Solaris Installation and Administration

Varian NMR Spectrometer Systems
With VnmrJ Software

Pub. No. 01-999270-00, Rev. A0604

Solaris Installation and Administration

Varian NMR Spectrometer Systems
With VnmrJ Software

Pub. No. 01-999270-00, Rev. A0604

VARIAN

Solaris Installation and Administration
Varian NMR Spectrometer Systems
With VnmrJ Software
Pub. No. 01-999270-00, Rev. A0604

Revision history:
A0604 – Initial release for VnmrJ 1.1D

Applicability:
Varian NMR spectrometer systems with Sun workstations
running VnmrJ software

By Rolf Kyburz (rolf.kyburz@varianinc.com)
Varian International AG, Zug, Switzerland,
and
Gerald Simon (gerald.simon@de.varioan.com)
Varian GmbH, Darmstadt, Germany

Additional contributions by Frits Vosman, Dan Iverson, Evan Williams, George Gray,
Technical writer: Mike Miller, Everett Schreiber
Technical editor: Dan Steele

Overview of Contents

Disclaimer 15
Foreword 15
Acknowledgments 15
Conventions in This Manual 16
Chapter 1. Solaris 9 Installation 17
Chapter 2. Solaris 8 Installation 29
Chapter 3. Solaris Patch Installation and Error Messages 37
Chapter 4. Solaris Reference Information 41
Chapter 5. Files, Permissions, and Owners 53
Chapter 6. File Security, Repair, and Archiving 65
Chapter 7. UNIX Commands 73
Chapter 8. Software Operating Levels 83
Chapter 9. Shell Scripts 95
Chapter 10. UNIX System Customization 111
Chapter 11. Disk Organization 115
Chapter 12. Networking via Ethernet 129
Chapter 13. Other Networking Options 157
Chapter 14. Text Editing 163
Chapter 15. Using Tapes and Floppy Disks 171
Chapter 16. Output Devices, Terminals, Serial Ports 185
Chapter 17. X Window System Operation 193
Chapter 18. Periodic Maintenance and Troubleshooting 201
Annotated Bibliography 221
Glossary of UNIX, Sun, and VnmrJ Terms 229
Index 277

Table of Contents

Disclaimer 15

Foreword 15

Acknowledgments 15

Conventions in This Manual 16

Chapter 1. **Solaris 9 Installation** 17

1.1 Solaris 9 Compatibility 17
1.2 Solaris 9 Quick Installation 18
1.3 Booting from the Solaris 9 CDs or DVDs 18
Boot Failure from DVD 18
Bootup from the CD-ROM 19
1.4 Solaris 9 Detailed Installation 20
Language and Locale 20
Identify This System 20
Time Zone Information 21
Configuring Solaris for Installation 22
Interactive Installation 22
1.5 Additional Solaris 9 Software 26
1.6 Sun Management Center 3.0 Platform 4 Update 27
1.7 Adding the VNMR Icon to the CDE Tool Bar 27

Chapter 2. **Solaris 8 Installation** 29

2.1 Solaris 8 Compatibility 29
2.2 Solaris 8 Quick Installation 30
2.3 Solaris 8 Detailed Installation 31
Language and Locale 31
Identify This System 31
Time Zone Information 32
Configuring Solaris for Installation 33

Chapter 3. **Solaris Patch Installation and Error Messages** 37

3.1 Patches 37
Preparing for Patch Installation 37
Installing Solaris Patches 38
3.2 Error Messages 39
Device Busy 39
“Can’t connect to X11 window server” 39
Locator Shows Only Error 39
Missing Fonts 39

Chapter 4. **Solaris Reference Information** 41

4.1 Mounting a CD-ROM 41

Volume Manager 41
Mounting a Local CD-ROM Drive 41
Mounting a Remote CD-ROM Drive 42
4.2 Shutting Down the Sun Computer 44
4.3 Opening a Host or Terminal Window 45
4.4 System Requirements 46
Sun Requirements 46
Solaris Software Group Space Requirements 46
4.5 Disabling and Reenabling a X453A Thinnet Coax Ethernet Board 47
Disabling the X453A Thinnet Coax Ethernet Board 47
To Reenable the X453A Thinnet Coax Second Ethernet Board 48
4.6 Collecting System and Network Information for Solaris 49
Host Name 49
Primary Network Interface 49
IP Address 50
Name Service Type 50
Domain Name 50
Name Server Host Name and IP Address 50
Enable IPv6 51
Proxy Server (Sun Blade 100 and Blade 1000) 51
Subnet Mask 51
Setting the Time Zone 51
4.7 Choosing a Root Password 52
4.8 Choosing a User Password 52

Chapter 5. Files, Permissions, and Owners 53
5.1 How to Get File Information 53
5.2 Protection Bits 53
5.3 Owners and Groups 55
Changing File Ownership 55
Changing Group Ownership 55
5.4 Special Permissions 56
5.5 Hard Links, Symbolic Links: the UNIX File System 58
File System Tree 58
File Storage 59
Directory Files 60
Hard Links 61
Symbolic Links 63

Chapter 6. File Security, Repair, and Archiving 65
6.1 fsck Command 65
6.2 ufsdump Command 66
6.3 ufsrestore Command 68
6.4 tar Backups 69
6.5 Backups Using cron 69
6.6 Data Compression 69
What Files Can Be Compressed? 70
Procedures for Compressed Archiving 70

Chapter 7. UNIX Commands 73
7.1 Command Syntax 73
Command Arguments 73
Online Command Reference 74
Command Grouping 74
Background Execution 74
Input and Output Redirection 74
7.2 File Names 75
File Name Suffix 75
Dot Files 75
File System Hierarchy 75
Wildcard Characters 76
Keyboard Shortcuts for Entering File Names 76
7.3 Important UNIX Commands 77
Directory Commands 77
File Handling Commands 77
Text Commands 79
Process Management and System Administration Commands 79
7.4 Interactive Work with the C Shell 80
7.5 Shortcuts within Open Look and CDE 81

