser.** This provides for the easy disposition of print files from **Voyager.** The **ibmlaser** printer prints up to 168 columns at 48 lines per page. The paper is eight and one-half by fourteen inches fan-fold. It also has an alternate name, **ibmlaserp,** which prints up to 80 columns at 66 lines per page rotated on the same paper To obtain printed output you may use the *lpr* command as follows:

lpr -C delivery_info -Jgroup_id -Pprinter_id file_name

If you have established default delivery information with the *DELIVER* environment variable (as illustrated in the sample *.login* file in section 4.1.4), then you need only specify the *file_name,* and the system will use other defaults. By default output is routed to **ibmlaser** and your default account is used for *group_id* (See section 1.3.1). If you use the *-C* option, then the first eight characters of your *delivery_info* should include a bin number or building. Only the first eight characters of *delivery_info* appear as a banner, so that information should be sufficient to insure delivery to you. The remaining characters are printed on the banner page. **If there are blanks in** *delivery_info,* **then the entire string must be surrounded by double quotes** (" ").

A simple script to print a file on the default printer of a remote TCP/IP computer is:

cat print_file | remsh remote_host lpr -C delivery_info

This script requires that you have your *.rhosts* files properly set up on both the local and remote hosts.

You may set the environment variable LPP to 48 (See section 4.1.4) in your *.login* file to force listings from *segldr* or the FORTRAN compilers to paginate correctly for the paper used by the central site printer *ibmlaser.*

4.2.1 PostScript Printing

There is a laser postscript printer called *psjet+* accessible from **Eagle** and **Mustang.** It uses standard eight-and-one-half by eleven inch paper and can print up to eight pages per minute. Users are requested to keep print jobs under 50 pages or under 2 megabytes, since *psjet+* is a relatively slow printer. The Adobe software package *transcript* (which includes *ptroff, enscript* and other utilities) is installed on **Eagle** and **Mustang** as well. Use *man transcript* for more information. Document CX-24 discusses the *nroff* and *troff* text formatting utilities. The various macros such as *-ms, -me* and *-man* are described in CX-9 and on *man* pages.

4.3. Network File System [a]

The Sun Microsystems Network File System (NFS) is utilized primarily between **Eagle** and **Mustang,** and is described in Document CX-14. You are allocated permanent file space on a disk physically connected to **Eagle.** When you login to **Mustang,** your file system from **Eagle** is mounted via NFS as your home directory for **Mustang.** From your perspective, you only have to manage that single file space, and it seems to be local to **Mustang.** However, the efficiency of using NFS mounted is enhanced when you are logged into **Eagle,** where your permanent files physically reside.

NFS on the CRAY supercomputers is used to support the Mass Storage Subsystem. User files are not crossmounted between **Voyager** and **Sabre.**

5. REMOTE LOGIN AND FILE TRANSFER UTILITIES

TCP/IP (Transmission Control Protocol/Internet Protocol) is a set of computer networking protocols that allow two or more computers (called hosts) to communicate with one another. There are utilities *(telnet* and *rlogin)* that allow you to login to other hosts on your network just as if they were connected directly to your terminal. Other utilities *(ftp* and *rcp)* provide efficient transfer of files between hosts on the same network. The utilities *remsh* (on the CRAY computers) and *rsh* (on the CONVEX computers) provide the capability to execute a command on a remote machine for which you could use *rlogin.* These five utilities are described in the *CONVEX Networking Utilities User's Guide* (CX-12), the *CRAY Networking Utilities User's Guide* (CR-11) and Appendix A of CR-2. A brief description of each follows this section.

In addition utilities that allow you to archive and retrieve files from any of the **SNS** computers to the Mass Storage Subsystem (MSS) have been developed at LaRC (See section 5.5).

5.1. telnet

The *telnet* utility provides a virtual terminal that appears to be directly connected to the remote host. You have full user capabilities and privileges on the remote host until you choose to break the connection. To access a remote host with *telnet* type:

telnet remote_host

where *remote_host* is the name (such as **Voyager, Sabre, Eagle** or **Mustang).** When you use *telnet,* you are always prompted for your login name and password. Machines that you can access via *telnet* are listed in the */etc/hosts* file.

5.2. rlogin, rsh and remsh

The *rlogin* command is similar to *telnet* in that it provides a virtual terminal with full user capabilities. The major additional feature of *rlogin* is that it allows automatic login to remote hosts. You are not prompted for your login name when you remotely login to machines for which you have valid accounts. If you have set up a file called *.rhosts,* which defines all of your accounts, then you are not prompted for a password. It is recommended that your *.rhosts* files have the permissions set to "rwx------" on. Your *.rhosts* files should have as few entries as practical. To use the Mass Storage Subsystem (MSS) from **Voyager, Eagle** and **Mustang,** you must have a *.rhosts* file on **Sabre,** as discussed in section 5.5. A typical *.rhosts* file for use with the **SNS** computers might have the following entries:

eagle.larc.nasa.gov login_name
voyager.larc.nasa.gov login_name
voyager-b.larc.nasa.gov login_name
voyager-uif.larc.nasa.gov login_name
sabre.larc.nasa.gov login_name
sabre-b.larc.nasa.gov login_name
sabre-uif.larc.nasa.gov login_name

The reason for having three entries for **Voyager** and **Sabre** is that each machine has two connections to the HYPERchannel (i.e. *sabre.larc.nasa.gov* and *sabre-b.larc.nasa.gov* for the CRAY Y-MP) and one connection to the UltraNet (i.e. *voyager-uif.larc.nasa.gov* for the CRAY-2), each with a different internet address. Unless both HYPERchannel connections are specified in the *.rhosts* files, the commands *rlogin, rcp, rsh* and *remsh* may fail or not work as expected. Your *login_name* is specified to insure that you won't be prompted for a password. The UltraNet connections need to be specified so that the Mass Storage Utilities (See section 5.5) work correctly.

Additionally *rlogin* passes on your terminal type to the remote host, unless overwritten by your *.login* file on the remote host. To access another host via *rlogin,* type

rlogin remote_host

where *remote_host* is the destination machine. The */etc/hosts* file contains the valid host names to which you can login.

The *remsh* utility on the CRAY computers is functionally the same as the *rsh* utility on the CONVEX computers. Both allow you to execute a single command on a remote machine. The syntax is:

[remsh] or [rsh] remote_host command

The *remote_host* is defined as it was for *rlogin* and the *command* may have arguments such as:

remsh eagle ls -al

Without a command, the *remsh* and *rsh* utilities function like *rlogin.*

5.3. ftp

The utility *ftp* (File Transfer Protocol) is a communications utility which utilizes the TCP/IP protocol to access and copy files between networked computer systems. There is a *help* command on-line with *ftp.* Detailed information about ftp may be found in CR-11 and CX-12.

