

AD-A105 077

BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA
UNIX NSW FRONT END ENHANCEMENTS. VOLUME II.(U)

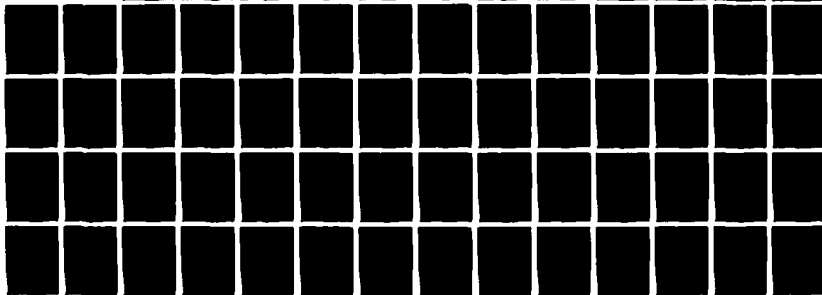
F/6 9/2

UNCLASSIFIED

JUN 81 H O LIND
BBN-4571-VOL-2

F30602-80-C-0062
RADC-TR-81-164-VOL-2 NL

1 1 1
2 2 2
3 3 3



END
DATE
FILMED
0-81
DTIC

AD A105077

LEVEL 12

RADC-TB-81-164, Vol II (of two)
Final Technical Report
June 1981



UNIX NSW FRONT END ENHANCEMENTS

Bolt Beranek and Newman, Inc.

Henrik O. Lind

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

DTIC
ELECTE
OCT 6 1981
S D
A

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

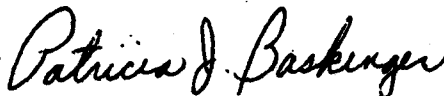
FILE: CORN

81 10 5 053

This report has been reviewed by the RADC Public Affairs Division and is releasable to the National Technical Information Service (NTIS). It will be releasable to the general public, including foreign nationals.

RADC-TR-81-164, Volume II (of two) has been reviewed and is approved for publication

APPROVED:



PATRICIA J. BASKINGER
Project Engineer

APPROVED:



JOHN J. MARCINIAK, Colonel, USAF
Chief, Information Sciences Division

FOR THE COMMANDER:



JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (ISCP) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

MISSION **of** **Rome Air Development Center**

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control, Communications and Intelligence (C³I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are: communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence collection and handling, information system technology, ionospheric propagation, solid state sciences, materials physics and electronic reliability, maintainability and compatibility.

17 TR-81-164-Vol-2

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-81-164, Vol II (of two)	2. GOVT ACCESSION NO. AD-A105 077	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) UNIX NSW FRONT END ENHANCEMENTS	5. TYPE OF REPORT & PERIOD COVERED Final Technical Report Jan 80 - Oct 80	6. PERFORMING ORG. REPORT NUMBER FEN-4571-Vol 2
7. AUTHOR(s) Henrik O. Lind	8. CONTRACT OR GRANT NUMBER(s) F30602-80-C-0062	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman, Inc. 50 Moulton St. Cambridge MA 02138	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62702F 55812124	
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (ISCP) Griffiss AFB NY 13441	12. REPORT DATE June 1981	13. NUMBER OF PAGES 64
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same	15. SECURITY CLASS. (of this report) UNCLASSIFIED	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Same		
18. SUPPLEMENTARY NOTES RADC Project Engineer: Patricia J. Baskinger (ISCP)		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Computer Networks Software Systems Network Operating Systems		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The effort to develop a UNIX NSW Front End is part of the National Software Works (NSW) program sponsored jointly by the Air Force and ARPA. The goal of the NSW program is to develop a network operating system (called the NSW system or NSW for short) that provides an effective environment for software production, software configuration control, and software maintenance. →		

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Users access the National Software system by means of an NSW Front End. The objective of the UNIX Front End project was to develop an NSW Front End that runs on a DEC PDP-11 computer under the UNIX operating system.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

NSW UNIX FRONT END USER'S MANUAL

Henrik O. Lind

A

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
2. TYPING CONVENTIONS	5
2.1 Typing Conventions in Exec Mode	5
2.1.1 Command Entry Conventions	5
2.1.1.1 Command Fields	5
2.1.1.2 Command Field Delimiters	6
2.1.2 Special Function Characters	9
2.1.2.1 Command-Editing Characters	10
2.1.2.2 Help Facilities	12
2.1.2.3 Status Query Character	
Help Calls	13
2.1.4 Query Mode	14
2.1.5 Suppress Output Capability	15
2.2 Typing Conventions in Tool Mode	15
3. RETURN MODES	17
3.1 Immediate Return Mode	17
3.2 Deferred Return Mode	20
3.3 Changing Return Mode	20
3.3.1 From Immediate to Deferred Return Mode	21
3.3.2 From Deferred to Immediate Return Mode	21
3.4 Escaping from Deferred Return Mode	21
3.5 Viewing Output From A Command	22
3.5.1 In Immediate Return Mode	22
3.5.2 In Deferred Return Mode	23
4. NSW FILE SYSTEM	25

4.1	File Names	25
4.2	FileSpecs	26
4.3	Scopes	27
5.	USING NSW SERVICES	29
5.1	NSW Service Names	30
5.1.1	NSW Tool Names	30
5.1.2	TELNET Connection and UNIX Subshell Names	30
5.2	Invoking an NSW Service	31
5.3	Ending Use of an NSW Conversational Partner	31
5.3.1	NSW Commands to Stop Conversational Partners	33
5.3.2	Stopping Conversational Partners Directly	33
5.4	Resuming Use of an NSW Conversational Partner	34
6.	STARTING AND STOPPING THE FRONT END	35
6.1	Starting the Front End	35
6.1.1	Front End Dispatcher	36
6.2	Stopping the Front End	36
6.2.1	Autologout	37
7.	SUMMARY OF NSW COMMANDS	39
7.1	ABORT Command	39
7.2	ALTER Command	39
7.3	COPY Command	40
7.4	DELETE Command	41
7.5	DESCRIBE Command	42
7.6	DISCARD Command	43
7.7	DISPLAY Command	44
7.8	HELP Command	45
7.9	JOB Command	45
7.10	LOGIN Command	46
7.11	LOGOUT Command	47
7.12	NET Command	48
7.13	NEWS Command	49
7.14	PASSWORD Command	49
7.15	QUIT Command	50
7.16	RENAME Command	50

7.17	RESUME Command	51
7.18	SEMAPHORE Command	51
7.19	SET Command	52
7.20	SHELL Command	53
7.21	SHOW Command	53
7.22	STOP Command	55
7.23	TELNET Command	55
7.24	TERMINATE Command	56
7.25	USE Command	57

APPENDIX A.	FRONT END CONFIGURATION CONTROL	59
-------------	---------------------------------	----

A.1	Event Logging	60
A.2	Diagnostic Typeouts	60
A.3	Example of Initialization File	61
A.4	Error Log Files	62

1. INTRODUCTION

In the National Software Works (NSW), the component which provides the user interface to the system is called the Front End (FE). In this document, the characteristics of the UNIX NSW Front End are described.

