

4. SYSTEM USE ISSUES

This chapter provides basic system information to get you started in NoteCards on your Sun Workstation after your system administrator has installed NoteCards.

Site Initialization File

When NoteCards starts, it reads in the NoteCards site initialization file, `init.NoteCards`. This file sets the pointer to fonts, site parameters and the like. You need to edit this file if you move the system files from their standard locations. Initialization files are discussed in detail in Appendix D. Unless there are compelling reasons for moving files from their preset directories, we strongly counsel against moving them.

Starting NoteCards

Running NoteCards requires both an executable (`lde`) and a virtual memory image (a `sysout`). Generally, NoteCards requires a host access key matched to the host ID of the processor on which it is running. In addition, if NoteCards is to perform direct network communication (for Xerox PUP or XNS protocols), it needs a separate program, `ldeether`, to initialize the Ethernet. To run NoteCards on the Sun Workstation, follow these steps.

1. Exit `sunttools` or any other windowing system. NoteCards provides its own window system and must not run simultaneously with others.
2. Kill all your user processes (these have console as the control `tty`). Check to make sure you have killed any "selection_ `svc`" process.

Note: If you do not perform this step, certain error messages from UNIX (e.g., "file system full") cause those processes to print to the console, resulting in scrolling of the display .

3. Find out which directory the software is stored on and add that directory to your path if necessary:

```
prompt# set path=($path/usr/local/lde/install.sunosx.machinex)
```

Replace `install.sunosx.machinex` with one of the following:

```
install.sunos3.mc68020
install.sunos4.mc68020
install.sunos4.sparc
```

depending on which machine and SunOS release you are using. You can also add this to your `.login` or `.cshrc` file.

4. Invoke NoteCards by typing the name of the program, e.g.,

```
prompt% lde NoteCards.sysout -k 'this host-key'
```

If you are using either Xerox XNS or Xerox PUP Ethernet protocols, type instead

```
prompt% ldeether NoteCards.sysout -k 'this host-key'
```

This, in turn, runs `lde` and allows it to use the Ethernet directly. As explained in Chapter 3, this command can be replaced by an alias or a shell script.

Where Notecards Looks for Your Sysout

The system searches for the sysout to be used in the following order.

- *command line*

The name of the sysout file can be given on the command line when starting NoteCards; e.g.,

```
prompt% lde NoteCards.sysout -k 'this host-key'
```

- `LDESOURCESYSOUT`

If no sysout file name is given on the command line, the value of the environment variable `LDESOURCESYSOUT` is used as the name of the sysout file.

- `~/lisp.virtualmem`

Finally, NoteCards looks for a file `lisp.virtualmem` on the home directory of the current user.

Once NoteCards has started, it allocates virtual memory space to hold the working environment, maps the screen frame buffer into its address space, and reads the sysout. As pages from the sysout are read into the display area, they appear on the screen. If the sysout was written while running on a machine with a different size display, the image will appear garbled. This process takes several seconds. After NoteCards starts running, it readjusts the display to the current size.

Exiting NoteCards and Saving State

This section explains how to save a NoteCards system, leave NoteCards, and restart.

On the Sun, `lde` is an ordinary UNIX program that allocates a 45-megabyte data area, reads into that area several megabytes of data (the NoteCards sysout), and modifies it there. Under UNIX, the program's data requirements (which include the sysout) are handled by UNIX; all NoteCards does is modify in virtual memory a copy of your original sysout file. UNIX, transparently to NoteCards, handles all real memory swapping. This has several consequences related to starting, saving, and restarting sysouts.

When you exit NoteCards and save the memory image, you are writing a copy of the virtual memory file, which UNIX has been managing, out to disk. Unlike more usual programs, this file preserves the complete state of the system as it exists when you exit. This means that you can exit a NoteCards session in the middle of editing a file, without saving the file, and when you restart the system the file will be as you left it. However, we always advise you to save your files before you exit.