Chapter 8. Software Operating Levels 83
8.1 Solaris Run Levels 83
Run Level 0—Monitor Mode 83
Run Level 1—Single-User Mode 85
Run Level 2—UNIX Without Networking 85
Run Level 3—Networking Included 86
init and shutdown Commands 86
How Does the UNIX Bootup and Shutdown Work? 87
8.2 Login Shell 89
8.3 Windowless Multiuser Mode 90
8.4 Graphical User Interface 90
Common Desktop Environment 90
Shell Tool, Command Tool, Dtterm, and Xterm 91
VNMR/VnmrJ 92
8.5 Shutting Down the System 93
To Shut Down the Entire System 93
To Shut Down Acquisition Computer or Acquisition Cabinet 93

Chapter 9. Shell Scripts 95
9.1 C Shell Scripts 95
C Shell Selection, Comments 95
Set-Variables 96
Environment Variables 96
Arithmetics with Variables 98
Flow Control 99
Arguments 102
Interactive Input 102
Here Documents 103

9.2 Bourne Shell Scripts 103
Bourne Selection, Comments 103
Variables 104
Environment Variables 104
Arithmetic with Variables 104
Flow Control 104
Conditional Execution 106
Looping 107
`case` Construction 107
Arguments 108
Interactive Input 109
Here Documents 109
Procedures 109
9.3 How to Write Shell Scripts 110

Chapter 10. UNIX System Customization 111
10.1 General UNIX Customization 111
.login File 111
.cshrc File 112
.exrc File 113
.profile File 113
.mailrc File 113
.indent.pro File 113
10.2 X Window System Configuration Files 113
.xinitrc File 113
.Xdefaults File 113
.xlogin File 114
10.3 CDE Customization Files 114
10.4 VNMR and Other Application Configuration Files 114

Chapter 11. Disk Organization 115
11.1 Disk Partitioning 115
Naming Disk Slices 115
A Typical Solaris System Disk Layout 117
How is a Disk Organized Internally? 117
ufs File System 118
11.2 Solaris Directory Structure 119
Files in the Root Slice 120
/tmp Slice 122
/var Slice 122
/opt Slice 123
/usr File System 123
Home Directory Slice, /export/home 124
Files on Other Disks or Disk Slices 126
11.3 Multiuser Setup for VnmrJ 126

Chapter 12. Networking via Ethernet 129
12.1 Ethernet Configurations 129
Standard Ethernet and Thinnet 129
Twisted-Pair Ethernet 132

12.2 Basic Ethernet Network Access 133
Ethernet Hardware Address 133
Numeric IP Address 133
Netmasks, /etc/netmasks 134
/etc/hosts File 135
/etc/hostname.*, /etc/nodename Files 136
12.3 Network Access and Security 136
Global Access Control, /etc/hosts.equiv 136
Individual Access Control, ~/.rhosts 137
12.4 Checking the Network 138
12.5 Disabling the Ethernet 139
12.6 Remote Login, Remote Shells, Remote Copy 140
12.7 Remote Printing and Plotting 140
12.8 Advanced Networking: Connecting to the Internet 141
Defining the Default Gateway (/etc/defaultrouter) 141
Activating DNS (/etc/resolv.conf, /etc/nsswitch.conf) 141
12.9 Mounting File Systems (mount, /etc/dfs/dfstab) 142
Local Mounting 142
Unmounting a Local File System 143
Remote (NFS) Mounting 143
Unmounting a Remote File System 144
NFS Mounting Options, Sharing a File System 144
Limitations 145
NFS Mounting and `root` Access 145
12.10 Automatic Mounting (/etc/vfstab) 146
12.11 Automounter 148
How it Works: An Example 148
Automount Maps (/etc/auto_master) 149
12.12 Using FTP 149
Using FTP Interactively 150
Making FTP More Automatic 151
Calling ftp Within Shell Scripts 152
12.13 NIS and NIS+ 153
12.14 Heterogeneous Networks 153
12.15 Remote Home Directories 154
Using the Automounter 154
Using Standard NFS Mounting 154
12.16 Gateway Machines 155
Spectrometer Hosts Should Not Act as Gateways! 155
12.17 Diskless Systems 156

Chapter 13. Other Networking Options 157

13.1 Hardware Options 157
13.2 Direct Connections 158
Transfers without Software Protocol 158
Using tip on a Direct Line 159
Using uucp on a Direct Line 159
13.3 Using a Modem 159
13.4 Using Mail on a Client Machine 160

13.5 Avoiding Disk Full Problems with Mail 160
Receiving Large Mail Files 160
Sending Large Mail files 160
How to Avoid Problems with Mail 161
13.6 Other Software 162

Chapter 14. Text Editing 163
14.1 Text Editors dtpad and textedit 163
dtpad 163
vi 164
14.2 Text Editor vi 164
Command Mode 164
Insert Mode 166
Last Line Mode 166
Other vi commands 167
Problems with Running vi 167
14.3 Other Editors 168
ed and ex Editors 169
emacs Editor 169

Chapter 15. Using Tapes and Floppy Disks 171
15.1 tar Command 171
Command Syntax 171
tar Files 172
QIC Tape Formats 173
Blocking Factor 173
Writing Files 173
Tape Catalogs 174
Reading Files 174
Verbose Option 175
Making Things Simpler 175
Using tar to Transfer Directories 175
Using Multifile Tapes 176
Using tar Remotely 176
15.2 ufsdump, ufsrestore, and dd Commands 177
Reading Gemini and VXR-4000 Tapes 177
15.3 Using Floppy Disks 178
Formatting Floppy Disks 178
Floppies with the tar File System 179
Floppies with DOS Format 179
Floppies with UNIX File System 180
Transferring Large Files Via Floppy Disks 182
Reading bar Floppies 183

Chapter 16. Output Devices, Terminals, Serial Ports 185
16.1 /vnmr/devicenames File 185
16.2 /vnmr/devicetable File 186
16.3 Printing and Plotting under VnmrJ and VNMR 187
Printing and Plotting Files 187
vnmrprint Shell Script 187

vnmrplot Shell Script 188
16.4 UNIX Aspects of Printing and Plotting 188
How to Use Printers Under Solaris 189
How Does Solaris Printing Work? 189
16.5 Null Modems for Printers and Plotters 189
16.6 Fixing Problems with Printing and Plotting 190
16.7 Connecting Serial Terminals 190
16.8 Using Extra Ports 191
16.9 Using an I/O Port for Acquisition Diagnostics 191