If you are logged into **Voyager** and wish to transfer files to or from **Eagle,** then type

ftp eagle

and the system responds with prompts for your login name and password for **Eagle.** If you do not wish to be prompted for your login name and password every time that you use *ftp* between two machines, then you may create a file called *.netrc,* which contains validation information for the machines between which you frequently exchange files. The file *.netrc* must have all file permissions turned off for your group and others and all of your permissions turned on. After creating *.netrc* with an editor use the command

chmod 700 .netrc

to properly establish the permissions (See section 2.2.8). If the file access permissions are set incorrectly, then *ftp* does not work. A sample *.netrc* file for user *john* might contain the following entries:

machine *eagle.larc.nasa.gov* **login** *john* **password** *sm1thy*
machine *z.larc.nasa.gov* **login** *123456n* **password** *sm1thy*
machine *voyager.larc.nasa.gov* **login** *john* **password** *sm1thy*
machine *sabre.larc.nasa.gov* **login** *john* **password** *sm1thy*

The entries in **boldface** are required keywords. You do not need entires for both **Eagle** and **Mustang** since their file systems are cross mounted under NFS. See Document N2-46 for information relative to establishing a file similar to *.netrc* on the NOS computers. It is recommended that the *.netrc* file contain as few entries as practical.

Once you are properly validated for the remote machine, the prompt *ftp>* will appear and you can enter any *ftp* command. Some common commands are:

get remote_file
put local_file
mget remote_files
mput local_files

These commands are used to transfer one *(get* and *put)* or multiple files *(mget* and *mput)* between the two computers. The *get* and *mget* commands fetch the requested files from the remote host. The *put* and *mput* commands transfer a local file to the remote host. You can also use the UNIX-like *ftp* commands *ls* and *cd* on the remote host. To exit *ftp,* type *quit* or *bye* and your connection is closed. Documents CR-11 and CX-12 describe all the *ftp* commands.

5.4. rcp

The *rcp* command may be used instead of the *ftp* utility to transfer files between networked computers. The syntax is:

rcp remote_host1:filex remote_host2:filey

If you are logged onto *remote_host1* or *remote_host2,* that computer does not need to be specified in the *rcp* command. Both file names may be specified by path names (See section 2.2.6). Your *.rhosts* file must be properly established on all the *remote_host* machines involved in the *rcp* (See section 5.2).

5.5. Mass Storage Subsystem [a]

The Mass Storage Subsystem consists of the CRAY Y-MP, **Sabre,** 2 StorageTek STK 4400 tape silos and a Sun Microsystems workstation, **Winnie Mae.** The Sun workstation manages the database of file locations on the STK cartridge tapes in the silos.

The movement of files between computers on the high-speed network and MSS is accomplished with UNIX utilities developed within ACD. The existing commands are: *masput, masget, masmkdir, masrm, masls, masmv, maschmod, maschgrp, masrmdir,* and *maspwd.* The *maspwd* utility is new and several of the other commands have had new options added. The symbol "@" was chosen to represent the path to a user's home directory on MSS in the same sense that "~" provides a shorthand notation for UNIX path names (See section 2.2.6).

You must create a *.rhosts* file on Sabre to use the MSS utilities from Voyager, Eagle and Mustang, since the MSS utilities require the remote execution of commands on Sabre. The following lines should be added (replacing *login_name* with your login id) to your *.rhosts* file (see section 5.2), to use the MSS utilities from any **SNS** computer other than **Sabre.**

eagle.larc.nasa.gov login_name
voyager.larc.nasa.gov login_name
voyager-b.larc.nasa.gov login_name
voyager-uif.larc.nasa.gov login_name
sabre.larc.nasa.gov login_name
sabre-b.larc.nasa.gov login_name
sabre-uif.larc.nasa.gov login_name

Metacharacters may now be used when referring to MSS files. When they are used, metacharacters must be preceded by a backslash "\". For example, to list all MSS files with names that begin with the letter "c", you could use the following:

*masls c**

You may also access another user's MSS files if the correct permissions are set on the files you wish to transfer.

5.5.1. masput

The *masput* utility puts one or more files onto MSS. The syntax is:

masput [-r] fl [... fn] [dest]

A list of file(s) (*fl* through *fn*) must be specified with either a full path name or one relative to the current working directory. On MSS, the destination, *dest,* may be a file name if only one file is being placed on mass storage, but must be a directory if more than one file is being sent out to MSS. If more than one file name is specified and *dest* is not a valid directory on MSS, an error message is returned. The *-r* option causes a directory and all subdirectories to be copied the the specified MSS destination, *dest.* All necessary subdirectories are created on MSS.

To archive the file named *results* to mass storage:

masput results @

The file *results* remains on the SNS file system. Since only one file is being archived in this example, it is not necessary to specify the MSS home directory explicitly. The command

masput results

would have the same effect. To archive the files *results1* and *results2* to mass storage under the directory *Cases.*

masput results1 results2 @/Cases

The destination directory could also be specified by *Cases.* The directory, *Cases,* must exist on mass storage. To archive *run.out* to MSS and rename it *results*

masput run.out results

where *results* is a file in your mass storage home directory, which may or may not have previously existed. If *results* is the name of an existing file, it is overwritten. The destination file could also be designated by *@/results.*

To archive your directory */scr/xyz/test* and all its subdirectories to your MSS home directory:

masput -r /src/xyz/test @

5.5.2. masget

The *masget* utility retrieves files from MSS. The syntax is:

masget [-r] fl [... fn] dest

One or more files (*fl* through *fn*) may be retrieved from mass storage together. If the *-r* option is specified, an entire MSS directory and all it's subdirectories are copied to the destination, *dest*. All necessary subdirectories are created on the local machine. If only one file is retrieved, then *dest* may be a file name, otherwise, it must be a valid directory name on the **SNS** file system. To retrieve the file *results* from mass storage to your current working directory:

masget results .

To retrieve your MSS directory *Cases* and all subdirectories to your current working directory:

masget -r Cases .