The virtual memory file that the command **Save image & Exit** writes is normally `~/lisp.virtualmem` (i.e., the file `lisp.virtualmem` on your home directory). However, you can override this default by setting the environment variable `LDEDESTSYSOUT`. For example, you might want to keep virtual memory images on your directory `/users/smith`. You would accomplish this by using the C-Shell command `setenv`. For example:

```
prompt% setenv LDEDESTSYSOUT "/users/smith/lisp.virtualmem"
```

You can also rename the output file as well as its destination. For example, if you wanted the output file to be called `NoteCards.vm`, you would issue the command,

```
prompt% setenv LDEDESTSYSOUT "/users/smith/nc/NoteCards.vm"
```

Place this command in your `.cshrc` or `.login` file to have it executed every time you log in.

In another situation, for instance during a demonstration, you might not want the memory image saved. In this case, you can reset `LDEDESTSYSOUT` to `/dev/null`. For example:

```
prompt% setenv LDEDESTSYSOUT "/dev/null"
```

Cursor tracking interferes with writing out the screen bitmap as part of the `NoteCards` memory image. For this reason, `NoteCards` takes the cursor down before saving a virtual memory image. When this happens, the message

```
Saving VMem, taking mouse down
```

appears in the prompt window, and cursor tracking is disabled.

Because the virtual memory file need not already exist to run `NoteCards`, you can receive the following file errors :

```
File-System-Resources-Exceeded  
Protection-Violation  
File-Wont-Open
```

When working in `NoteCards` on a Sun workstation, you should periodically load a fresh `sysout`. Older `NoteCards` `sysouts` don't run as well as fresh `sysouts` due to a number of factors such as fragmentation of memory, increased working set, and more objects taking up various spaces.

Executing the commands **Save image & Exit** or **Save VM** saves the virtual memory image to a file, which preserves your working state. However, it is important to realize that each time you do this, the file grows in size. Over the course of use, a virtual memory file can easily grow from its original 7-megabyte size to 9 or 10 megabytes.

Keyboard Interpretation

This section describes how NoteCards interprets the Sun Type 3, and Type 4 keyboards. These key assignments were chosen to maximize compatibility with both the Xerox workstation keyboard and the normal Sun keyboards. You can attach a Sun Type 3 or Type 4 keyboard template, which also shows the NoteCards keyboard assignments, to your Sun Type 3 or Type 4 keyboard. Both templates are included with your NoteCards documentation set.

Sun Type 3 Keyboard

Figures 4-1 through 4-3 show NoteCards's key assignments for the Sun Type 3 keypads.

Stop	Again
Help	Undo
Same	Move
Open	Copy
Find	Delete

**Figure 4-1. Sun Type 3
Left Key Pad**

Num Lock	Scroll Lock	Break
7 Home	8 ↑	9 PgUP
4 ←	5	6 →
1 End	2 ↓	3 PgDN
Ins	DOIT	Caps Lock

**Figure 4-2. Sun Type 3
Right Key Pad**

Center	Bold	Italic		Case		Strikeout		Underline		Super Sub	Large Small	Margin	Back Word
Esc	! 1	@ 2	# 3	\$ 4	% 5	↑ 6	& 7	* 8	(9	) 0	← -	+ =	 \ ~ ;
Tab	Q q	W w	E e	R r	T t	Y y	U u	I i	O o	P p	{ [	}]	Backspace
Ctrl	A a	S s	D d	F f	G g	H h	J j	K k	L l	:	; "	' ,	Return
Shift		Z z	X x	C c	V v	B b	N n	M m	< ,	> .	? /	Shift	LF
Caps	Meta	Space										Expand	Next

Figure 4-3. Sun Type 3 Center Key Pad Interpretation

Sun Type 4 Keyboard

Figures 4-3 through 4-5 show the keyboard and the left and right key pads for the Sun Type 4 keyboard.