Chapter 17. X Window System Operation 193
17.1 X Window System Environment 193
17.2 Using a Workstation as an X Server 193
Setting Up a System 193
Using the RS/6000 as an X Server 194
17.3 Using a Personal Computer as an X Server 195
Starting Up VNMR Using a Personal Computer 195
Starting X Client from a Telnet Session 196
Controlling xterm Window Appearance 198
Solving Networking Problems 198
17.4 Customizing the X Environment 199

Chapter 18. Periodic Maintenance and Troubleshooting 201
18.1 Daily and Periodical Care of Computer Hardware 201
18.2 Managing Free Space on Disks 201
18.3 Shutting Down and Restarting UNIX 202
Normal UNIX Shut Down 202
Shutting Down UNIX in an Emergency 204
Restarting UNIX 204
18.4 Automatic File System Maintenance 205
18.5 Troubleshooting 206
Acquisition Status Codes 206
Display Hardware Console Status (UNITY*INOVA*) 208
Global Parameter File Corrupted 209
Insufficient Swap Space 209
tmp Partition Fills Up 210
Password Invalid 210
System Hangs 210
Changing the Name of a Solaris Host 211
Ethernet Network Troubleshooting 211
System Failure 213
18.6 Activating a Second Hard Disk After Solaris is Installed 215
To Install a New Disk 215
To Partition and Label the New Disk 217
To Create the New File System on the New Disk 219
18.7 Active SCSI Termination 220

Annotated Bibliography 221

Glossary of UNIX, Sun, and VnmrJ Terms 229

Index 277

List of Figures

Figure 1. Example Disk Layouts for Solaris 9 24
Figure 2. VNMR Icon in CDE Tool Bar 27
Figure 3. Example Disk Layouts for Solaris 8 35
Figure 4. UNIX File System Tree Example 59
Figure 5. UNIX File Storage 60
Figure 6. Structure of Directories and Subfiles 62
Figure 7. New Type of File Structure 63
Figure 8. Symbolic Link 63
Figure 9. Point-to-Point Thinnet Network 131
Figure 10. Performance Meter with Nonexistent Ethernet 132
Figure 11. Local Network with Tape Drive 176
Figure 12. Example of `app-defaults` X Resource Setup File 199

List of Tables

Table 1. Solaris 9 Preinstallation Worksheet 19
Table 2. Alternate Layout for Solaris 9 Disk Slice Sizes for VnmrJ or VNMR 23
Table 3. Solaris 9 Disk Slice Sizes for VnmrJ or VNMR 25
Table 4. Solaris 8 Preinstallation Worksheet 30
Table 5. Solaris 8 Disk Slice Sizes for VnmrJ or VNMR. 34
Table 6. Sun Workstations, Sun Architecture, and Solaris Versions 46
Table 7. Solaris 9 Software Group Space Requirements 46
Table 8. Solaris 8 Software Group Space Requirements 46
Table 9. Solaris 7 Software Group Space Requirements 47
Table 10. Usable Tape Lengths of Tape Cartridges 67
Table 11. Dump Scheme for Frequently Used Disk Partitions 67
Table 12. Examples of File Compression. 70
Table 13. Shell Tools and Their Properties 92
Table 14. Physical Disk Size Compared to Usable Disk Space 119
Table 15. Lock Types 125

Disclaimer

The information in this manual is intended to assist users in topics beyond the normal NMR spectrometer system hardware and software support provided by Varian. All information is provided on an as-is basis, and Varian support and service personnel are unable to answer questions or solve problems related to the information in this manual. For further assistance, consult local experts or textbooks in the field.

Foreword

The material for this manual was collected and written for the Varian user courses for NMR spectrometers using Sun graphics workstations as host computers. As this material has grown, it has become an important source of information for anybody dealing with such instruments: scientists, operators, Varian service personnel, and also applications scientists, including the authors.

Notice that the present version of this manual specifically refers to Solaris 2.5 (and VNMR 5.x, and 6.1, as much as VNMR specifics are involved). It is at least partially inconsistent with earlier Sun operating system and VNMR releases, in particular with SunOS 4.x. Printing has changed again with Solaris 2.6—the current version of the manual only covers Solaris 2.5 (SVR4, `lp`) printing.

This manual covers Sun Workstations and their operating system and does not cover any other computers that VNMR supports (such as the IBM RS/6000 series of graphics workstations, or SGI workstations) and their operating systems.

It must be understood that such material can never be complete nor really up to date. We are in a constant process of improving and completing it. It is hard enough to keep up with the development at VNMR, let alone Sun. Suggestions for improvement are always welcome. Chapters 16 and 20 are not written yet—they will be added in a future version of the manual. The same is true for some sections in the chapters 4, 5, and 19.

Rolf Kyburz (rolf.kyburz@ch.varian.com) &
Gerald Simon (gerald.simon@de.varian.com)
Zug and Darmstadt
February 1998

Acknowledgments

The authors would like to thank his colleagues for reviewing and verifying parts of the manual, Günter Hirtz (Varian, Darmstadt) for sharing with us his in-depth expertise in Solaris printing, and Mr. Mark Vine (Rhône-Poulenc Rorer Ltd., Dagenham, U.K.) for major contributions to the literature list as well as other chapters. Suggestions from colleagues and Varian customers that were communicated in *Magnetic Moments* or in internal publications have been adapted and incorporated into this material.

Conventions in This Manual

The following notational conventions are used throughout VNMR manuals:

- Typewriter-like characters identify VNMR and UNIX commands, parameters, directories, and file names in the text of the manual:

  The `shutdown` command is in the `/etc` directory.

- Typewriter-like characters also show text displayed on the screen, including the text echoed on the screen as you enter commands:

  ```
  Self Test Completed Successfully.
  ```

- User input is shown in usually shown in bold type:

  ```
  # cd /cdrom
  # ls
  cdrom0    solaris_2_5_1_desktop_1_1
  #
  ```

- Input or output that depends on local use is shown in italics:

  ```
  Login: root
  Password: root_password
  ```

- Optional input is shown by angled brackets:

  `seqgen s2pul<.c>` means that `seqgen s2pul.c` and `seqgen s2pul` are functionally the same.