Although a basic understanding of the NSW system is presupposed, some remarks are in order. NSW is a distributed network operating system. Being an operating system, it supports a file system and access to software tools with which the user may manipulate files. Being a distributed system, the component pieces of the system are typically scattered over multiple hosts.

The user accesses NSW via the Front End component. The controlling component of NSW, the Works Manager (WM) component, is generally on a different host and is involved in most of the user's interactions with NSW. If the user invokes the use of an NSW software tool, which usually is on yet a different host, he will involve the NSW Foreman (FM) component on that host.

Therefore, network interactions are an integral part of using NSW, though these interactions are invoked implicitly by the user when he issues an NSW command. Practically, this means that NSW commands may take greatly varying amounts of time to complete, depending on the system loads of the hosts involved,

the load on the network, and the complexity of the transactions between the NSW components required. Some commands are executed directly by the Front End and complete immediately. For other commands that require interactions with other components, the Front End will announce the initiation and completion of the command.

It is possible to operate the Front End in two "return modes" (see Chapter 3). In "immediate return" mode, after the user has initiated a command, he will be able to enter additional commands before the first command has completed. For "deferred return" mode, having initiated a command, the Front End will not accept additional commands until execution of the first command has been completed.

The user is either communicating with the NSW operating system or with an NSW conversational partner, i.e., an NSW tool, a TELNET connection, or a UNIX subshell (see Chapter 5). When the user communicates with the NSW operating system, he is said to be in "exec mode", and what he types is interpreted according to the conventions of exec mode. When the user communicates with an NSW conversational partner he is said to be in "tool mode", and what he types is interpreted according to the conventions of tool mode. The conventions of tool mode are simply those of the NSW conversational partner with which the user is communicating.

The Front End provides means for a user to switch between exec mode and tool mode.

2. TYPING CONVENTIONS

2.1 Typing Conventions in Exec Mode

In exec mode, the user may enter NSW commands, find out the status of commands already entered, display the output of commands, or suppress the output of commands.

2.1.1 Command Entry Conventions

2.1.1.1 Command Fields

NSW commands are composed of one or more command fields. Except in a few cases, each command field is formed by upper- or lower-case alphabetic, numerics, underline ('_'), hyphen ('-'), and period ('.'); upper- and lower-case alphabetic are treated as equivalent.

There are some command fields which the Front End can "recognize" and some which it cannot "recognize." Recognizing a field means Front End matches what the user has typed in against a set of strings which are acceptable in that position in the command. The user need only type enough of the string to uniquely specify which one is intended.. For example, after the user has typed 'show files', he must type one of five possible strings for the next command field to specify an access type (i.e., 'all', 'any', 'copy', 'delete', or 'enter'). Any other

string will not be recognized.

There are some fields for which recognition cannot be supported. For example, if a command field requires an NSW filename, recognition is not a practical option (since the file catalog is maintained by the Works Manager component which resides on another host).

The significance of recognition is discussed in the following sections.

2.1.1.2 Command Field Delimiters

Command fields are separated by command field delimiters. These delimiters indicate that the user has finished one command field and wishes the Front End to accept it. The Front End has five command field delimiters: space, tab, carriage-return, linefeed, and comma. We will describe each of them.

- o SPACE or TAB (End of Field)

Typing SPACE or TAB will cause the Front End to try to recognize the command field if recognition is supported for that command field; otherwise, the Front End will attempt to verify that the command field is of the right "type", e.g., if an integer is required in the command field, the Front End will verify that every character is numeric.

If the command is not recognized or is of the wrong "type", the Front End will output a signal (bell) to the user, followed by CR/LF, a prompt for the command field, followed by the command field already typed by the user. The user may then use the command editing features to modify the field, cancel the field, or abort the command

(see Section 2.1.2.1). For example, if the user typed 'fides' instead of 'files' in a SHOW FILES command, the user's command line would develop in time as follows:

```
NSW: show fides
                                ;user types SPACE
NSW: show fides [bell]
      (info about) fides
```

and the cursor will be positioned after the 's' in 'fides'.

Or, in a DISPLAY command, if the user types an incorrect integer, e.g., 'lx', the user's command line would develop in time as follows:

```
NSW: disp lx
                                ;user types SPACE
NSW: disp lx [bell]
      (output for command) lx
```

and the cursor will be positioned after the 'lx'.

o CR or LF (End of Command)

Typing CR (carriage-return) is how the user should indicate that he has reached the end of the NSW command he is entering. If LF (linefeed) is typed in the middle of entering a command, it is equivalent to CR in function. However, if no command is being entered and the user types LF, it is treated as a request to "display the output of the oldest completed NSW command"; if no such output exists, LF is equivalent to CR (refer to Section 3.5.1 below).

When the user types CR, the Front End first checks that the previous command field is valid. If it is not valid, the Front End will treat it as an invalid field as described above.

If the command field is valid and it is the last command field of a valid command, the Front End asks the user to confirm that this command is indeed what he wishes to enter. At this point, the user may confirm positively

by typing another CR, which echoes as '!' followed by CR/LF. In this case, the Front End begins to execute the command. Consider the following example of a command line as it develops in time:

```
NSW: show node
                                ;user types CR
NSW: show node! [confirm]
                                ;user types confirming CR
NSW: show node! [confirm] !
    [2: "show node" initiated]
```

Or, the user may choose not to confirm the command by using any of the command-editing characters described in Section 2.1.2.1 below to modify or abort it.

Every NSW command requires two CR's to be initiated, one to signal that the specification of the command is complete and the second to confirm the command. If the command field is valid, but it is not the last command field of a complete command, the FE echoes '!', outputs a signal, and prompts the user for the next field. So, for example, the user's command line might develop in time as:

```
NSW: show
                                ;user types premature CR
NSW: show ! [bell]
    (info about)
                                ;FE prompts for next field
```

COMMA (Begin Command Modification)

A command modification is a qualification of a complete NSW command. Although only the LOGOUT command currently has command modifiers defined for it, it is likely that this feature will be more common in the future. Command modification is invoked by typing ',' (comma), followed by CR. For example, the unmodified LOGOUT command would be entered as:

```
NSW: logout! [confirm] !
```

With the 'fast' modification, the LOGOUT command would be entered as:

```
NSW: logo,  
      fast! [confirm] !
```

The effect of "LOGOUT, FAST" is to invoke a logout sequence with the qualification that it be done "fast" i.e., aborting any active NSW tools. So, this command is a modified form of a normal LOGOUT command.