Stop	Again
Props	Undo
Same	Copy
Open	Move
Find	Delete
Help	

*Figure 4-3. Sun Type 4
Left Key Pad*

Break	PrSc	scroll lock	num lock
=	/	*	-
7 Home	8 ↑	9 PgUP	+
4 ←	5	6 →	
1 End	2 ↓	3 PgDN	DOIT
Ins		Del	

*Figure 4-4. Sun Type 4
Right Key Pad*

F1 Center	F2 Bold	F3 Italic	F4 Case	F5 Strike	F6 Under	F7 Super	F8 Large	F9 Margin	F10	F11	F12	 \ _	Delete Word	
Esc	! 1	@ 2	# 3	\$ 4	% 5	↑ 6	& 7	* 8	(9	) 0	← -	+ =	Back Space	
Tab	Q q	W w	E e	R r	T t	Y y	U u	I i	O o	P p	{ [	}]		Return
Ctrl	A a	S s	D d	F f	G g	H h	J j	K k	L l	: ;	" ,	~ '		
Shift		Z z	X x	C c	V v	B b	N n	M m	< ,	> .	? /	Shift	LF	
Caps	Meta	Left Spc	Space									Right Spc	Expand	Next

Figure 4-5. Sun Type 4 Center Key Pad Interpretation

Note: In SunOS 4.0, the NEXT (ALT/GRAPH) key on the Type 4 keyboard is inaccessible. Later versions of SunOS are believed to fix this.

Console Messages

Under SunOS, various system processes and operations attempt to log information on the console. Since NoteCards takes over the screen, console messages are redirected; a background process in NoteCards causes them to appear in the prompt window.

When NoteCards is run remotely (i.e., not from the console), most console, or operating system, messages are printed in the prompt window. Some messages may also appear in the middle of the NoteCards display screen or on the remote tty. This occurs because UNIX is often confused about where to send messages. NoteCards is normally run remotely only for debugging purposes.

CAUTION

Critical UNIX system processes can hang if the buffer holding console messages fills. There are two points to watch for:

NoteCards uses a temporary file, `/tmp/XXXX-lisp.log`, where XXXX is the user's login name, to buffer console messages before printing them. Do not delete this log file while NoteCards is running. If the log file is deleted, console messages can no longer be printed in the NoteCards prompt window.

The process `\10MBWATCHER`, used to watch for Ethernet packets, reads console messages. Thus, you should never kill the `\10MBWATCHER` process, even if you don't use the Ethernet capabilities of NoteCards. If you do kill the `\10MBWATCHER` process, console messages cannot be printed and the operating system can hang.

File Compatibility

Sysout Compatibility between Sun and Xerox Workstations

Sysout files created on Xerox workstations can be moved to Sun Workstations. However, sysout files created on Sun Workstations cannot be moved to Xerox workstations.

File Compatibility between Sun and Xerox Workstations

Some care must be taken in moving files to and from Xerox workstations and services, since the default end-of-line convention in UNIX is to terminate lines with the line feed (LF) character, while traditionally Xerox systems have terminated lines with the carriage return (CR) character. In particular, if you use some other file transfer mechanism, such as FTP or Kermit, be careful to transfer `.TEDIT` and `.NOTEFILE` files in binary mode.

In NoteCards on the Sun Workstation, the default end-of-line convention for all text files is line feed (LF). The default end-of-line convention for all binary files is carriage return (CR); this is because CR is used internally in the system.

When working with the Xerox protocols XNS and PUP as well as Sun NFS, it is important that you use the suffix `.TEDIT` on all TEdit files, the suffix `.SKETCH` on all Sketch files, and the suffix `.NOTEFILE` on all notefiles for NoteCards to treat the files properly when moving them from one device to another.

TEdit files not transferred in binary mode frequently get "Font not found." error messages when you try to open them. The carriage returns, which are represented by the number 13, are reinterpreted as font information and the system breaks trying to look for a font of size 13. These files are irrecoverably lost.