- Lines of text containing command syntax, examples of statements, source code, and similar material are often too long to fit the width of the page. To show that a line of text had to be broken to fit into the manual, the line is cut at a convenient point (such as at a comma near the right edge of the column), a backslash (\) is inserted at the cut, and the line is continued as the next line of text. This notation is familiar to C programmers. Note that the backslash is not part of the line and, except for C source code, should not be typed when entering the line.

- Because pressing the Return key is required at the end of almost every command or line of text you type on the keyboard, use of the Return key is usually mentioned only in cases where it is *not* used. This convention avoids repeating the instruction “press the Return key” throughout most of this manual.

Chapter 1. Solaris 9 Installation

Sections in this chapter:

- 1.1 “Solaris 9 Compatibility,” this page
- 1.2 “Solaris 9 Quick Installation,” page 18
- 1.3 “Booting from the Solaris 9 CDs or DVDs,” page 18
- 1.4 “Solaris 9 Detailed Installation,” page 20
- 1.5 “Additional Solaris 9 Software,” page 26
- 1.6 “Sun Management Center 3.0 Platform 4 Update,” page 27
- 1.7 “Adding the VNMR Icon to the CDE Tool Bar,” page 27

This chapter describes how to install Solaris 9 software on a Sun computer to be used as an NMR spectrometer host.

Be sure the Solaris 9 version you plan to install is compatible with your Sun computer. For example, the May 2002 version of Solaris 9 is available for the Ultra 10, Ultra 60, Blade 100, and Blade 2000 workstations.

CAUTION: **If Solaris 9 is preinstalled on your computer, do not use the preinstalled software. In order for VnmrJ or VNMR 6.1C to properly operate, you MUST install Solaris 9.**

Start with *Solaris Quick Installation* in the next section to start the Interactive Solaris Installation program. Then, if you have any questions about a particular Solaris installation window, refer to the corresponding step in the section *Solaris Detailed Installation*, which starts on page 20.

CAUTION: **Do not use these instructions to load any other versions of Solaris except version 9 or load any other versions of VnmrJ or VNMR, unless so instructed in writing by Varian (not by Sun Microsystems, Inc.). Varian's software engineers and applications scientists have tested the compatibility of the Solaris, VnmrJ, and VNMR versions used in this manual and found it to be safe and fully operational for the supported Varian NMR spectrometer systems.**

1.1 Solaris 9 Compatibility

Solaris 9 is compatible with VnmrJ and VNMR 6.1C. For systems with VNMR 6.1C, be sure to follow the instructions in section 1.7 “Adding the VNMR Icon to the CDE Tool Bar,” page 27. Solaris 9 is supported only on UNITY*INOVA* and *MERCURY*-series systems.

Solaris 9 is a 64-bit operating system that generally retains compatibility and functionality for the existing 32-bit environment. Systems that can only run the 32-bit mode, including

sun4c, sun4d, and sun4m platforms, do not require the flash PROM update. For these systems, do not select 64-bit mode.

Be sure the Solaris 9 version you plan to install is compatible with your Sun computer. For example, the May 2002 version of Solaris 9 is available for the Ultra 10, Ultra 60, Blade 100, and Blade 2000 workstations.

Ultra 1 and Ultra 2 computers require a flash PROM update before they can run the 64-bit mode of the Solaris 9 operating environment. The flash PROM update is described in the Sun manual *Solaris 9 Sun Hardware Platform Guide*.

1.2 Solaris 9 Quick Installation

1. The Solaris installation program asks several questions about your network and system setup. Having this information before beginning the installation can save time later on.

   Table 1 lists the topics of the required information. Fill in the table before starting the installation. Consult a system/network administrator if necessary.

   If you need more information on a particular topic, refer to the "Collecting System and Network Information for Solaris," page 49.

2. Power up the computer and peripherals if not already done.

3. Get to the `ok` prompt:

   - On a running system, log in as `root` and enter:

     ```
     # init 0
     ```

   - On a new system, press **Stop-A** (L1-A on some keyboards) to stop the default boot routine.

   Wait for the `ok` prompt. If a `>` prompt appears, enter **`n`**.

4. Choose the installation media. You have two choices for installing Solaris 9:

   - The Solaris 9 Operating Environment DVD (if your system has a DVD-ROM drive). Insert the DVD and follow the instructions.
   - CD-ROMs 1 and 2

1.3 Booting from the Solaris 9 CDs or DVDs

You can install Solaris 9 from two CD-ROMs or a DVD-ROM. This section describes the Initial Installation procedure for installing Solaris 9 using the following CD-ROMs:

- "Solaris 9 Software, 1 of 2"
- "Solaris 9 Software, 2 of 2"

Boot Failure from DVD

If you install Solaris 9 from the DVD-ROM, be aware that if your system has a Toshiba SD-M1401 DVD-ROM with firmware revision 1007, it will be unable to boot from the Solaris 9 DVD. You will see the following error message:

```
Boot from DVD media fails on systems with Toshiba SD-M 1401
DVD-ROM
```

Table 1. Solaris 9 Preinstallation Worksheet

Category	*Comments*	*Your Configuration*
Host Name	Choose a host name; for example, `mercury300`, `inova750` **DO NOT USE:** `inova`, `inovaauto`, `gemcon`, or `wormhole`	
Primary Network Interface	For systems with a second Ethernet board, choose the one that will interface with the building or LAN network (not the NMR console). Use: `le0` for 10baseT Ethernet boards; `hme0` for Ultras or 10/100baseT Ethernet boards, and `eri0` for Blades.	
IP Address	Your network IP number for networked systems, or use 10.0.0.1 for non-networked spectrometers with no second Ethernet board	
Default Route	Choose Find One or Specify One If you choose Find One, the system finds the default route information for you.	
Name Service	Choose NIS+ or NIS if the system is known to the name server, Choose Other if the site is using DNS, DCE, or similar. Choose None for no name service.	
Domain Name	Your network domain name; for example: `our.domain`	
Name Server	Choose Find One or Specify One If you choose Find One, the system finds name server information for you. If you choose Specify One, you will be asked for the following: Name Server Hostname – for example, `ourserver` Name Server IP Address – for example, `195.5.2.25` Subnet Mask – for example: `255.255.255.0`	
Proxy Server	Give name, e.g., `proxy.`*`domain`*`.com`	
Client Services Allocations	Used for allocating file systems for disk-less clients. VnmrJ or VNMR does not require client services to be set up.	
Disk Layout	Disk layout sizes vary according to disk size. The installation procedure suggests proper sizes. The recommended layout would include `/` and `swap`, with `/export/home` as either a directory or slice. Entire Distribution plus OEM package requires up to 2.2 GB. VnmrJ or VNMR requires from 175 to 300 MB.	