Naturally, before the command modification request can be accepted by the Front End, it must check that the previous command field is valid and that the command supports modification.

2.1.2 Special Function Characters

Several characters have special functions when typed in exec mode. These characters are used to facilitate the entry of NSW commands in several ways: correct mistakes made in typing, find out a list of possible options in a command field, find out what the field expected next is, erase a partially (or completely) entered command and start over, and so on.

Broadly, there are three categories of actions which these special characters perform:

1. Command-Editing

The user may alter all or part(s) of a command he has typed in.

2. Help Facilities

The user may find out how to construct legal NSW

commands.

3. Status Query

The user may find out the status of NSW commands in progress, the status of NSW conversational partners, the Front End return mode, and so on.

The following considers each category in turn.

2.1.2.1 Command-Editing Characters

In the process of entering an NSW command, the user may make a mistake or change his mind about which command or command arguments to use. The special characters which have command-editing functions defined enable the user to make such changes. The following characters are available:

- o DELETE character (Delete Character function)

When the user types the DELETE character, the previous character typed will be removed from the command he is entering. To indicate that the character has been removed, a visible indication, consisting of a backslash followed by the character removed, is printed on the terminal.

It is possible to erase an entire command by successive uses of this character if desired, though there are quicker ways to do so. In using the Delete Character function, prompts are automatically skipped over, i.e., the user does not have to delete prompts one character at a time.

- o CONTROL-W character (Delete Field function)

When the user types control-W, an entire field of his command will be removed. If he is in the middle of typing in a command field when he types control-W, that field will be removed. If it is not in the middle of a field, the previous field will be removed. A visible

indication, consisting of "<" followed by the first letter of the field removed, is typed out on the user's terminal to indicate that the field has been removed.

It is possible to erase an entire command by successive uses of this character if desired, though there is a special character set aside for that function. As with deleting characters, so in this case prompts are ignored (i.e., they are automatically skipped over).

- o CONTROL-R character (Retype Command function)

When the user types control-R, the entire command he has entered thus far, including prompts, is displayed to him on a new line. This function is useful for seeing the state of the command if many alterations have been made to it.

- o CONTROL-X character (Abort Input function)

When the user types control-X, the entire command entered thus far is deleted, and the Front End is ready to accept a new command from the beginning. The standard Front End exec mode prompt "NSW: " is printed to indicate the Front End's readiness to receive a new command.

A note of caution: The character control-X has other defined uses in the Front End as well as invoking the Abort Input function when entering commands. See Sections 2.1.5 and 3.4 below for other uses of control-X.

- o CONTROL-V character (Quote Character function)

When the user types control-V, the following character is "quoted", i.e., it is treated as an "ordinary" character having no special functions and is placed into the command the user is entering. Any character may be quoted.

For example, some computer systems allow embedded spaces in filenames. Without a means of telling the Front End to "treat this space as an ordinary character", i.e., not as a command field terminator, the use of such systems is restricted.

2.1.2.2 Help Facilities

In entering an NSW command, the user may need assistance in constructing the command. The special characters which invoke help facilities enable assistance to be given by the Front End.

The following characters are available:

- o ESCAPE character (Recognize and Complete function, and/or Print Prompt function)

The ESCAPE character has a dual function defined for it. Which function is invoked will depend on where in the process of command entry the ESCAPE character is typed.

When the user types ESCAPE in the middle of typing a command field, the Front End attempts to recognize the field. If the field is recognized, the remainder of the field is typed out on the user's terminal (i.e., completed) and is also saved internally by the FE. If the field is not recognized, the user is given a signal (bell) to indicate that either (1) the field does not match any valid string in that position, (2) the field matches more than one valid string, or (3) recognition is not supported for that field. This use of the ESCAPE character is called the Recognize and Complete function.

If the user types ESCAPE and there is no command field partially entered, the Front End will display a prompt enclosed in parentheses indicating the nature of the next command field expected. This use of the ESCAPE character is called the Print Prompt function.

- o QUESTION MARK character (Show Options function)

When the user types "?", the Front End types out information about the set of all possible strings which the user may enter at that point.

If recognition is supported for the current or next command field, a list of possible options is typed out on the terminal. If a command field has been partially entered and the user types "?", the FE will type only a list of options having the initial string already typed in.

If recognition is not supported for the current or next command field, a description of the kind of item expected is typed out on the terminal.

2.1.2.3 Status Query Character

At any time in the user's Front End session, he may find out the status of the session by typing the Status Query character control-T. Typing control-T will cause information about the user's state with respect to NSW to be typed out. This information includes:

- current date and time
- whether the user is logged in to NSW or not
- the Front End return mode
- information and status about active NSW commands
- information and status about NSW conversational partners

The Status Query function invoked by typing control-T is equivalent to typing the following sequence of NSW commands:

```
SHOW STATUS USER-STATE
SHOW STATUS RETURN-MODE
SHOW ACTIVE COMMANDS
SHOW ACTIVE TOOLS
```

Refer to the descriptions of these commands in chapter 7 for detailed information concerning them.

2.1.3 Help Calls

Because of the distributed nature of the NSW operating system, it is occasionally necessary for remote NSW components to

request further information from the user to enable them to carry out their tasks. This typically occurs with file specifications. For example, if the user tries to invoke an NSW file operation (e.g., DELETE) for which there is no file or multiple files matching his file specifier (see chapter 4 below), the Works Manager sends a message to the user asking him to alter or disambiguate his file specifier. Such a message is called a Help call.

All typing conventions of exec mode are in effect when the user is answering a Help call.

2.1.4 Query Mode

A few NSW commands have many arguments. To reduce the confusion and error associated with such long and complicated commands, the Front End supports a special input mode called "query mode." If the user is entering an NSW command for which query mode is implemented, the Front End will automatically begin querying the user for the fields of the command. The user is thus freed from having to remember the arguments and their order. All of the typing conventions of exec mode remain in effect.

Currently, query mode is implemented only for the NET command (see Section 7.12).

2.1.5 Suppress Output Capability

At any time in exec mode, the user may suppress output being typed out by the Front End by typing control-X.

For example, if an NSW command such as 'show files any *' produces an extremely large amount of output, and if the user does not wish to see all of it, he may use control-X.

2.2 Typing Conventions in Tool Mode

When the user is in communication with an NSW conversational partner (see Chapter 5), the Front End is in tool mode, and the typing conventions in effect will be those of the conversational partner. In tool mode, however, the Front End always interprets two characters in a special way; they are control-N and control-V:

- o CONTROL-N character (Switch to Exec Mode function)

When the user types control-N, he will invoke a change from tool mode to exec mode. The NSW conversational partner he has been communicating with will remain in the state just prior to typing control-N. That is, the connections will remain open, and any output from these will be buffered by the Front End. The user may resume communication with any NSW conversational partner by using the RESUME command.