The file *results* could be renamed *oldresults* in your current working directory with the following:

masget results oldresults

To retrieve all of the *.f* files in your MSS directory *sources* and place them in your current working directory:

masget source/.f .*

You can also retrieve files from another users mass storage directory, provided that you have read permission. To get the file *results* from the home directory of user *john* and rename it *res.john* under your subdirectory *Cases* in your current working directory:

masget @john/results Cases/res.john

5.5.3. masmkdir

The *masmkdir* utility creates a mass storage directory for you. By default you have a home directory on mass storage which has the name @ or *@login,* where @ is equivalent to ~ in the *csh.* To create a subdirectory named *Cases* under your MSS home directory:

masmkdir @/Cases

The directory created is always assumed to be relative to your current working directory on MSS.

5.5.4. masrm

The *masrm* utility removes a designated file or files from MSS. The syntax is:

masrm [-ir] fl [... fn]

The *-r* option removes all files, recursively, from the specified directory down, including all subdirectories. This option should be used with caution. The *-i* option causes a prompt to appear that asks you to confirm that you really want to remove the file(s). The *-i* option is ignored on all machine except **Sabre.** To remove the files *source* and *results1* (which resides in the subdirectory *Cases*) from mass storage:

masrm source Cases/results1

5.5.5. masls

The *masls* utility lists files that you have stored on MSS. Without any arguments, *masls* lists your mass storage home directory. It has several options that correspond to the options for the UNIX utility *ls.* These options are described in the *man* page for *masls.* For some frequently used options, the syntax is:

masls [-lRF] [fl ... fn]

The *l* option gives a long listing including file sizes and permissions. By default, only the file names are listed. The *R* option lists all files at or below the specified directory. A list of files and/or directories (*fl* through *fn*) to be listed can also be provided. The *-F* option lists MSS directories with a backslash '/' appended to their names. To list all the files of user *john,* along with their permissions:

masls -lR @john

To list all of your files with names that begin with the string *case1* :

*masls case1**

5.5.6. masmv

The *masmv* utility moves a designated MSS file or files to a different MSS directory. It is also used to rename a MSS file. The syntax is

masmv f1 [... fn] dest

where the last name specified is always the destination. If there are three or more names as arguments, the last name must be a MSS directory name. If only two names are given and the last is not a directory, then the file is renamed. If the second name is an existing file, that file is overwritten. If you have files called *results1* and *results2* in your home directory on MSS and create a MSS directory called *Cases,* then the command

masmv results1 results2 Cases

moves the two files into the directory *Cases.*

5.5.7. maschmod

The *maschmod* utility is used to change the access permissions of your MSS files and directories. The syntax is

maschmod mode f1 [f2 ... fn]

where *mode* is either a three digit octal number or a symbolic representation. The *mode* is defined in the same manner as *chmod* command (See section 2.2.8). For example,

maschmod 440 results source

allows you and any member of your group to read the files *results* and *source* from MSS. No one else can access those files. If the string *440* is replaced by *600* then you can read or write those files on MSS but no one else has any access. Then to grant members of your group permission to read those files, you could use

maschmod g+r results source

to add group read permission.

5.5.8. maschgrp

The *maschgrp* utility changes the group to which your MSS files and directories belong. The syntax is

maschgrp group_id fl [... fn]

where *group_id* is one of your valid **SNS** accounts (i.e. one beginning with the letter "a" as described in section 1.3.1).

5.5.9. masrmdir

The *masrmdir* utility is used to delete directories from MSS. The directory to be removed must be empty. To remove all the files and the directory use the *masrm -r* command (See section 5.5.4). The syntax is:

masrmdir directory_name(s)

The utility issues a warning if any directories are not empty and continues processing.

5.5.10. maspwd

The *maspwd* utility echos your MSS current working directory as supplied by the *MASCD* shell variable. Use the *maspwd* utility to verify your MSS current working directory before using *masput* and *masrm,* especially when using metacharacters.

5.5.11. MASCD [a]

The shell variable *MASCD* is used to change your current MSS working directory. It is used by all MSS commands. A reference to any MSS file that does not begin with the "@" symbol is relative to your current MSS working directory. If you have not set the *MASCD* variable, all references are considered to be relative to your MSS home directory "@". Any absolute reference to MSS files ignores the *MASCD* variable. To set this variable in the C-shell:

setenv MASCD mss_directory

The *mss_directory* specified must begin with "@" or *@login_name/*. Caution is advised when using the *MASCD* variable. The *maspwd* utility (See section 5.5.10) prints the current MSS working directory. You should use it to avoid placing files in the wrong directory or removing the wrong files.

6. GRAPHICS ON THE SNS COMPUTERS [a]

A variety of commercial and locally developed graphics software packages are installed on the **SNS** computers. The intent of this section is to provide a brief description of each package and to suggest sources for additional documentation, site specific usage information, and user support.

Additional documentation is available for individual graphics packages from the Operations Control Office (OCO) at 864-6562. All supported packages provide *man* pages for package specific access information such as required libraries and supported device drivers. All graphics package are not installed on all **SNS** computers. The *notes graphics* category on **Eagle** and **Mustang** contains the latest information on the status of graphics installation. Questions or problems concerning the functionality or use of supported graphics software or device interfaces should be directed to 864-8532 or via e-mail to *grafhelp@eagle*.

6.1. Precision Visuals Graphics Software

The PVI graphics software is an integrated system of graphics software tools, marketed by Precision Visuals, Inc. of Boulder, Colorado. It has packages to do presentation graphics, scientific charts, contour plots, and line art drawings - all of which invoke the device independent DI-3000 graphics library. The PVI software is not available on **Sabre.** The graphics tools communicate through the PVI Metafile System which stores pictures in a device-independent picture file. The DI-3000 metafile can be post-processed to obtain publication quality charts from the ACD production plotters.

The *di3load* command is used to compile and load a DI-3000 application program using the following syntax:

di3load -SS -MF -TP -CN -CGL program.f

A previously compiled code may be loaded by replacing *program.f* with *program.o*. The other options have the following meanings:

SS	- Use the Segment Storage option.
MF	- Use the MetaFile option.
TP	- Use the DiTextPro option.
CN	- Use the Contouring option.
CGL	- Use the Common Graphics Library.

Several environment variables must be preset prior to invoking *di3load* Check the *man* pages for *di3load* for definition of those variables.

6.1.1. DI-3000

DI-3000 is a modular 2-D/3-D library of over 200 FORTRAN-callable subroutines which provide color, drawing primitives, viewing, full graphics input, and a graphics data structure. It is device independent and provides support for a variety of graphics applications. It is highly interactive and has device drivers for most of the graphics devices in the current graphics environment. The *DI-3000 User's Guide* (Document G-5) and *DI-3000 Quick Reference Guide* are available from OCO.