To solve this problem, apply patch 111649-03, or a later version, to update the Toshiba SD-M1401 DVD-ROM drive's firmware. The patch is included on the Solaris 9 Supplement CD-ROM. See the README file for detailed instructions on loading the patch.

Bootup from the CD-ROM

1. Insert the CD-ROM labeled "Solaris 9 Software, 1 of 2" or the DVD and boot the system:

   ```
   OK> boot cdrom
   ```

2. Use the information written in Table 1 to fill in the Solaris installation windows.

   If you have questions about a particular Solaris installation window, refer to the step in the section "Solaris 9 Detailed Installation," next that corresponds to that window.

1.4 Solaris 9 Detailed Installation

The following procedures correspond to the Solaris 9 installation windows as they appear.

- "Language and Locale," page 20
- "Identify This System," page 20
- "Time Zone Information," page 21
- "Configuring Solaris for Installation," page 22
- "Interactive Installation," page 22

Language and Locale

1. **Select Language and Locale** – Select **English** from the Languages list. Choose one of the following selections from the Locales List:
   - English (C – 7-bit ASCII)
   - USA–English (8859-1)
   - USA–English (ISO-8859-15)

   There is a pause.
2. **The Solaris Installation Program** – Read the information and click **Continue**.

Identify This System

This group of windows is used to identify the system. When a Confirm Information window appears, check the listed information with what you wrote in Table 1.

1. **Identify This System** – Select **Continue** to begin identifying the system.
2. **Network Connectivity** – Choose **Yes** or **No** according to the following:
   - Choose **Yes** for UNITY*INOVA* or *MERCURY*-series systems.
3. **DHCP** – Specify whether your system should use or should not use DHCP for network interface configuration.
4. **Primary Network Interface** – Select the primary network interface from the list. Choose an interface (in order of preference): `eri0`, `hme0`, `le0`.
5. **Host Name** – Enter the host name of the computer. Refer to Table 1.

   The host name identifies the system on the network. We recommend choosing a host name that is all lower case. **DO NOT USE** `inova`, `inovaauto`, `gemcon`, or `wormhole` as a host name, these are reserved for the console. Use host names like `mercury300` or `inova750`.

   The host name must be unique within the domain in which it resides, and the name must be at least two characters, containing letters, digits, and minus signs (–).
6. **IP Address** – Enter the Internet Protocol address for your system. Refer to Table 1.

   The IP address must follow the site's address conventions. IP addresses contain four sets of numbers separated by periods (e.g., 129.200.9.1).
   - For systems with two Ethernet interfaces (e.g., one built-in and one add-on board), use an IP address provided by your network administrator.
   - For systems with one Ethernet interface *that only will be connected to the NMR console*, use 10.0.0.1 for the IP address.

- For systems with one Ethernet interface *that will be connected to a network*, use an IP address provided by your network administrator.

7. **Subnets** – Specify whether your system is or is not part of a subnet.

   This screen appears only when a second Ethernet board is installed and enabled.

8. **Netmask** – for example: `255.255.255.0`
9. **IPv6** – Specify whether the next generation Internet Protocol should or should not be enabled on your system. There is a pause.
10. **Default Route** – Specify how to set the default route. Choose **Find one** or **Specify one**.
11. **Confirm Information** – Verify that the information in the window is correct.
12. **Configure Security Policy** – Select **Yes** or **No** to configure Kerberos Security. At the time of this manual's publication, Varian has not verified the compatibility of the Kerberos Security software with VnmrJ or VNMR.
13. **Confirm Information** – Verify that the information in the Kerberos window is correct.

The following windows are used to identify the name service and network used by your computer. When a Confirm Information window appears, check the listed information with what you wrote in Table 1.

1. **Name Service** – Select a name service. Refer to Table 1.
   - Select **NIS+** or **NIS** if the system is known to a name service.
   - Select **DNS** if the site is using DNS or a similar name service.
   - Select **LDAP** if the site is using LDAP or a similar name service.
   - Select None if the site is not using a name service.
2. **Domain Name** – Enter the domain name in which the system resides. Refer to Table 1.
3. **Name Server** – Choose **Find one** or **Specify one**. Refer to Table 1.
   - If you choose Find one, the system finds name server information for you.
   - If you choose Specify one, you will be asked for the following:

     Name Server Hostname – for example, `ourserver`

     Name Server IP Address – for example, `195.5.2.25`
4. **Confirm Information** – Compare the information listed in this window with the information written in Table 1.

Time Zone Information

This group of windows is used to set the default time zone of your system. When a Confirm Information window appears, check the listed information.

- **Time Zone** – Enter the time zone information, geographic region, date, and time for your location.

When a Confirm Information window appears, check the listed information.

Configuring Solaris for Installation

After the system is identified, the interactive installation program displays the next group of screens, which ask for the following information:

- Upgrade or initial installation
- System type
- Software group
- Disks
- Preserve data
- Automatically layout file system
- Mount remote file systems

Interactive Installation

1. **Solaris Interactive Installation** – The **Initial** option overwrites the system disks when the new Solaris software is installed. You can accept the defaults or customize how Solaris is installed:
   - Select the type of Solaris to install.
   - Select disks to hold software you've selected.
   - Specify how file systems are laid out on disks.

   Select the way to install the Solaris software:
   - **Standard** installs your system from a Standard Solaris Distribution.
   - **Flash** installs your system from one or more Flash archives.
2. **Select Geographic Regions** – Select the geographic regions for which support should be installed, then click **Continue**.
3. **Select 64 Bit** – Select 64-bit if you want to include the Solaris 64-bit packages on this system.
4. **Select Software** – Choose **Entire Distribution plus OEM**.