- o CONTROL-V character (Quote Character function)

When the user types control-V, the following character typed will be read by the Front End and transmitted to the conversational partner without invoking any special

functions that may be associated with it. Thus, in particular, the sequence <control-V><control-N> causes the control-N character to be sent to the NSW conversational partner without invoking a change from tool mode to exec mode (see discussion on control-N above). Also, for example, the sequence <control-V><control-V> causes <control-V> to be sent to the conversational partner. When further special characters are defined for the Front End in tool mode, control-V will always enable these to be sent without invoking the special functions defined for them.

3. RETURN MODES

Because of the distributed nature of NSW, the time to execute NSW commands may vary from nearly instantaneous to times on the order of minutes. As noted in the introduction, the longer delays occur for two reasons: (1) the inherent complexity of certain commands, i.e., the nature and number of network transactions involved, and (2) delays on the local host, remote hosts, and the network.

For these reasons, the Front End supports two "return modes": immediate return mode and deferred return mode. A return mode defines when control is returned to the user relative to the completion of an NSW command issued by the user. This section describes the two Front End return modes.

3.1 Immediate Return Mode

The default return mode of the Front End is immediate return mode. In immediate return mode, control is returned to the user immediately after the Front End has initiated an NSW command. After control is returned to the user, (i.e., after he has received the "NSW: " prompt), he may issue other commands as he wishes.

For each NSW command issued in immediate return mode which

generates remote activity, the Front End assigns a command number. The number is announced to the user as the integer at the left of a command announcement, e.g.:

```
NSW: logout! [confirm] !  
      [5: "logout" initiated]
```

In this case, the command number is 5. NSW commands which manipulate other NSW commands refer to them by their command numbers. Thus, the DISCARD and DISPLAY commands take a command number as an argument and, respectively, discard or display the output from the command having the command number.

When a command completes, the user is notified that it has completed. Depending on the command, there are three cases:

1. If the command produces output and the output produced is short, it is displayed automatically to the user.
2. If the command produces output and the output produced is long, it is not displayed automatically, and the user must view the output with DISPLAY command.
3. If the command initiates a NSW conversational partner, the user is notified and must use a RESUME command to switch to the conversational partner.

For example, suppose the user wishes to know which files are accessible to him. The terminal session may appear as follows:

```

NSW: show files any *! [confirm] !
      ;user issues command
      [2: "show files any *" initiated]
      ;FE announces its initiation
      .
      .
      .
NSW:
      [2: "show files any *" complete; output ready]
      ;FE announces completion of command
NSW: display 2 ! [confirm] !
      ;user views output with a DISPLAY
      ;command
      <list of NSW files in user's node>

```

Or, the user may wish to use an NSW tool:

```

NSW: use sos-ie! [confirm] !
      ;user issues command
      [3: "use sos-ie" initiated]
      ;FE announces its initiation
      .
      .
      .
NSW:
      [3: "use sos-ie" completed; lsos-ie ready to use]
      ;FE announces completion of command
NSW: resume lsos-ie! [confirm] !
      ;user begins communication with
      ;tool

```

Commands are not guaranteed to complete in the order in which they were issued. Commands which are simpler or which produce less output may complete ahead of previously-issued, complex commands or those which generate much output. Therefore, a command which depends on the completion of a previously-issued command should not be entered until the previous command completes. The Front End provides the capability to issue

several (currently, 10) independent commands. In the future, a means for sequencing commands may be implemented.

3.2 Deferred Return Mode

Unlike immediate return mode, in deferred return mode, control is not returned to the user until the command is completed. That is, terminal type in prior to the completion of the current command is merely buffered (but not processed or echoed) until the command has completed. In deferred return mode, the Front End outputs a more limited set of announcements to the user; e.g., "command initiated" informs the user his command has been initiated. There is no need for the full range of user announcements, since output from completed commands is automatically displayed to the user and communication with requested NSW services is automatically invoked.

3.3 Changing Return Mode

The user may at any time change the Front End return mode by using the SET RETURN-MODE command, which takes the desired return mode as its argument (see Section 7.19). The effects of the change are described below.

3.3.1 From Immediate to Deferred Return Mode

When the user changes from immediate to deferred return mode, all pending commands (i.e., those issued, but not yet completed) complete as if issued in immediate return mode. That is, immediate return mode user announcements appear, and the user must explicitly request to view the output from the commands or to transfer to the relevant conversational partner.

After changing to deferred return mode, all subsequently entered commands behave as described in Section 3.2 above.

3.3.2 From Deferred to Immediate Return Mode

When the user changes from deferred to immediate return mode, all subsequent commands behave as described in section 3.1 above.

3.4 Escaping from Deferred Return Mode

It would be both inconvenient and potentially disastrous if there were no way for the user to regain control of the terminal while waiting for a command to complete in deferred return mode.

For this reason, control-X can be used as the control character for escaping from deferred return mode. If while waiting for a command to complete in deferred return mode, the

user types control-X, he will receive an NSW command prompt and may issue a new NSW command, and the command upon which the user was waiting is treated as having been issued in immediate return mode. Use of the control-X character in this way does not change the Front End return mode.

3.5 Viewing Output From A Command

As noted, the return mode of the Front End affects the way the user may view the output or results of a command. Some commands cause actions outside the Front End to be initiated, whereas other commands can be executed directly by the Front End. Regardless of return mode, if the user issues a command which can be executed directly by the Front End, the output or result of the command will be displayed directly to the user.

3.5.1 In Immediate Return Mode

If the user is in immediate return mode and issues an NSW command which initiates a remote action, resulting output is handled as follows:

- o If the command generates a small amount of output (128 characters or less), it is displayed automatically to the user.
- o If the command generates a greater amount of output (more than 128 characters), it is buffered, and the user is notified that the output is ready; the user may view the output of the command by using the DISPLAY command,

followed by the command number of the NSW command (see Section 7.7) for which output is ready.

- o If the command causes a Help call to the FE (see Section 2.1.3) to be generated, the user is notified that Help is needed for the command; the user may view the Help request by using the DISPLAY command, followed by the command number of the NSW command (see Section 7.7) for which Help is needed.
- o If the command generates an error in execution, any error output is displayed automatically to the user.

The DISPLAY command may be issued with a null (empty) command number argument; this defaults to the oldest completed NSW command (see Section 7.7 below).

Linefeed (LF) has a special function when typed at the beginning of a command line. It is treated as a shorthand for a DISPLAY command with a null command number argument; that is, it is treated as a request to display the output of the oldest completed NSW command.