Using SunOS Files from NoteCards

You can access any mounted SunOS file system directly from NoteCards. The mounted file system is available as an I/O device of the NoteCards environment. This file system appears as the local disk of NoteCards, even though it may be a remotely mounted file system of networked Sun file servers.

Many of the file devices to which the NoteCards environment can talk, including PUP, XNS file servers, the {CORE} device, and others, have facilities that are not directly supported by SunOS. For example, many file systems have file version numbers and case insensitive file search conventions.

NoteCards on the Sun Workstation has two distinct "host" names that can be used to access the SunOS file system. These host names are provided for compatibility with existing applications and tools. They also simultaneously allow natural interaction with the SunOS file system. The names are:

- {DSK} On the Sun Workstation, the {DSK} device allows you to access the file system using similar conventions to those used for Xerox services. In particular, {DSK} files have version numbers; {DSK} file name recognition also ignores the case of letters.
- {UNIX} The {UNIX} device lets you use the mounted file systems with the normal naming conventions of the SunOS file system. {UNIX} files do not have version numbers, and the file name recognition treats lowercase letters as distinct from their uppercase equivalents.

File streams can be opened or closed on both devices. The reason for having both devices is to more easily support the running of applications that were originally developed using Xerox services, while still allowing new applications to interact more naturally with UNIX.

File Naming Conventions

In NoteCards, a file name (pathname) consists of a collection of fields: the *host*, *directory*, *name*, *extension* and *version*. These fields are optional. The standard NoteCards syntax for these fields is:

```
{host}<directory>name.extension;version
```

The *directory* field can be a directory path consisting of a sequence of directory and subdirectory components. Slashes (/) and right angle brackets (>) can be used to delimit a directory name; there is no distinction made between them. However, you should use one delimiter or the other consistently. What the directory returns is unpredictable if you mix delimiters. Note that square brackets ([]) are not acceptable as directory delimiters.

Duplicated directory delimiters are treated as a single delimiter. Thus, the following two file names specify the same file:

```
{DSK}<LISP>USERS>FOO.;1  
{DSK}</LISP/USERS/>FOO.;1
```

Common {DSK} and {UNIX} Naming Conventions

- To include a special character (e.g., > or ;) in a file name, precede it with a single quote ('). To include a single quote in a file name, precede it with another single quote. You can quote any of these characters: <, >, ;, ~, and a period (.). The following examples show how the single quote notation on {DSK} and {UNIX} is used.

{DSK} Name From Lisp

```
foo'>bar.baz;1
foo';bar.baz;1
foo''bar.baz;1
```

File Name From SunOS

```
foo>bar.baz
foo;bar.baz
foo'bar.baz
```

- {DSK} and {UNIX} do not allow you to use either the slash (/) or the NUL character in file names. Thus, you cannot name files containing these characters.
- Both {DSK} and {UNIX} can handle the following characters, which were defined as special characters in Medley Release 2.0: backslash (\) and tilde (~).
- {DSK} and {UNIX} can distinguish between a file name with a period at the end (e.g., foo.) and a simple file name (e.g., foo). The final period is preceded with a single quote, as shown in the following example:

{DSK} Name From Lisp

```
foo.;1
foo'..;1
```

File Name From SunOS

```
foo
foo
```

- On {DSK} and {UNIX}, the C-Shell and SunOS directory notations (~, ., and ..) are supported in the Notecards directory specification. The tilde character (~) is allowed at the very beginning of the directory specification of a pathname. A combination of relative path specifiers (~, ., ..) is supported. The tilde character corresponds to the user's home directory at login. The period (.) corresponds to the current working directory. Two periods (..) indicates the parent of the current working directory.
- File names are returned by the system (e.g., INFILEP) in more canonical form. The function which returns the full file name returns it in the canonical form, as in {DSK}<usr>etc rather than {DSK}/usr/etc/. This change will make some tools which depend on the conventional file name representation described in the *Interlisp-D Reference Manual* work correctly on the Medley file system (e.g., COPYFILES).