   We do not recommend customizing the software packages.
5. **Select Disks** – Select the disks on which to install Solaris:
   a. Select disks from the Available Disks side.
   b. Click **Add** to move the disks to the Selected Disks side.
6. **Preserve Data?** – If you want to preserve a file system, keep it from being overwritten, click the **Preserve** button.

   In the Preserve screen, click the box next to the disk slices you would like to preserve (e.g., `/data`).
7. **Automatically Layout file Systems?** – Click **Auto Layout**.

   We suggest selecting the following file systems to automatically layout:
   - `/`
   - `swap`

   With only `root` (`/`) and `swap` selected, the other files systems in the list are collapsed into `root`. While this makes the space requirements for the `root` partition (or slice) bigger, you will not be constrained by fixed-sized slices; you will

have more of the disk to work with. Also, this configuration can save up to 300 MB of disk space.

For an alternate file system layout see Table 2.

Table 2. Alternate Layout for Solaris 9 Disk Slice Sizes for VnmrJ or VNMR

Slice	*Mount Point*	*Size*	*Description*
0	/	**1019 MB**, at least	Contains /, /opt, /usr, /var Add 200-400 MB if options are loaded, heavy printing, use is expected as with an SMS, log files can be large.
2	overlap	Entire disk	Spans the entire disk.
3	swap	**513 MB**	Space for virtual memory.
7	/export/home or /data	Remainder	/export/home contains VNMR and VnmrJ user accounts. Give it as much disk space as possible.

If the disk has enough space, the /export/home directory is created and placed in its own partition. If the disk does not have enough space, you must create /export/home later.

Alternatively, you can choose a different file system layout. Be sure, however, to always have at least the root (/) and swap file systems. Items that are selected are set up in separate partitions. Items that are not selected are collapsed into the parent file system (they become a directory); this decreases the number of file systems but increases the size of the parent file systems. For example, we usually select /usr but not /usr/openwin. This way, /usr/openwin becomes contained by /usr.

8. **File System and Disk Layout** – Look at the File System column. At least the root (/) and swap file systems must be present. The /export/home file system might also be present; if not, you can create a /export/home directory (using mkdir) later. Click **Continue** and go to the next step.

   Figure 1 shows some example disk layouts as they appear in the Customize Disks window, which is opened by clicking **Customize** in File System and Disk Layout.

   If you chose to create more file systems than just root (/) and swap, compare the sizes listed under Size with the values listed in Table 3.

   - If the sizes in the window are the same or bigger than Table 3, click **Continue**. Note that if a /export/home file system does not exist, you must create a directory called /export/home for VnmrJ or VNMR to use.
   - If the sizes in the window are smaller than Table 3, click **Customize**.

     Customize Disks – Alter the size fields on the listed disks to match the sizes in Table 3. Refer to Figure 1 for an example disk layout for a system with an external and internal hard disk.

     The first external disk is identified by **Disk: c0t0d0** and the first internal disk is identified by **Disk: c0t1d0**. Typically, the system writes as many file systems as possible on the internal disk and places the rest on the external disk.

     Your goal should be to match the sizes shown in Table 3 while trying to give as much space as possible to /export/home.

     On a two-disk system, you might see /export/home0 on the external disk and /export/home on the internal disk. If /export/home0 is bigger than

18 GB internal (c0t1d0) with minimum slices

	Disk: c0t1d0	17269 MB
0	/	1819
1	swap	512
2	overlap	17269
3		
4	/export/home	15938
5		
6		
7		

Capacity: 17269 MB
Allocated 17269 MB
Free: 0 MB

Boot Device: c0t3d0s0

With this layout, the directory `/export/home` must be created with the `mkdir` command before VnmrJ or VNMR can be installed.

15.6 GB external (c0t0d0) and 12 GB internal (c0t1d0) with minimum slices

	Disk: c0t0d0	13040 MB
0		
1		
2	overlap	13040
3		
4		
5		
6		
7	/export/home	13040 MB

Capacity: 13040 MB
Allocated 13040 MB
Free: 0 MB

Boot Device: c0t3d0s0

	Disk: c0t1d0	10020 MB
0	/	1819
1	swap	512
2	overlap	10020
3		
4		
5		
6	data	7689
7		

Capacity: 10020 MB
Allocated 10020 MB
Free: 0 MB

With this layout, `/export/home` is automatically created in a slice on the external hard disk (c0t0d0)

Figure 1. Example Disk Layouts for Solaris 9

Table 3. Solaris 9 Disk Slice Sizes for VnmrJ or VNMR

Slice	*Mount Point*	*Suggested Value*	*Description*
0	/	**137 MB**, at least	Contains directories and files essential for system operation; e.g., kernel, device drivers, boot programs.
1	/var	**62 MB**, at least	Contains systems files that are likely to change over the life of the system, e.g., compilations, mail files, uucp files, print spool files. Print files and end log files can get large. We recommend 200 MB or more.
2	overlap		Spans the entire disk.
3	swap	**2 x RAM** or at least **512 MB**, whichever is more.	Space for virtual memory. Must be at least twice the amount of installed RAM or 100 MB, whichever is more. For example, for 32 MB of RAM use at least 100 MB for swap, or for 64 MB of RAM use 128 MB of swap (which is twice the RAM). Note that swap space can be added after Solaris is installed without repartitioning the disk.
4			
5	/opt	**25 MB**, at least	Contains mount points for third-party, unbundled software, and patches. If you plan to install other software, make this larger accordingly.
6	/usr	**1546 MB**	Contains many of the standard UNIX programs, including OpenWindows and CDE files and programs.
7	/export/home or /data	Remainder	/export/home contains VNMR and VnmrJ user accounts. Give it as much disk space as possible.

/export/home, change /export/home on the internal disk to /data, and then change /export/home0 on the external disk to /export/home. That way, the entire external disk is available for /export/home, where VnmrJ or VNMR is installed (see Figure 1).

9. **Mount Remote File Systems?** – Click the **Remote Mounts** button to set up mounts to remote file systems.