As noted in Section 3.5.2 above, if the user escapes from deferred return mode with control-X, any previous commands issued in deferred return mode will be treated as if issued in immediate return mode. The output of such commands should also be viewed as described in this section.

3.5.2 In Deferred Return Mode

If the user is in deferred return mode, generally no action

is required to view the output of any command. With one exception, output is displayed automatically to the user. The exception is the case in which a NSW command has initiated a remote action and the user has used control-X to escape from waiting on its completion. Output from this command is viewed by the conventions of viewing output generated by a NSW command in immediate return mode.

4. NSW FILE SYSTEM

The NSW supports a file system which the user may use for his work. There are NSW commands to copy, rename, and delete files, as well as to transfer files into and out of NSW. These commands are described in detail in Chapter 7.

4.1 File Names

An NSW file name may consist of one to ten components. Each component may consist of upper- or lower-case letters, numerals, underline ('_'), or hyphen ('-') (see Section 2.1.1.1). Components may be from one to twelve characters in length, and they are separated by periods ('.'). Upper- and lower-case letters are considered identical. For example, A.B.C.D is a valid file name and refers to the same file as a.b.c.d, a.B.c.D. and so on. The components have no semantic significance to the NSW operating system other than for use as keys for file directory lookup operations, but users generally find it useful to give files mnemonic names.

Thus, NSW file names may in principle be quite long. Several provisions have been made to reduce the need to type the full file name when interacting with NSW. Most notably, filespecs are abbreviations of file names and are described next.

4.2 FileSpecs

A filespec is an abbreviation of an NSW file name. It may contain all or some of the components of the full NSW file name. When a filespec is given to NSW, the system will attempt to find the NSW file which contains the filespec as part of the file name. If there is more than one NSW file whose file name matches the filespec, the user will be asked to disambiguate the filespec by means of a Help call.

For example, if the full NSW file name of a file is alpha.beta.gamma.delta.epsilon, then alpha is a filespec, as are alpha.beta, beta, beta.gamma, gamma.delta.epsilon, and so on.

Also, the components in a filespec may be separated by ellipses ('...', i.e., three periods). Ellipses in a filespec mean that the file or files in question has/have some given component followed, immediately or not, by another given component. In the example above, alpha...gamma is a valid filespec and so, e.g., are gamma...delta and gamma...epsilon. More than one set of ellipses can appear in a filespec, but they may not be adjacent. For example, alpha...gamma...epsilon is a valid filespec, but alpha.....epsilon is not.

It is possible to force NSW to interpret the filespec given as matching the end of the filename. This is done by appending a

pound sign ('#') to the filespec. Thus, gamma...epsilon# would match any file name which contains a component called 'gamma', followed, immediately or not, by a component called 'epsilon', with the requirement that 'epsilon' be the last component of the file name.

The next section will discuss the roles of scopes in file-manipulation. There is a special character provided to override the normal scoping in effect and to cause NSW to search all of its file space, as opposed to that portion of the file space within the user's scope. The dollar sign ('\$') is prefixed to a filespec to force the user's scopes to be overridden.

4.3 Scopes

A scope is a leading component or components of a NSW file name which is/are prefixed to file names or filespecs given by the user. There are three kinds of scopes: copy, delete, and enter scopes. These scopes are in effect when the user is doing file copy, delete, and enter (i.e., file creation) operations, respectively. Each NSW node has default scopes associated with it, and the scopes may be altered by the user using the ALTER command (see Section 7.2).

To illustrate, suppose the user wishes to make a copy of a file, his copy scope is BBN and COMPASS, and his enter scope is

BBN. If the user types:

NSW: COPY X X.COPY

the system uses the copy scopes to form the filespecs BBN...X and COMPASS...X and searches for files that match these specifications. So, either BBN.X or COMPASS.X would match the filespec. The NSW file name for the file created by the copy operation is formed by appending the name specified by the user, called the entry name, to the enter scope: e.g., BBN.X.COPY. The entry name should be viewed as the terminal string of a filename; it would be meaningless to interpret it as a filespec, in particular, if it contained ellipses.

Scopes do not alter the access rights of an NSW node, but rather serve to restrict the portion of NSW file space which is searched when trying to match a filespec.

Suppose in the example above that both files BBN.X and COMPASS.X existed. The user would be asked to specify which file he wished to have copied by means of a Help call, and the copy operation would then continue. However, had the user altered his copy scope previously to drop either the BBN or COMPASS scope, there would have been no ambiguity in file name, assuming, of course, that the filespec X uniquely specified a file within the BBN or COMPASS scope.

5. USING NSW SERVICES

The Front End allows the user to invoke and interact with three types of NSW services. A service is a program or set of programs which perform operations for the user. The Front End supports access to the following NSW services:

o ENCAPSULATED NSW TOOLS

These are programs encapsulated to run on NSW tool-bearing hosts. They operate on NSW files, which usually are transported to and from the host, and accept NSW file specification conventions. Examples of such programs are editors, compilers, loaders, etc.

Encapsulated NSW tools form the most significant subclass of NSW services. They are invoked by using the USE command (see Section 7.25). For brevity, we may refer to them as "NSW tools", or simply as "tools."

o TELNET CONNECTIONS

Using the TELNET command (see Section 7.23), the user may establish a standard TELNET connection to a remote host.

o UNIX SUBSHELLS

Using the SHELL command (see Section 7.20), the user may invoke a UNIX subshell accessible from the Front End.

When these services are invoked (i.e., when they are active), we adopt the convention that they are called NSW conversational partners. A total of eight NSW conversational partners may be in use at any time.

5.1 NSW Service Names

5.1.1 NSW Tool Names

Each NSW tool has a tool name and a tool-instance name. The tool is invoked by the user using the tool name, but is subsequently referred to by the tool-instance name.

The tool name is a program name, followed by '- ', and by a two-character string identifying which NSW tool-bearing host the program runs on. For example, 'teco-ie' is a TECO tool which runs on host ISIE and is distinct from 'teco-ic', which is a TECO tool that runs on host ISIC. The user may find out which NSW tools are available to him by using the SHOW NODE command (see Section 7.21). The tool-instance name for a tool consists of the tool name, preceded by an FE-assigned numeric prefix, e.g., 'lteco-ic'.

5.1.2 TELNET Connection and UNIX Subshell Names

The names for TELNET connections and UNIX subshells are selected by the user at invocation time. These names are specified as arguments of the TELNET and SHELL commands and may be any character string.

5.2 Invoking an NSW Service

Using the USE, TELNET, or SHELL commands, the user may invoke an NSW service to use as an NSW conversational partner. Refer to the descriptions of these commands in Chapter 7 for details of using the commands.