6.1.2. Extended Metafile System

The Extended Metafile System is an interactive software system for the storage, retrieval, manipulation, and transport of graphical images for viewing and hardcopy. The system includes support for both the Precision Visuals proprietary metafile format and the ANSI and ISO approved CGM (Computer Graphics Metafile) standard format. The *Metafile Translator User's Guide* (Document G-6) is available from OCO. Additional Metafile System access information may be obtained on-line with the *man mftran* command.

6.1.3. DI-TEXTPRO

DI-TEXTPRO is a DI-3000 text option for providing high-quality polygonal text. It features twelve typefaces, italic for all typefaces, an outline/solid fill option, and full support for superscripting, subscripting, and underlining. The DI-TEXTPRO documentation is Appendix D of the *DI-3000 User's Guide* and is available from OCO.

6.1.4. Contouring System

The Contouring System is a library of FORTRAN-callable subroutines which provide contouring and grid generation capabilities. It features include a 2-D "quick-look" map with linearly interpolated contour lines, a 2-D map with smooth contour lines, and a 3-D surface map with hidden lines removed. The *Contouring System User's Guide* (Document G-8) is available from OCO.

6.1.5. PVI Device Drivers

The PVI device drivers interface DI-3000 and the high-level PVI graphics software to the actual graphics terminals, workstations, and peripherals. Each PVI device driver is documented by a Device Driver Guide which explains the capabilities of the hardware and the extent of support in the device driver. The Device Driver Guides are available from OCO.

6.1.6. ACD Graphics Production Device Interface

The ACD Graphics Production Devices are maintained and operated for the LaRC user by the Analysis and Computation Division (ACD). The hardware is accessed through either a vector or raster graphics device filter. The generic vector filter is called *mfdev* and the generic raster filter is *rmdev*. The three characters *dev* specify the particular filter (See **Table 6.1**). The filter *mfdev* is used to port DI-3000 metafiles to vector devices and *rmdev* is used to port raster data to raster devices in RM (See section 6.4) format. The device driver guides are available from OCO. Additional information may be obtained on-line with either the *man mfdev* or *man rmdev* commands.

6.2. Common Graphics Library

The Common Graphics Library (CGL) is an application-independent FORTRAN-callable graphics package designed to generate publication and/or viewgraph quality charts and graphs. The library is designed for users with no graphics experience, yet is versatile enough to compose and design detailed charts for publication purposes. The current version of the CGL uses DI-3000 as its underlying graphics package. The *Common Graphics Library (CGL) Volume 1: LEZ User's Guide* (Document G-12) is available from OCO. CGL access information may be obtained on-line with the *man cgl* or *man di3load* (See section 6.1) commands.

6.3. NCAR GRAPHICS [a]

NCAR Graphics ia a collection of FORTRAN 77 programs and subroutines used to generate and plot computer graphics suitable for the display of scientific data. Individual NCAR Graphics utilities can contour data fields on regularly-spaced and irregularly-spaced grids, produce world maps using any of ten projections, display 2-D vector fields, draw 3-D displays of functions of two variables, and draw iso-surfaces from a 3-D array. NCAR Graphics Version 3.00 includes a level 0a GKS package that generates CGM (Computer Graphics Metafile) metacode along with CGM translators and accompanying device drivers.

The NCAR Graphics Version 3.00 documentation includes the *NCAR, Version 3 Reference Manual* (Document G-23) and the *NCAR, Version 3 User's Guide* (Document G-24), which are both available from OCO. NCAR Graphics access information may be obtained on-line with the *man ncargintro* command.

6.4. RM/RMT

In order to provide some degree of compatibility for locally generated raster image data, LaRC has adopted a generic raster format known as Raster Metafile (RM) format. A prototype version of the Raster Metafile Translator (RMT), a command-driven program designed to permit reading/writing, display and manipulation of RM formatted images, is provided.

No formal documentation on the RM format or the RMT program is available at this time. Preliminary information of the RM format is available through OCO. RMT function descriptions are provided by the on-line help command. RMT access information may be obtained on-line with the *man rmtran* command.

6.5. RASLIB

RASLIB is a library of subroutines developed at LaRC to provide a means of generating solid or smooth filled raster images of 2-D or 3-D grids. It can eliminate hidden surfaces, and contains graphics primitives for setting pixels, drawing lines, and filling polygons. Text information may be overlaid in a variety of character sizes and colors. Windowing and clipping are supported. A color look-up table is provided and may be manipulated by the user. The "RASDOC" manual is available from OCO. RASLIB access information may be obtained on-line with the *man raslib* command.

6.6. PLOT3D

PLOT3D allows computational fluid dynamics researchers to interactively view body geometry; computational grids; vortex lines; shock waves; particle paths in a flow; and physical variable such as density, pressure, entropy and momentum components. The user may select wire-frame or shaded-surface displays. Physical variables may be represented as individual contours or as smoothly interpolated shades, where color is used to indicate various levels of a variable's values. Displays can be annotated with titles and color bars. PLOT3X outputs the results of a PLOT3D viewing session in a form that can be input to the Graphics Animation System (GAS) to create an animation sequence that can be videotaped. The PLOT3D display is actually viewed on a Silicon Graphics IRIS workstation using the Remote Graphics Library (RGL).

6.7. GAS

The Graphics Animation System (GAS) executes on a Silicon Graphics IRIS workstation and not on any of the SNS computers. It provides researchers with a fast and simple viewing capability as well as more complex rendering and animation features for PLOT3D graphics files.

6.8. LARCGOS

The LaRC Graphics Output System (LARCGOS) is a library of graphics subroutines oriented toward passive, 2-D graphics output. LARCGOS is available only on **Mustang** and **Eagle.** The library provides the basic graphics primitives, as well as subroutines for generating arcs, circles, and rectangles, and outputting text strings with a variety of fonts. In addition, some higher level capabilities are provided through routines which, for example, scale data, draw a grid, or draw and annotate axes. A publication module allows users to produce report figures suitable for the LaRC publication process. The *Langley Graphics System* (Document G-3) is available from OCO. LARCGOS access information may be obtained on-line with the *man larcgos* command. The LARCGOS library is linked to a user's program on the *fc* command line. The library is located in:

/usr/local/graphics/larcgos/larcgos.a

Running the executable file generated by *fc* creates the NOSPLT plot vector file.

6.9. PICSURE [a]

PicSure is a command driven software tool capable of producing line charts, bar charts, text charts, and pie charts. Its symbol definition and locator input features also facilitate building organization charts and flow diagrams. Other important features include dynamic positioning of chart elements, control of the size and color of chart elements, control of all text attributes, and fast assembly of multicharts into a single image. PicSure uses the DI-3000 library as a kernel and therefore has access to the complete set of DI-3000 device drivers and metafile system .