   This window enables you to specify and verify remote file systems to mount from a server. While you can always add remote mounts later, adding them now might be more convenient.

10. **Profile** – Check the information in the window. Click **Begin Installation** if the information is acceptable. Click **Change** if you need to change anything.

11. **Reboot After Installation?** – Select **Auto Reboot**. The system automatically reboots after Solaris is installed.

    The system automatically reboots after Solaris is installed. Then, you are asked for a root password. For more information about creating a root password, refer to "Choosing a User Password," page 52.

12. After the system has rebooted, you are prompted to activate the power saving shutdown (Autoshutdown). Select **No**.

    A prompt for the second CD appears. Insert the “Solaris 9 Software, 2 of 2” CD and follow the instructions.

At this stage if you are ready to install VnmrJ or VNMR. Refer to the *Software Installation* manual that shipped with your spectrometer.

1.5 Additional Solaris 9 Software

After you have installed Solaris 9, there are five additional CDs that you can install.

- **Solaris 9 Languages** contains messages files and other software in languages other than English. The appropriate contents are installed from this CD if you configure your system to a nonEnglish locale. If you used the Solaris 9 Installation CD, any software needed from the Solaris 9 Languages CD was automatically installed.
- Two CDs labeled **Solaris 9 Documentation** contain documentation sets in PDF and HTML formats for different language groupings. The sets provide manuals for users, administrators, and developers. You can also read the manuals directly from the CD. If you used the Solaris 9 Installation CD, any software needed from the Solaris Documentation CDs was automatically installed.
- **Solaris 9 Software Supplement** contains additional software for use on Sun hardware products. This CD might also contain patches needed by some systems, such as a firmware patch that enables some older DVD drives to boot this Solaris release from DVD media. For most software on this CD, read the chapter “Installing Software from the Solaris 9 Software Supplement CD” in the *Solaris 9 Hardware Platform Guide*.
- **Solaris Software Companion** contains a collection of Linux applications and other Free and Open Source software for the Solaris 9 operating environment. For more information, visit the site *http://www.sun.com/software/solaris/freeware*. The contents of this CD are not included in the Solaris 9 Operating Environment DVD.

1.6 Sun Management Center 3.0 Platform 4 Update

Also available are three CDs for installing Sun Management Center 3.0, which is designed to simplify the task of monitoring and managing Sun components by providing a single point of management.

1.7 Adding the VNMR Icon to the CDE Tool Bar

After you have installed VNMR 6.1C and if your system is running Solaris 9, you must edit the `makeuser` script in the `/vnmr/bin` directory to get the VNMR icon to appear in the CDE tool bar, as shown in Figure 2.

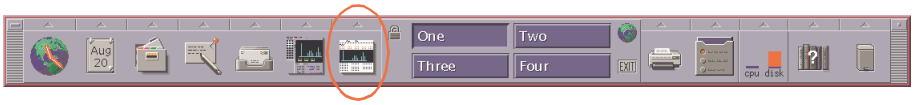

Figure 2. VNMR Icon in CDE Tool Bar

1. Open a Solaris terminal window or host window.
2. Change directories to `/vnmr/bin`:
   `vnmr1> cd /vnmr/bin`
3. To edit the `makeuser` script, enter **`vi makeuser`**
4. Scroll through the `makeuser` script and look for the following line:
   `5.7 | 5.8 )`
5. Change the line so that it reads
   `5.7 | 5.8 | 5.9 )`
6. Save your change by pressing the **Esc** key and typing
   `:wq!`
7. Run the `makeuser` command as described in the *Software Installation* manual.

Chapter 2. Solaris 8 Installation

Sections in this chapter:

- 2.1 “Solaris 8 Compatibility,” this page
- 2.2 “Solaris 8 Quick Installation,” page 30
- 2.3 “Solaris 8 Detailed Installation,” page 31

This chapter describes how to install Solaris 8 software on a Sun computer to be used as an NMR spectrometer host.

CAUTION: **If Solaris 8 is preinstalled on your computer, do not use the preinstalled software. In order for VnmrJ or VNMR to properly operate, you MUST manually load Solaris 8.**

Start with *Solaris Quick Installation* in the next section to start the Interactive Solaris Installation program. Then, if you have any questions about a particular Solaris installation window, refer to the corresponding step in the section *Solaris 8 Detailed Installation*, which starts on page 31.

CAUTION: **Do not use these instructions to load any other versions of Solaris except version 8 or load any other versions of VnmrJ and VNMR, unless so instructed in writing by Varian (not by Sun Microsystems, Inc.). Varian's software engineers and applications scientists have tested the compatibility of the Solaris, VnmrJ, or VNMR versions used in this manual and found it to be safe and fully operational for the supported Varian NMR spectrometer systems.**

2.1 Solaris 8 Compatibility

Solaris 8 is compatible with VnmrJ and VNMR 6.1C (with appropriate patches). Solaris 8 is supported on UNITY*INOVA*, *MERCURY*-series, UNITY*plus*, UNITY, VXR-S, and *GEMINI 2000* systems.

Solaris 8 is a 64-bit operating system that retains compatibility and functionality for the existing 32-bit environment.

Ultra 1 and Ultra 2 computers require the flash PROM update before they can run the 64-bit mode of the Solaris 8 operating environment. The flash PROM update is described in the Sun manual *Solaris 8 Sun Hardware Platform Guide*.

Systems that can only run the 32-bit mode, including sun4c, sun4d, and sun4m platforms, do not require the flash PROM update. For these systems, do not select 64-bit mode.

2.2 Solaris 8 Quick Installation

1. The Solaris installation program asks several questions about your network and system setup. Having this information before beginning the installation can save time later on.

   Table 4 lists the topics of the required information. Fill in the table before starting the installation. Consult a system/network administrator if necessary.

   If you need more information on a particular topic, refer to "Collecting System and Network Information for Solaris," page 49.