In immediate return mode, the invocation process is a two-step affair: (1) NSW does everything necessary to connect the conversational partner to the Front End; (2) using the RESUME command, the user begins interacting with the conversational partner. The invocation process for the NSW tool 'qedx-rm' might appear as follows:

```
NSW: use qedx-rm! [confirm] !
```

```
NSW:
```

```
[5: "use qedx-rm" complete; lqedx-rm ready to use]
```

```
NSW: resume lqedx-rm! [confirm] !
```

```
[now talking to lqedx-rm]
```

In deferred return mode, it is not necessary to use the RESUME command to initiate interaction with the conversational partner. Interaction is initiated automatically.

5.3 Ending Use of an NSW Conversational Partner

The use of NSW conversational partners may be ended in several ways. The user may issue an NSW command to do this, he

may issue a command to the conversational partner to stop itself, or (in case of an error) the conversational partner may stop itself automatically.

In any case, the user will be notified that the conversational partner is "closed." If there is buffered output on the connection, the user is notified and must (in immediate return mode) use a RESUME command to view the output. Finally, for NSW tools, there is NSW accounting information displayed to the user.

As an example, suppose a user wishes to terminate the active NSW tool '3teco-r2'; the terminal session might appear as follows:

```
NSW: terminate 3teco-r2! [confirm] !  
      [3: "terminate 3teco-r2" initiated]
```

```
NSW:  
      [3: 3teco-r2 closed (output waiting)]
```

```
      ;user notified that tool is closed, but  
      ;there is buffered output; the RESUME  
      ;command can be used to see it
```

```
NSW: resume 3teco-r2! [confirm] !  
      [now talking to 3teco-r2]
```

```
      <the tool's buffered output is displayed>
```

```
NSW:  
      [3: "terminate 3teco-r2" complete; 3teco-r2 closed]
```

```
      <NSW accounting information is displayed>
```

5.3.1 NSW Commands to Stop Conversational Partners

The use of NSW tools is ended by the QUIT ABORT, QUIT TERMINATE, or TERMINATE commands. "Aborting" a tool means that no files will be delivered from the tool's workspace to the NSW file system before the tool halts. "Terminating" a tool means that all appropriate files in the tool's workspace will be delivered to the NSW file system before the tool halts.

For TELNET connections and UNIX subshells, there is no distinction between "aborting" and "terminating." TELNET connections are ended by the QUIT ABORT, TERMINATE, or QUIT TELNET commands. UNIX subshells are ended by the QUIT ABORT, TERMINATE, or QUIT SHELL commands.

It is expected that the QUIT SHELL, QUIT TELNET, and QUIT TERMINATE commands will be removed from the UNIX NSW Front End command language. The user is encouraged to use the TERMINATE command instead.

5.3.2 Stopping Conversational Partners Directly

The user may end the use of an NSW conversational partner by ordering it (in its own syntax) to stop. For example, TECO tools are stopped by the familiar `^;h^` command; QEDX is stopped by `^q^`.

For TELNET connections, the connection is closed

automatically by some hosts, e.g., TOPS-20 hosts, when the user logs out. For some other hosts, TENEX hosts, for example, the connection remains open after the user has logged out, and he must use a TERMINATE command to close it.

There is no means implemented to force a UNIX subshell to stop itself. A TERMINATE command must always be used.

5.4 Resuming Use of an NSW Conversational Partner

The user is connected to an NSW conversational partner by means of the RESUME command (see Section 7.17), which takes as its argument the NSW tool-instance name, or the name of the TELNET connection or UNIX subshell which the user assigned at invocation.

The RESUME command is used in the following instances:

- o When an NSW conversational partner has finished its startup and the user is ready to switch to it for the first time.
- o When the user wishes to return to an NSW conversational partner which he has talked to previously.
- o When the conversational partner's connection has been closed, but buffered output remains to be viewed by the user.

6. STARTING AND STOPPING THE FRONT END

6.1 Starting the Front End

The Front End may be started manually by the user from the UNIX shell, or it may be entered directly at login to UNIX (instead of entering a shell as is usual).

To start the Front End manually from the shell, the user types:

```
% nsw [init]
```

where init is an optional UNIX pathname to the Front End initialization file to be used to define the current Front End configuration. If init is omitted, the Front End will attempt to use fe-init as the default initialization file. Under normal circumstances, the default initialization file will be present, and the init argument may be omitted. If the initialization file is not found, execution is aborted. See Appendix A for details concerning the initialization file.

It is possible to cause a user to enter the Front End directly when he logs in to UNIX by inserting the UNIX pathname for the Front End initialization process into the user's entry in the system password file /etc/passwd. This pathname will replace that of the top-level UNIX shell which is normally present in the

system file.

Refer to the Front End Program Maintenance Manual for information concerning the Front End initialization process.

In order for the Front End to perform its tasks, it requires the use of MSG, the NSW inter-process communication mechanism. The UNIX implementation of MSG contains central MSG processes, through which all inter-host communication occurs, and local MSG processes, which interface the Front End to central MSG. Central MSG processes will be started automatically if there are none present at Front End startup time. A local MSG process will be started automatically for each Front End started.

The Front End indicates that it has correctly started by printing Front End "news" for users, followed by a herald, and the exec mode prompt "NSW: ".

6.1.1 Front End Dispatcher

There is no NSW Front End dispatcher for UNIX systems.

6.2 Stopping the Front End

The Front End is normally terminated by the user issuing a STOP command. In this case, if the Front End was started manually from the shell, the user will return to the shell;

otherwise (i.e., if the Front End was started automatically at login), the user will be automatically logged out of UNIX. The local MSG process associated with each Front End will stop automatically when the Front End stops. The central MSG processes, however, will not stop automatically when all Front Ends have stopped, but must be terminated manually using the UNIX kill command.

6.2.1 Autologout

An Autologout scenario will be initiated if the user's TELNET connection to the UNIX host on which the Front End is running breaks. In this case, if the user is still logged in to NSW, the Front End will stop all active NSW tools in such a way that they may be restarted at a later time, abort all pending Help calls, log out of NSW, and then terminate itself.

7. SUMMARY OF NSW COMMANDS

This chapter presents a summary of the UNIX NSW Front End command language. For each command, a description of its function, command syntax, parameters required, and additional information is given.

7.1 ABORT Command

(Not currently implemented)

Cancels an NSW command which has been issued.

Syntax:

abort <Command-number>

Parameters:

<Command-number> is an integer assigned by the FE and is associated with each command

Notes:

1. The command number is assigned by the FE when the command is issued, and it is made known to the user at that time (see Sections 3.1 and 3.5).

PRECEDING PAGE BLANK-NOT FILMED

7.2 ALTER Command

Used to change the file scopes in effect for NSW filespecs and entry names.