PicSure is documented in the *PicSure User's Guide* (Document G-11) and the *PicSure Quick Reference Guide* available from OCO. PicSure access information may be obtained on-line with the *man picsure* command.

UNIX Option	Description	cal34	cal11	ver39	vera	verb
-h	height of plot	34	11	39	8	10
-w	width of plot	34	11	39	8	10
-x	x offset	0	0	na	na	na
-y	y offset	0	0	na	na	na
-p	page size	yes	yes	na	na	na
-r	rotate 90 deg	yes	yes	na	na	na
-c	number of copies	na	na	na	na	na
-1	width of pen 1	na	na	2	2	2
-2	width of pen 2	na	na	2	2	2
-3	width of pen 3	na	na	2	2	2
-4	width of pen 4	na	na	2	2	2
-5	width of pen 5	na	na	2	2	2
-6	width of pen 6	na	na	2	2	2
-7	width of pen 7	na	na	2	2	2
-8	width of pen 8	na	na	2	2	2
	max height	34	11	39	8.11	10.66
	max width	1220	1320	156	8.91	14.91

UNIX Option	Description	cel35	cel16	cel8	cel4	ibm3800
-h	height of plot	na	na	na	na	7.5
-w	width of plot	na	na	na	na	7.5
-x	x offset	na	na	na	na	0
-y	y offset	na	na	na	na	0
-p	page size	na	na	na	na	na
-r	rotate 90 deg	na	na	na	na	na
-c	number of copies	1	1	1	1	na
-1	width of pen 1	na	na	na	na	na
-2	width of pen 2	na	na	na	na	na
-3	width of pen 3	na	na	na	na	na
-4	width of pen 4	na	na	na	na	na
-5	width of pen 5	na	na	na	na	na
-6	width of pen 6	na	na	na	na	na
-7	width of pen 7	na	na	na	na	na
-8	width of pen 8	na	na	na	na	na
	max height	na	na	na	na	13.87
	max width	na	na	na	na	7.50

Table 6.1 SNS Options for ACD Production Devices

NOTE: All optional parameters are not applicable to every device. Check the ACD Production Device Driver Guides available from OCO. Numeric values supplied for options are the defaults. The string "na" implies the parameter is not applicable to the device, while the string "yes" implies that the parameter may be set by the user, but no default is supported.

7. MATHEMATICAL LIBRARIES [a]

The International Standard Mathematical Library (IMSL) and the locally developed library LARCLIB are on all the **SNS** computers. In addition the CRAY library LIBSCI and the CONVEX library VECLIB are also available. This section describes the libraries. To link any of these libraries, except LIBSCI, to your program requires that the loader option *-llibname* be added to your *fc* or *cf77* command line, where *libname* could be *imslib, veclib* or *larclib,* depending on the machine and which libraries that you wanted to load. The CRAY LIBSCI is automatically loaded with your FORTRAN program. If multiple libraries are specified, they are searched from left to right to satisfy external references within your code.

There is an important difference in placement of the library loader option between the CRAY and CONVEX compile and load commands. If your program is called *prog.f* and you wish to load the IMSL library, the syntax on the CRAYs is

cf77 -limslib prog.f

while for CONVEX, the syntax is

fc prog.f -limslib

The *CRAY Mathematical Libraries* (CR-3) and *CONVEX Mathematical Libraries* (CX-3) have information specific to the different computers and a list of all the LARCLIB subroutines. Information on LIBSCI and VECLIB is in *CRAY UNICOS Math and Scientific Library Reference Manual* (CR-29), *CONVEX VECLIB User's Guide* (CX-16), *CONVEX EISPACK and LINPACK Subroutines* (CX-26), and *CONVEX LSQPACK User's Guide* (CX-17).

7.1. IMSL [a]

The IMSL is a collection of mathematical and statistical routines written in FORTRAN. These routines can be used to solve linear equations and eigenvalue problems, invert matrices, solve ordinary and partial differential equations and evaluate a variety of special functions. The statistical routines calculate means, standard deviations and other statistics as well as providing several statistical analyses. On CONVEX computers, the IMSL routines may call subroutines which are in VECLIB; thus, it is recommended that both libraries be loaded when you use IMSL to avoid unsatisfied external references. To load IMSL on **Eagle** or **Mustang** use:

fc prog.f -lveclib -limslib

To load IMSL on **Voyager** or **Sabre** use:

cf77 -limslib prog.f

On-line documentation is available with the command *imsldoc,* which is located in the directory */usr/local/unsupported/bin.* The IMSL Interactive Documentation Facility (IDF) is scheduled to be put on **Eagle** in the spring of 1992. IMSL manuals are available for perusal in the Output Bins area of Building 1268, Room 1047. A brief description of IMSL subroutines is found in CR-3 and CX-3.

7.2. LIBSCI [a]

The scientific applications subroutines in LIBSCI are optimized for the CRAY computers. The routines include LINPACK, EISPACK, some of the BLAS (Basic Linear Algebra Subroutines), Fast Fourier Transforms, gather/scatter, search, sort and filter routines, matrix operations, linear recurrence routines and mathematical functions. The LIBSCI routines are listed in *CRAY UNICOS Math and Scientific Library Reference Manual* (CR-29). Descriptions of individual EISPACK and LINPACK subroutines may be found in chapters 4 and 5 of CX-16 and in chapters 2 and 3 of CX-26.

LIBSCI is automatically loaded with your FORTRAN program, but it is an external library. Several LIBSCI functions are declared to be COMPLEX, DOUBLE PRECISION or INTEGER. To insure correct results, you are cautioned to declare these functions in appropriate type declaration statements within the subroutines where they are referenced. Examples of these functions are CABS, DABS and IABS to compute absolute values and CSUM to sum complex variables.

7.3. VECLIB [a]

The VECLIB library is a collection of mathematical subroutines supplied by CONVEX. Included are the Basic Linear Algebra Subprograms (BLAS); the software packages LINPACK, EISPACK and LSQPACK; and subroutines for calculating FFT's, correlations and convolution. It also contains vector functions such as dot product (SDOT) and sum of the elements of a vector (SUM), which have CRAY equivalents. Detailed information about some routines may be found in *CONVEX VECLIB User's Guide* (CX-16); however, many LINPACK and EISPACK routines are not described in CX-16, but brief descriptions may be found in *CONVEX EISPACK and LINPACK Subroutines* (CX-26).