{DSK} Naming Conventions

File access to the {DSK} device goes through the following file name transformation when actually accessing the SunOS file system:

- Mixed case letters are read as such.
- File name searches are done case-sensitive first; if a match is not found, the system then does a case-insensitive search.
- A left angle bracket (<) is translated to a slash (/), the delimiter for the root directory.
- {DSK} supports relative pathnames. You can specify relative pathnames by omitting a slash (/) or left angle bracket (<) as the first character in the directory field. For example:

`{DSK}foo.fee` and `{DSK}~/foo.fee` are relative to your UNIX home directory (`~/foo.fee`).

`{DSK}./foo.fee` is relative to your current working directory (`./foo.fee`).

`{DSK}../foo.fee` is relative to the parent directory of your current UNIX working directory (`../foo.fee`).

The Medley 2.0 `{DSK}` device supports the notation in which the three meta characters (`'`, `...`, and `~`) are used together, as shown in the following example:

```
{DSK}~/../tom/foo.c
```

In this example, the `{DSK}` device interprets `tom` as one of the subdirectories of the parent directory of the user's home directory.

`{DSK}` also supports the tilde-name (`~name`) convention. `{DSK}` interprets `{DSK}~tom/foo.c` as a file named `foo` on `tom`'s home directory. In this notation, the user name is case-sensitive (e.g., `~tom` and `~Tom` are treated as different users).

Version Numbering

The UNIX file system does not support version numbers in file names; `{DSK}` emulates versions with a naming convention. This section explains how `{DSK}` version numbers are represented in the SunOS file system.

- When you create a completely new file, it appears in the SunOS file system without a version number.

{DSK} Name From NoteCards	File Name From SunOS
----------------------------------	-----------------------------

<code>bar.baz;1</code>	<code>bar.baz</code>
------------------------	----------------------

- When you create (from NoteCards) a file with a version other than 1, NoteCards adds version numbers to that file name, as a trailing number between tildes, e.g., "`myfile.~12~`" for the twelfth version of `myfile`.

The following shows some examples of equivalent file names in NoteCards and SunOS.

{DSK} Name From NoteCards	File Name From SunOS
----------------------------------	-----------------------------

<code>bar.baz;1</code>	<code>bar.baz.~1~</code>
<code>bar.baz;2</code>	<code>bar.baz.~2~</code>
<code>bar.;23</code>	<code>bar.~23~</code>

- NoteCards always maintains a versionless file which is hard-linked to the highest extant version of the file (i.e., they are two names for the very same file). This file name does not appear in the `{DSK}` directory listing.

From {DSK}	From SunOS
-------------------	-------------------

<code>foo.c;15</code>	<code>foo.c (hard linked with <code>foo.c.~23~</code>)</code>
<code>foo.c;23</code>	<code>foo.c.~15~</code>
	<code>foo.c.~23~</code>

Similarly, a file created in UNIX with no version number is treated by `{DSK}` as the highest version.

- When you create a new version of a file, the versionless-file link is broken, and the versionless file is hard-linked to the new highest version.

From {DSK}

foo.c;15
foo.c;22
foo.c;24 (new file)

From SunOS

foo.c (hard linked with foo.c.~24~)
foo.c.~15~
foo.c.~22~ (no link with foo.c)
foo.c.~24~ (new file, link from foo.c)

- When you delete the highest version of a file, the versionless file is also deleted. If any older versions of the file remain, a new link is created from the versionless name to the highest version extant. For example, if you have the files

From {DSK}

foo.c;1
foo.c;2

From SunOS

foo.c (linked to foo.c.~2~)
foo.c.~1~
foo.c.~2~

and you delete foo.c;2 from {DSK}, the resulting files are:

From {DSK}

foo.c;1

From SunOS

foo.c (linked to foo.c.~1~)
foo.c.~1~

- When you rename a file it works the same as deleting the file under the old name then creating it under the new name. For example, if you have the following {DSK} files,

From {DSK}

foo.c;1
foo.c;2
fee.c;1
fee.c;2

From SunOS

foo.c (linked to foo.c.~2~)
foo.c.~1~
foo.c.~2~
fee.c (linked to fee.c.~2~)
fee.c.~1~
fee.c.~2~

and you rename "foo.c" to "fee.c", your renamed {DSK} files and the linked SunOS files would appear as

From {DSK}

foo.c;1
fee.c;1
fee.c;2
fee.c;3

From SunOS

foo.c (linked to foo.c.~1~)
foo.c.~1~
fee.c (linked to fee.c.~3~)
fee.c.~1~
fee.c.~2~
fee.c.~3~ (renamed file)

- When a file has a name suffix that is not a valid version number (e.g., myfile.~12x~), the suffix is regarded as part of the file name.

From {DSK}

myfile.~12x~;1

From SunOS

myfile.~12x~

Pathnames

A pathname on {DSK} is always case insensitive. When you specify a file, the {DSK} device handler first searches for the file with the specified name. If no such file is found, it then searches for a file with the same spelling but different case.

If a pathname on {DSK} has no directory specification, a tilde-slash combination (~/) is used, i.e., the NoteCards directory specification {DSK}foo is the equivalent of UNIX ~/foo.

{UNIX} Naming Conventions

For the {UNIX} device, file name translation takes place only on the directory. An initial left angle bracket (<) is treated as if it were an initial slash (/); both signify a path relative to the SunOS file system root directory; if there is no initial left angle bracket or slash, the directory is relative to the current working directory. Initially this is the working directory where NoteCards was started; you can change it using the CHDIR function, described below. Tilde (~) translates to your home directory.

For example,

```
{UNIX}myfile/abc
```

means the file abc on the ./myfile/ directory.

The {UNIX} device does not recognize version numbers, does not return them, and ignores them for recognition.

No case translation or recognition is done; upper- and lowercase letters are treated as distinct.

Examples:

{UNIX} Name From NoteCards	File Name From SunOS
<foo>fee>bar.baz;1	/foo/fee/bar.baz;1
<foo>fee/bar.;1	/foo/fee/bar.;1
<foo/fee>	/foo/fee/
</foo/fee/>	/foo/fee/
/foo/fee/bar.~1~	/foo/fee/bar.~1~
/foo/fee/	/foo/fee/

In the first two examples the ;1 is treated as part of the file name, not the version number. Note in the last two examples that translation is not done.

Directories

In places where NoteCards expects a directory name, {UNIX} paths must end with a slash (/).

Directory Enumeration

You cannot use the wildcard character, asterisk (*), in subdirectories for either {DSK} or {UNIX} devices. For example, typing

```
{DSK}/users/x*/foo
```

into the FileBrowser prompt window will return

```
No files in group {DSK}<users>x*>foo.*;*
```

Enumeration of files in directories differs between {DSK} and {UNIX} devices. On the {DSK} device, a versionless file which has a link to the highest version file is not enumerated in a directory.

On the {UNIX} device, all files are enumerated in a directory. For instance, if the following SunOS files, linked with `foo.c.~2~` exist

```
foo.c
foo.c.~1~
foo.c.~2~
```

the {DSK} directory enumeration, would look like this:

```
> (DIRECTORY ' {DSK}/users/venue/*)
({DSK}/users/venue/foo.c;1
{DSK}/users/venue/foo.c;2)
```

The {UNIX} directory enumeration, on the other hand, would look like this:

```
> (DIRECTORY ' {UNIX}/users/venue/*)
({UNIX}/users/venue/foo.c
{UNIX}/users/venue/foo.c.~1~
{UNIX}/users/venue/foo.c.~2~)
```

Directory Creation

{DSK} When you write a new file on {DSK}, if the directory named in a pathname does not exist, the {DSK} device handler creates the directory automatically. This feature is provided for compatibility with Xerox services.