Syntax:

```
alter add <AccessType-Add> <Scopelist>
      or
alter drop <AccessType-Drop> <Scopelist>
```

Parameters:

<AccessType-Add> is 'all', 'copy', 'delete', or 'enter'

<AccessType-Drop> is 'all', 'any', 'copy', 'delete', or 'enter'

<Scopelist> is a list of scopes to be added to or dropped from the scopes currently in effect

Notes:

1. Refer to Section 4.3 for information on scopes.

7.3 COPY Command

Used to create a new file which is a copy of an existing file.

Syntax:

copy <Filespec> <Entrypname>

Parameters:

<Filespec> is a filespec for the file to be copied

<Entrypname> is the entryname for the new copy of file

Notes:

1. Refer to Sections 4.2 and 4.3 for information on filespecs and entrynames, respectively.

7.4 DELETE Command

Used to delete NSW files.

Syntax:

delete <Filespec1> ... <FilespecN>

Parameters:

<FilespecI> is the filespec for the ith file in the list
to be deleted

Notes:

1. The FE will type out the complete file name corresponding to each of the filespecs, and the user will be asked to confirm or abort each delete operation.
2. Refer to Section 4.2 for information on filespecs.
3. Currently the DELETE command will accept only a single filespec.

7.5 DESCRIBE Command

Prints a short description of NSW commands and selected terms.

Syntax:
describe <NSW-item>

Parameters:

<NSW-item> is a command name or an NSW term

Notes:

1. Question mark (?) may be used to obtain a list of the <NSW-items> which may be described. At present, the following FE information may be DESCRIBEd: all NSW commands, and the items 'command-parsing', 'editing', 'entryname', 'filename', 'filespec', 'scope', 'syntax'.

7.6 DISCARD Command

Discards output produced by a completed command or by a Help call.

Syntax:
discard <Command-number>

Parameters:

<Command-number> is an integer assigned by the FE and is associated with each command

Notes:

1. The command number is assigned by the FE when the command is issued, and it is made known to the user at that time (see Sections 3.1 and 3.5).
2. If a Help call is discarded, the call is automatically aborted.

7.7 DISPLAY Command

Displays output produced by a completed command or by a Help call.

Syntax:
display <Command-number>

Parameters:

<Command-number> is an integer assigned by the FE and is associated with each command

Notes:

1. The command number is assigned by the FE when the command is issued, and it is made known to the user at that time (see Sections 3.1 and 3.5).

7.8 HELP Command

Prints a short "help" string of useful information to guide the user in the use of the Front End.

Syntax:
help

Parameters:

none

7.9 JOB Command

Determines and prints the status of a batch job.

Syntax:
job <Job-id>

Parameters:

<Job-id> is an NSW-assigned integer referring to the batch job

Notes:

1. The <Job-id> is made known to the user by the batch tool to which the user has submitted a job.

7.10 LOGIN Command

Creates an NSW user session.

Syntax:
login <PName> <NName> <Password>

Parameters:

<PName> is a project name

<NName> is a node name

<Password> is the node password

Notes:

1. The node password is not echoed by the Front End.

7.11 LOGOUT Command

Terminates an NSW user session by logging user out of NSW.

Syntax:

```
logout
      or
logout,
fast
      or
logout,
move <PName> <NName> <Password>
```

Parameters:

[for unmodified command and 'fast' modification]

none

[for 'move' modification]

<PName> is project name

<NName> is node name

<Password> is node password

Notes:

1. If there are any active NSW tool sessions, the unmodified LOGOUT and LOGOUT with 'move' modification will fail.
2. If the 'fast' modification is used, all interactive NSW tools will be aborted before logging out.
3. If the 'move' modification is used, a new NSW session will be created for the specified node after first logging out the current session.

7.12 NET Command

Moves files between the NSW file system and the file systems of ARPANET hosts. There are three variations of the command. 'Net export' moves a file from NSW file space to an ARPANET host file system; 'import' moves a file from an ARPANET host file system into NSW file space; and 'transport' moves a file between two ARPANET host file systems.

Syntax:

```
net export <Filespec> <Destination file info>
      or
net import <Origin file info> <Entrypname>
      or
net transport <Origin file info> <Destination file info>
```

Parameters:

<Origin file info> and <Destination file info> is information to identify a file not in NSW file space to be used as the origin or destination of a file movement operation. The information consists of: ARPANET host name or number, name of external directory, name of external file, physical type of external file, and external directory password.

<Filespec> is the filespec of a file in NSW file space from which the file movement takes place.

<Entrypname> is the entryname for the file in NSW file space to which the file movement takes place.

Notes:

1. To obtain the external file information, the Front End enters query input mode wherein the user is prompted for the parameters required to specify the external non-NSW file or files.
2. Some ARPANET hosts distinguish between upper and lower-case alphabetics and may treat other characters

in special ways. This should be borne in mind when entering the <origin file info> and <destination file info>.

7.13 NEWS Command

Prints recent news relevant to Front End users.

Syntax:
news

Parameters:
none

7.14 PASSWORD Command

Changes the login password for the current node.

Syntax:
password <Pw1> <Pw2> <Pw3>

Parameters:

<Pw1> is old node password
<Pw2> is new node password
<Pw3> is new node password (identical to <Pw2>)

Notes:

1. To ensure privacy of user passwords, the password parameters are not echoed by the Front End.
2. The user is required to specify the new password twice to minimize the chance of mistyping it.

7.15 QUIT Command

Ends use of an NSW conversational partner.

Syntax:

```
quit abort <Tool-name>
      or
quit shell <CP-name>
      or
quit telnet <CP-name>
      or
quit terminate <Tool-name>
```

Parameters:

<Tool-name> is an NSW tool-instance name

<CP-name> is the name assigned by the user for a TELNET connection or UNIX subshell conversational partner

Notes:

1. The TERMINATE command is equivalent to QUIT SHELL, QUIT TELNET, and QUIT TERMINATE and will eventually replace them (see Sections 5.3.1 and 7.24).
2. For the QUIT ABORT command, files are not delivered from the NSW tool's workspace to the NSW system.

7.16 RENAME Command

Changes the name of an NSW file.

Syntax:

```
rename <Filespec> <Entryname>
```

Parameters:

<Filespec> is the filespec for the file to be renamed

<Entryname> is the new name for the file

7.17 RESUME Command

Switches from exec mode to tool mode, connecting the user to the specified NSW conversational partner.

Syntax:

```
resume <CP-name>
```

Parameters:

<CP-name> is name of NSW conversational partner to be resumed

Notes:

1. Refer to Section 5.2 for information on creating NSW conversational partners.

7.18 SEMAPHORE Command

Manipulates and queries the status of the semaphore for an NSW file.