Users are warned that data precision is important when using VECLIB. CONVEX provides VECLIB and VECLIB8 for compatibility with the FORTRAN allowed data types. VECLIB or VECLIB8 must be explicitly loaded with the *fc* command. To load VECLIB or VECLIB8 on CONVEX use:

fc prog.f -lveclib

or

fc -p8 prog.f -lveclib8

See section 2.2 of the *CONVEX Mini Manual* (CX-1) for more discussion on passing arguments to VECLIB or VECLIB8.

7.4. LARCLIB [a]

The LARCLIB library is a collection of mathematical and statistical software developed at LaRC and other NASA centers. LARCLIB contains some of the subroutines found in the NOS-2 main mathematical library, FTN5MLB, and is available on all **SNS** computers. LARCLIB must be explicitly loaded with the *fc* and *cf77* commands. To load LARCLIB on **Eagle** or **Mustang** use:

fc prog.f -llarclib

To load LARCLIB on CRAY use:

cf77 -llarclib prog.f

On-line documentation is available with the command *larcdoc,* which is found in */usr/local/unsupported/bin.* More detailed information on LARCLIB routines can be found in CR-3, CX-3 and N2-3.

8. INFORMATION RESOURCES

Users of the **SNS** computers have several sources of information available to provide assistance. Since **Voyager, Sabre, Eagle** and **Mustang** all run a derivative of UNIX, much information is available on-line. However, ACD still provides substantial printed documentation (See section 7.1 of both CR-1 and CX-1), as well as consultation services and training that are described in the sections that follow.

8.1. On-Line Documentation

On-line documentation provided by CRAY and CONVEX includes the utility *man* and the Message-of-the-Day which are described in the next two sections. CONVEX also provide the utilities *info* and *apropos*. Local on-line documentation includes *imsldoc* and *larcdoc* which are described in sections 7.1 and 7.5 respectively.

8.1.1. man Pages

The *man* command displays the manual page(s) that describe any UNIX command. Most of these commands are described in CR-6 and CX-6. The *man* command only works for items that have an entry in the on-line documentation, so *man purge* yields no information on how to remove a file. It is necessary to learn that the command to remove a file is *rm* by some other means before using *man* to learn anything about removing UNIX files. If you use the *-k* option with the *man* command and specify a topic:

man -k topic

then *man* lists all commands relating to *topic*.

8.1.2 Message-of-the-Day

The Message-of-the-Day is found in the file */etc/motd*. It is automatically displayed whenever you login to any **SNS** computer. It is used to disseminate time-sensitive information. If you wish to review the message at a later time, use the command

more /etc/motd

to read the Message-of-the-Day, one page at a time.

8.1.3. info

The very first time you type *info* on either **Eagle** or **Mustang** several screens of information come up, explaining how to use the system before getting to the main menu. Once you have executed *info* only the main menu with eight items appears. Each item has one or more submenus that the user can peruse at leisure. The topics in the main menu are:

1. Contact other users or machines.
2. Use UNIX on-line help.
3. Execute commands.
4. Edit, find, print, modify, analyze, and archive files.
5. Develop programs.
6. Check user, job, or system status.
7. Modify system and file accessibility.
8. Perform arithmetic calculations.

The *info* command can also access information about commands by entering a related topic name. If you can't think of an appropriate topic or can't find information on the one you gave *info,* then you can look at the topic/command list for ideas. To access the topic/command list, type *t* from the main menu.

8.1.4. apropos

The *apropos* command may be used on the C2's to search the on-line manual for commands relating to a specific keyword. The syntax is:

apropos keyword

8.2. ACD Customer Services [a]

ACD consulting services have been decentralized. The User Consultation office has been closed. This decentralization puts you, the customer, in direct contact with the group or individual most familiar with your problem. The Operations Control Office, Network Support Office and LaTS Help Desk continue in operation. The CSCC Documentation Reference Library that was located in User Consultation has been relocated to the Output Bin Delivery area of room 1047 in Building 1268 (15 Ames Road).

The most effective contact for customer services is for the customer to directly call the appropriate ACD personnel or office. If you don't know who to contact, a comprehensive referral list may be found in the file *~acdcs/list* on all **SNS** computers (or in ACDLIST/UN=LIBRARY on the **Y** NOS computer). This list is also found in *Central Scientific Computing Complex Information Resources* (Document A-5). The referral list includes phone numbers and where appropriate, e-mail addresses and/or individual names. The goal is for initial contact back to the customer within 4 hours and problem resolution within 24 hours. This list will be enhanced and updated as required.

All phone numbers on the referral list have been supplied with PhoneMail. Use of e-mail is strongly encouraged as the primary method of inquiry. Any concerns or comments about ACD Customer Services may be sent to the ACD Customer Services Committee at Mail Stop 157B or via e-mail to:

acdcs@eagle.larc.nasa.gov

or to your LCUC representative. OCO may also be used as a focal point to relay concerns or suggestions (see **Table 8.1** on how to contact OCO).

ACD UNIX Support will provide assistance on a variety of topics, including **SNS** batch processing; the **SNS** Mass Storage Subsystem; UNIX questions; and referral to other ACD personnel. The phone numbers and e-mail addresses of the various offices providing full coverage during the prime shift are given in **Table 8.1.**

Office	Telephone	E-mail
ACD UNIX Support	864-UNIX	4unix@eagle
LaTS Help Desk	864-4444	
Network Support	864-7777	larcnet@ns
Operations Control	864-6562	oco@eagle

Table 8.1 ACD Customer Service Offices [a]

8.3. System Information [a]

The *notes* utility is installed on **Eagle** and **Mustang** only. It is for information of a lasting nature, until that material is incorporated into revised documentation. The set of topics is dynamic. By typing *notes* the current list of topics appears on your screen. Then by typing

notes topic_name

a menu appears listing all the notes for the topic. By typing the note number, the appropriate note is displayed to your screen. As of December 1991, these were the topics under *notes:*

CRadmin	- CRAY system management notices.
CRnews	- CRAY software/hardware announcements.
CRsyserrs	- CRAY system errors/problems/work-arounds.
CXadmin	- CONVEX system management notices.
CXnews	- CONVEX software/hardware announcements.
CXsyserrs	- CONVEX system errors/problems/work-arounds.
SNSnews	- Status information about SNS machines.
Xinfo	- X Window System information.
adase	- Ada Software Engineering special interest group.
announce	- Announcements of lectures, seminars, etc.
cbullet	- Computer Bulletins
docnews	- CSCC documentation notices.
dsasig	- Distributed System Administrators SIG.
general	- General purpose notes.
graphics	- Graphics News.
larcnet	- LaRC network news and announcements.
lbullet	- LaRCNET Bulletins
lcuc	- Langley Computer Users Committee.
mscnas	- CRAY NASTRAN information.
nfgripes	- Reports of problems with the notesfile program.
posix	- POSIX Standards.
sitelic	- LaRC Site License Information
tradoc	- CSCC Training & Documentation
unsup	- Unsupported Software announcements for CRAY/CONVEX.