If you try to connect to a nonexistent directory (using the Directory Connector tool), NoteCards beeps and prints the following message in the system prompt window

```
Not a valid directory name.
```

{UNIX} The {UNIX} device does not support such directory creation. An attempt to create a file on a nonexistent directory results in an error. NoteCards beeps and prints the following message in the system prompt window.

```
System call error: stat errno = 2
No such file or directory.
```

Directory Deletion

Neither {UNIX} nor {DSK} support automatic directory deletion. To delete a directory you must use the SunOS C-Shell command `rmdir`.

Open File Limit

The number of simultaneously open {DSK} and {UNIX} files must fall within the SunOS limits for a process. For OS 3.4, this number of open files may be configured, with 30 as the maximum permissible number of open files per process. This means that it is not possible to have more than 30 files open for a process, minus whatever files NoteCards has open for its own use, at any one time in the NoteCards system. If you try to open too many files, the system call error number 24,

```
Too many open files
```

appears in the prompt window.

For OS 4.0, the maximum number of files/processes that can be open at one time is 64, unless your kernel is configured otherwise.

Default Pathname

If no path is given, the {DSK} device defaults to the user's home directory, tilde-slash (~/). The {UNIX} device defaults to the current working directory. This current working directory can be changed with the Directory Connector tool. Note that the current working directory is also used to resolve the interpretation of the period (.) and double period (..) specifications at the beginning of a {DSK} pathname. The current working directory is the directory you were in when you started NoteCards, not your connected directory in NoteCards.

File Attributes

This section describes how the various file attributes are treated by NoteCards on the Sun Workstation and what they translate to in SunOS. For more discussion of file attributes, see Chapter 14 for a description of the FileBrowser.

WRITEDATE and CREATIONDATE

[File Attributes]

The date is reset to the current time whenever the contents of a file are modified. Since UNIX does not naturally support more than one date for file modification, the WRITEDATE and CREATIONDATE are treated identically by NoteCards, and by the {DSK} and {UNIX} devices.

TYPE

[File Attribute]

Returns the TYPE property of files; normally either TEXT or BINARY. However, UNIX does not distinguish between TEXT and BINARY files. Normally, programs will infer the type by the file extension. If no file extension is given, the default file type TEXT is used.

SIZE

[File Attribute]

Returns the file size. For compatibility with other NoteCards environments running on Xerox workstations and servers, the SIZE attribute is computed as the length of the file (in bytes) divided by 512 (rounded up).

File System Errors

Several types of errors may occur in the NoteCards file system.

When a remotely mounted file system or the NFS service is down, any attempt to access a file on the file system eventually results in a timeout error. The following error message is printed in the prompt window:

```
File access timed out
```

If the mounted device is mounted with the "hard" option, NoteCards continues to wait until the mounted device responds. During that time, user interrupts are not available. We recommend mounting remote file systems with the "soft" option. You can use the UNIX command `/etc/mount` to check the current mount options.

The following error messages may appear when there are NoteCards file errors:

```
Not owner
Device error:
Protection-violation
File-won't-open
Too-Many-Files-Open
File too large
File-System-Resources-Exceeded
Connection timed out
No-Such-Directory
Bad Host Name
FS-RENAMEFILE-SOURCE-COULDNT-DELETE
```

Another type of error occurs occasionally when the file system prints an incorrect message such as

```
File not found
```

A more accurate console message appears, at the same time, in the prompt window. This message appears in the SunOS message form

```
System call error:  open errno=13  Permission denied
```

See the *UNIX Interface Reference Manual*, Intro (2), for descriptions of all OS system call messages.

[This page intentionally left blank]