Syntax:

```
semaphore read <Filespec>
           or
semaphore set  <Filespec>
           or
semaphore unset <Filespec>
```

Parameters:

<Filespec> is the filespec for the file whose semaphore is being referenced

Notes:

1. Refer to 4.2 for information on filespecs.

7.19 SET Command

Sets local FE parameters.

Syntax:

```
set return-mode <New-RetMode>
```

Parameters:

<New-RetMode> is the Front End return mode desired and may be either 'immediate' or 'deferred'

Notes:

1. Refer to Chapter 3 for detailed information on FE return modes.

7.20 SHELL Command

Creates a UNIX shell as a new conversational partner. The new shell runs as an inferior process to the Front End (i.e., as a "subshell").

Syntax:

shell <CP-Name>

Parameters:

<CP-Name> is the name assigned by the user to the subshell conversational partner

Notes:

1. The user may terminate the subshell by issuing a TERMINATE command.
2. The Front End will notify the user when the subshell is ready to use.
3. The user must use a RESUME command to switch to the subshell after it is ready to use.

7.21 SHOW Command

Prints information about active commands, conversational partners, files, node, scopes, return modes, or user state.

Syntax:

```
show active {commands | tools}
      or
show files <AccessType> <FilespecI> ... <FilespecN>
      or
show node
      or
show scopes <AccessType>
      or
show status <Status>
```

Parameters:

[for 'show files']

<FilespecI> is filespec of NSW fileI to be shown

[for 'show scopes']

<AccessType> is 'all', 'any', 'copy', 'delete', or
'enter'

[for 'show status']

<status> is desired status information desired and may be
either 'return-mode' or 'user-state'

Notes:

1. Typing control-T is equivalent to issuing 'show status user-state'.

7.22 STOP Command

Terminates Front End and exits.

Syntax:
stop

Parameters:

none

Notes:

1. This command is always active and will operate even if the user is logged into NSW and/or has conversational partners.

7.23 TELNET Command

Creates a TELNET connection to a specified host as a new conversational partner.

Syntax:

telnet <Host-name or host-number> <CP-name>

Parameters:

<Host-name or host-number> is the ARPANET host name or number for the desired host

<CP-name> is the name under which the FE handles the TELNET connection conversational partner

Notes:

1. The name for the TELNET conversational partner is assigned by the user and may be any character string.
2. The Front End will notify the user when the TELNET connection is ready to use.
3. The user must use a RESUME command to switch to the connection after it is ready to use.

7.24 TERMINATE Command

Ends use of an NSW conversational partner. The precise manner in which the conversational partner is terminated depends upon its type (see Section 5.3.1).

Syntax:
 terminate <CP-name>

Parameters:

<CP-name> is name of NSW conversational partner to be terminated.

Notes:

1. If an NSW tool is terminated, all relevant files in its workspace are first delivered, and the NSW file catalog is updated before the tool is stopped.
2. The TERMINATE command is synonymous with QUIT SHELL, QUIT TELNET, and QUIT TERMINATE, and will eventually replace them.

7.25 USE Command

Starts up a NSW interactive tool as a Conversational Partner.

Syntax:
use <Tool-name>

Parameters:

<Tool-name> is name of NSW tool to use

Notes:

1. The Front End will notify the user when the tool is ready to use.
2. The user must use a RESUME command to switch to the tool after it is ready to use.
3. When using batch tools and management tools, it is preferable to switch into deferred return mode prior to issuing the USE command.

APPENDIX A FRONT END CONFIGURATION CONTROL

Most users need not concern themselves with the information in this appendix, since in normal usage, the default initialization file will be present. Some familiarity with the Program Maintenance Manual for the UNIX Front End is required to understand what follows.

The configuration of the Front End in use is controlled by the Front End initialization file, which is read in at initialization time. The structure of the initialization file is:

line 1: <UNIX pathname of FE User Process> <LogFlag> <VrbFlag>
line 2: <UNIX pathname of FE Protocol Process> <LogFlag> <VrbFlag>
line 3: <UNIX pathname of FE Tool Process> <LogFlag> <VrbFlag>
line 4: <UNIX pathname of FE text file>
line 5: <UNIX pathname of FE text index file>

<LogFlag> and <VrbFlag> are 16-digit bitstrings governing, respectively, Front End event logging and typeout of diagnostic information at run time. They are described next.

A.1 Event Logging

It is possible to log certain classes of events during the execution of the Front End by turning on bits of <LogFlag>. Each event is logged in a file whose file name begins with the UNIX process id, has a Front End process name abbreviation (i.e., "PP" for Protocol Process, "TP" for Tool Process, and "UP" for User Process), followed by 'log+'; for example, '34TPlog+.'

The following bits of <LogFlag> are defined, where the octal value of a bit is given at the left, followed by its effect:

- 01 log event seen by the user
- 02 log inter-process communication
- 04 log NSW system response
- 010 log network activity
- 020000 do not remove tool table file at end of session
(meaningful only for FE User Process)
- 040000 terminate process with core dump
- 0100000 display error log messages (see Appendix
A.3) on terminal, as well as writing
them to error log file

A.2 Diagnostic Typeouts

It is possible to cause the Front End to give diagnostic typeouts in the course of its execution, usually for debugging or

verification, by turning on bits of <VrbFlag>. The following bits of <VrbFlag> are defined, where the octal value of a bit is given, followed by its effect:

- 01 type out all NSWTP messages
- 02 type out inter-process communication information
- 04 type out MSG-related information
- 010 enable xon-xoff, i.e., the control-S/control-Q terminal flow control (meaningful only for FE User Process)

A.3 Example of Initialization File

The following is an example of a Front End initialization file used to test the Front End at BBN:

```
/usr/nswmsg/bin/user 0120017 0
/usr/nswmsg/bin/ptcl 0100017 0
/usr/nswmsg/bin/tool 0100017 0
/usr/nswmsg/fe-text
/usr/nswmsg/fe-txtindex
```

For all Front End processes, <VrbFlag> is zero, i.e., no diagnostic messages will be printed at run-time. For all three Front End processes, event-logging for all (four) possible classes of events is enabled, i.e., the rightmost four bits of <LogFlag> are turned on; and error log messages will be displayed on the user's terminal as well as being placed into an error log file (the leftmost bit of <LogFlag> is turned on). For the Front

End User Process, the additional bit turned on suppresses the deletion of the active tool table file used by the Front End at run-time.

A.4 Error Log Files

The Front End logs all errors encountered in the course of its execution in error log files. The file name of an error log file consists of a Front End process name abbreviation, followed by 'err+'; for example, 'PPerr+'. There are three error log files, one for each Front End process. Output directed to an error log file may also be displayed on the user's terminal by turning on a bit of <LogFlag> (see Appendix A.1) above.