The *notes* utility is described in Document CX-19. In particular, the topic *CRadmin* describes the NQS job size limits and file system management for **Voyager** and **Sabre,** while *CRsyserrs* discusses CRAY FORTRAN problems. The topics *CXadmin* and *CXsyserrs* are the comparable topics for **Eagle** and **Mustang.**

8.4. Training

The Learning Resource Center has a three part UNIX Fundamentals video course as well as the All-Hands-On UNIX Video Workshop. The workshop consists of ten video tapes, a textbook/workbook and a work disk of problems. Contact the Learning Resource Center at 864-2325 for more details on these courses. Additionally, ACD provides user training classes for UNIX, with examples geared toward the CONVEX computers, as well as classes that deal with specific issues of using the **SNS** computers in the LaRC environment. The next two sections describe the **SNS** and UNIX training classes supported by ACD.

8.4.1. SNS Training [a]

Each **SNS** class lasts from one-and-one-half to two hours. The following topics are covered:

1 SNS Programming Environment
2. Architecture and Performance of the SNS Computers.
3. SNS Job Execution.
4. Multitasking on the CRAY-2.
5. SNS Debugging.

For further information on **SNS** training, contact Wayne Murray at 865-1725.

8.4.2. UNIX Training

A basic series of UNIX classes (targeted for CONVEX UNIX) has been developed and is offered twice a year in the spring and autumn. There are seven classes in the series, each lasting about two hours, except for classes 6 and 7 which are each about one hour long. The topics covered include:

1. UNIX Basics.
2. ex and vi Editors.
3. FORTRAN: Compile, Load and Execute.
4. Customizing the C-shell Environment.
5. Pipes and Filters.
   Controlling Processes.
6. User to User Communications.
7. Networking.

Classes are also provided on other UNIX related topics such as the X Windows System. For further information on UNIX training, contact Wayne Murray at 865-1725.

8.5 Machine Availability [a]

The Message-of-the-Day, found in */etc/motd,* (see section 8.1.2) is used to announce periods of scheduled machine unavailability due to system time or preventative maintenance. Unfortunately, computers sometimes are unexpectedly out of service due to hardware or software failure. There is a mechanism for checking on machine availability without calling the Operations Support Office (OCO).

From any UNIX-based machine or workstation you can login to a Sun workstation called **OPS,** with the following:

telnet ops

You are prompted for a login id. If you use the special login id *ops,* you are not be prompted for a password. However you are able to read about both scheduled and unscheduled machine outages. Expected time of return to service is given whenever possible.

APPENDIX A - Guidelines for Sponsoring Unsupported Software

Definition

Unsupported Software on **SNS** computers is a collection of software sponsored by various users at LaRC, and is made generally available to the Langley user community. The directory, */usr/local/unsupported,* provides a mechanism for sharing this software with the rest of the user community. In general, the responsibilities of ACD are to verify that the software has been prepared properly for installation, and to install the software in the Unsupported Software directory. The responsibilities of the sponsor of Unsupported Software are outlined in the remainder of this document.

Introduction

When a user has a program that would be useful to other **SNS** users, and is willing to sponsor that program, the user submits a request through electronic mail for installation of software that meets the requirements outlined below. The sponsor then assumes responsibility for upgrading, fixing, and enhancing the software. An ACD "moderator" installs the software in the Unsupported Software directory. The user community is notified of the new software through the **SNS** *notes* utility.

For assistance in preparing software for installation, contact ACD UNIX Support at 864-UNIX (864-8649). Users are referred to appropriate personnel.

Sponsor's Responsibilities

The sponsor of Unsupported Software must comply with the rules listed below. Failure to adhere to these rules results in the removal of the software from the Unsupported Software directory.

1. Maintain an active **SNS** login and charge.

2. Ensure that the software does not violate any license agreements.

3. Determine, within reason, that imported software (eg. freeware, public domain, shareware) produces no unwanted or harmful effects when executed on **SNS** machines.

4. Ensure that the software performs as documented.

5. Accept responsibility for user questions and problem reports.

6. Provide to the ACD moderator all source code for the installation of the software (no binaries).

7. Verify that the software does not already exist on the system. The man command may be used.

8. Ensure that the file names associated with your software do not duplicate existing file names in the Unsupported Software directories.

9. Ensure that no file associated with your software in the Unsupported Software directory grows indefinitely or is globally writable.

10. Provide to the ACD moderator appropriate on-line documentation to enable others to make optimum use of the product, including at least one on-line *man* page. UNICOS *man* pages must be in ASCII format, and CONVEX UNIX *man* pages must be in *nroff* format. The *man* page is based on the standard *man* page in */usr/local/unsupported/doc/man.std.* **Note:** the sponsor's name and telephone number, and the Unsupported Software disclaimer must be included on every *man* page.

11. Provide the text for a note with the source code announcing the availability of the software. A standard note that should be used as a template is included in the file */usr/local/unsupported/doc/note.std.*

12. Provide a makefile for software installation based on the standard makefiles included in */usr/local/unsupported/doc/make.std.**. The makefile must recompile all binaries and set all file permissions.

    NOTE: ALL targets listed in the standard makefile must be included.

13. Test and migrate software to new releases of system hardware and software after a major **SNS** system upgrade.

14. Prepare all files for installation as described below.

Software Preparation for Submittal

No Unsupported Software resides in system directories other than those designated for this purpose.

Software Locations

The sponsor's makefile must place Unsupported Software files in the proper directories described below. The following is a list of */usr/local/unsupported* system directories, and a description of the types of files that reside in those directories:

adm	files for managing Unsupported Software
bin	executables
etc	miscellaneous system utilities
doc	text files
include files	including macros, declarations, etc.
lib	libraries
src	source code

The man pages reside in */usr/man/manp* and the file names require the *.p* extension.

Standard Installation Procedures

The sponsor must perform the following to prepare software for installation:

1. Prepare software as described in the section Sponsor's Responsibilities.

2. Create a source directory (with a descriptive name) containing everything needed for installation, i.e. source code, makefile, *man* page, announcement note, etc.

3. Change to the directory above the source directory and enter

   tar cf sw.tar sw

   where *sw* is the directory name for the software. This name cannot be the name of a directory already existing in */usr/local/unsupported/src*.

4. Send electronic mail to *unsup* on **Eagle, Mustang, Mickey's_choice** or **Voyager** and include the information listed in the file */usr/local/unsupported/doc/submit*. Print and sign the submit file, and mail it to the moderator at the mailstop indicated in the submit file.

The ACD moderator performs the following to install the software:

1. Start a typescript of the installation process to record the command sequence.

2. From */usr/local/unsupported/src,* enter

   tar xvf ~path_name/sw.tar

   where *~path_name/sw.tar* is the path name of the source code *tar* file.

3. From the newly-created */usr/local/unsupported/src/sw* source directory, the moderator enters

   make install

   (The sponsor must ensure that a proper makefile is included with the software, and that the moderator only has to enter *make install* from the source directory to complete the installation.)

4. If the installation is unsuccessful, the typescript of the installation is sent to the sponsor through electronic mail, and the software is removed from the Unsupported Software directories. When the sponsor has corrected the problem, the software can be reinstalled.

5. If the installation is successful, the moderator installs the announcement note prepared by the sponsor into the *notes category unsup and notifies the sponsor through electronic mail.*

Software Removal

A periodic examination of software in the Unsupported Software directory is conducted to determine if the software is still needed by the user community. Software that has not been used in 90 days, is removed without prior notice. The sponsor is notified through electronic mail after the software has been removed.

If the sponsor fails to follow the guidelines in this document, the software may be removed without prior notice.

A-8 SECTION GUIDE TO HIDDEN FILES

File	Description	Section
.exrc	Establishes environment for vi full screen editor	2.3.1
.forward	Tells mail where mail is to be forwarded	3.1.3
.mailrc	Establishes environment for electronic mail	3.1.4
.netrc	Allows use of File Transfer Protocol	5.3
.rhosts	Allows use of rlogin, rcp and remsh	5.2

A-8 SECTION GUIDE TO ENVIRONMENT AND C-SHELL VARIABLES

Variable	Description	
askcc	Ask mailx for a carbon copy prompt	3.1.4
autoprint	Automatically read next mail message	3.1.4
crt	Set the number of lines per screen for mailx	3.1.4
DELIVER	Set delivery information	4.1.4

A-8 SECTION GUIDE TO COMMANDS

Command	Description	Section
apropos	search for command related to keyword	8.1.4
cat	concatenate one or more files	2.2.7.3
cc	invoke C compiler	2.4.7
cd	change working directory	2.2.7.7
charge	change account	1.3.1
chmod	change file permissions	2.2.8
cp	copy one file to another	2.2.7.2
du	check disk usage	1.3.2
echo	echo message to screen	2.1.2
ed	invoke general purpose editor	2.3.2
emacs	invoke emacs editor on CONVEX	2.3.3
ex	invoke general purpose editor (subset of vi)	2.3.2
fmgen	automatic creation of makefile	3.3
fsplit	split FORTRAN source into separate .f files	3.3
ftp	transfer files between machines	5.3
grep	search a file for a character string	3.2.2
imsldoc	on-line documentation for IMSL	7.1
info	CONVEX on-line help utility	8.1.3
kill	stop a process	4.2.2
larcdoc	on-line documentation for LARCLIB	7.3
ln	create a link to another file	2.2.7.10
lpr	route file to line printer	4.4.2
ls	list file names in directory	2.2.7.1
mail	electronic mail for CONVEX	3.1
mailx	smart electronic mail for CRAY	3.1
make	utility to maintain large codes	3.3
man	obtain on-line documentation for UNIX commands	8.1.1
maschgrp	change group owner of mass storage file	5.5.8
maschmod	change permissions for mass storage file	5.5.7
masget	retrieve a file from mass storage	5.5.2
masls	list files on mass storage	5.5.5
masmkdir	create a directory on mass storage	5.5.3
masmv	move or rename file on mass storage	5.5.6
masput	store a file on mass storage	5.5.1
masrm	remove a file from mass storage	5.5.4
masrmdir	remove a directory from mass storage	5.5.9
mkdir	create a directory	2.2.7.8
more	filter to output only 1 screen of information at a time	3.2.1
mv	move file to new location or rename	2.2.7.5
nohup	allow background process to proceed after logout	4.2.1
notes	read information on various topics (CONVEX only)	8.3
passwd	change your password	4.1.1
pwd	print current working directory	2.2.7.6

quota	check file quotas	1.3.2
quotamon	monitor for exceeding soft limit	1.3.2
remsh	execute single command on remote host from CRAY	5.2
rcp	copy files between networked computers	5.4
rlogin	connect to remote host	5.2
rm	remove a link to a file	2.2.7.4
rmdir	remove a directory	2.2.7.9
rsh	execute single command on remote host from CONVEX	5.2
telnet	connect to a remote host	5.1
vi	invoke full screen editor	2.3.1

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 1992	Technical Memorandum

4. TITLE AND SUBTITLE
SNS Programing Environment User's Guide

5. FUNDING NUMBERS
505-90-53-02

6. AUTHOR(S) Geoffrey M. Tennille, Lona M. Howser, D. Creig Humes, Catherine K. Cronin, John T. Bowen, Joseph M. Drozdowski, Judith A. Utley, Theresa M. Flynn and Brenda A. Austin

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23665-5225

8. PERFORMING ORGANIZATION REPORT NUMBER
CSCC Doc A-8a

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-107565

11. SUPPLEMENTARY NOTES
Tennille, Howser, Humes, Cronin, Bowen, and Drozdowski: Langley Research Center, Hampton, Virginia.
Utley, Flynn, and Austin: Unisys Corporation, Hampton, Virginia.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 60

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document briefly describes the computing environment for the Supercomputing Network Subsystem (SNS) of the Central Scientific Computing Complex of the Langley Research Center. The major SNS computers are a CRAY-2, a CRAY Y-MP, a CONVEX C-210, and a CONVEX C-220. It describes the software that is common to all of these computers, including; the UNIX operating system, graphics, networking utilities, mass storage, and mathematical libraries. It also describes file management, validation, SNS configuration, documentation and customer services

The document is intended for all SNS users as a ready reference to frequently asked questions and to more detailed information contained within the vendor manuals. It is appropriate for both the novice and the experienced user.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Supercomputing, batch processing, graphics, networking, UNIX, mass storage	75
	16. PRICE CODE A04

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

NSN 7540-01-280-5500

Standard Form 298 (Rev 2-89)
Prescribed by ANSI Std Z39-18
298-102

NASA-Langley, 1992