Table 4. Solaris 8 Preinstallation Worksheet

Category	*Comments*	*Your Configuration*
Host Name	Choose a host name; for example, `inova750`. Do not use: `inova`, `inovaauto`, `gemcon`, or `wormhole`.	
Primary Network Interface	For systems with a second Ethernet board, choose the one that will interface with the building or LAN network (not the NMR console). Use: `le0` for 10baseT Ethernet boards; `hme0` for Ultras or 10/100baseT Ethernet boards, and `eri0` for Blades.	
IP Address	Your network IP number for networked systems, or use 10.0.0.1 for non-networked spectrometers with no second Ethernet board	
Name Service	Choose NIS+ or NIS if the system is known to the name server, Choose Other if the site is using DNS, DCE, or similar. Choose None for no name service.	
Domain Name	Your network domain name; for example: `our.domain`	
Name Server	Choose Find One or Specify One If you choose Find One, the system finds name server information for you. If you choose Specify One, you will be asked for the following: Name Server Hostname – for example, `ourserver` Name Server IP Address – for example, `195.5.2.25` Subnet Mask – for example: `255.255.255.0`	
Proxy Server (Solaris 8, 1/01 or later)	Give name, e.g., `proxy.`*`domain`*`.com`	
Client Services Allocations	Used for allocating file systems for disk-less clients. VnmrJ or VNMR does not require client services to be set up.	
Disk Layout	Disk layout sizes vary according to disk size. The installation procedure suggests proper sizes. The recommended layout would include `/` and `swap`, with `/export/home` as either a directory or slice.	

2. Power up the computer and peripherals if not already done.
3. Get to the `ok` prompt:
   - On a running system, log in as `root` and enter:
     `# init 0`
   - On a new system, press **Stop-A** (L1-A on some keyboards) to stop the default boot routine.

   Wait for the `ok` prompt. If the `>` prompt appears, enter **`n`**.

4. Installing Solaris 8 involves loading both CD-ROMs 1 and 2. **Do not use the Solaris 8 Installation disk! Insert the CD-ROM labeled "Solaris 8 Software, 1 of 2"** and boot up from the CD-ROM:

   ```
   OK> boot cdrom
   ```

5. Use the information written in Table 4 to fill in the Solaris installation windows.

2.3 Solaris 8 Detailed Installation

The following procedures correspond to the Solaris 8 installation windows as they appear.

- "Language and Locale," page 31
- "Identify This System," page 31
- "Time Zone Information," page 32
- "Configuring Solaris for Installation," page 33

Language and Locale

1. **Select Language and Locale** – Select **English** from the Languages list. Choose one of the following selections from the Locales List:
   - English (C – 7-bit ASCII)
   - USA–English (8859-1)
   - USA–English (ISO-8859-15)

   There is a pause.
2. **The Solaris Installation Program** – Read the information and click **Continue**.

Identify This System

This group of windows is used to identify the system. When a Confirm Information window appears, check the listed information with what you wrote in Table 4.

1. **Identify This System** – Click **Continue** to begin identifying the system.
2. **Network Connectivity** – Choose **Yes** or **No** according to the following:

   Choose **Yes**.

   This screen appears only when a second Ethernet board is installed and enabled.
3. **DHCP** – Specify whether your system should use or should not use DHCP for network interface configuration.
4. **Primary Network Interface** – Select the primary network interface from the list. Choose an interface (in order of preference): `eri0`, `hme0`, or `le0`.
5. **Host Name** – Enter the host name of the computer. Refer to Table 4.

   The host name identifies the system on the network. We recommend choosing a host name that is all lower case. Do not use `inova`, `inovaauto`, `gemcon`, or `wormhole` as a host name, these are reserved for the console. Use host names like `inova750`.

   The host name must be unique within the domain in which it resides, and the name must be at least two characters, containing letters, digits, and minus signs (–).
6. **IP Address** – Enter the Internet Protocol address for your system. Refer to Table 4.

The IP address must follow the site's address conventions. IP addresses contain four sets of numbers separated by periods (e.g., 129.200.9.1).

- For systems with two Ethernet interfaces (e.g., one built-in and one add-on board), use an IP address provided by your network administrator.
- For systems with one Ethernet interface *that only will be connected to the NMR console*, use 10.0.0.1 for the IP address.
- For systems with one Ethernet interface *that will be connected to a network*, use an IP address provided by your network administrator.

7. **IPv6** – Specify whether the next generation Internet Protocol should or should not be enabled on your system.
8. **Confirm Information** – Verify that the information in the window is correct.
9. **Configure Security Policy** – Select **Yes** or **No** to configure Kerberos Security. At the time of this manual's publication, Varian has not verified the compatibility of the Kerberos Security software with VnmrJ or VNMR.
10. **Confirm Information** – Verify that the information in the Kerberos window is correct.

The following windows are used to identify the name service and network used by your computer. When a Confirm Information window appears, check the listed information with what you wrote in Table 4.

1. **Name Service** – Select a name service. Refer to Table 4.
   - Select **NIS+** or **NIS** if the system is known to a name service.
   - Select **DNS** if the site is using DNS or a similar name service.
   - Select **LDAP** if the site is using LDAP or a similar name service.
   - Select None if the site is not using a name service.
2. **Domain Name** – Enter the domain name in which the system resides. Refer to Table 4.
3. **Name Server** – Choose **Find one** or **Specify one**. Refer to Table 4.
   - If you choose Find one, the system finds name server information for you.
   - If you choose Specify one, you will be asked for the following:

     Name Server Hostname – for example, `ourserver`

     Name Server IP Address – for example, `195.5.2.25`
4. **Subnets** – Specify whether your system is or is not part of a subnet.
5. **Netmask** – for example: `255.255.255.0`
6. **Confirm Information** – Compare the information listed in this window with the information written in Table 4. There is a long pause.

Time Zone Information

This group of windows is used to set the default time zone of your system. When a Confirm Information window appears, check the listed information.

- **Time Zone** – Enter the time zone information, geographic region, date, and time for your location.

When a Confirm Information window appears, check the listed information. There is a pause.

Configuring Solaris for Installation

After the system is identified, the interactive installation program displays the next group of screens, which ask for the following information:

- Upgrade or initial installation
- System type
- Software group
- Disks
- Preserve data
- Automatically lay out file system
- Mount remote file systems

1. **Solaris Interactive Installation** – Choose **Initial**.

   Read the information on the next screen and click **Continue**.

2. **Select Geographic Regions** – Select the geographic regions for which support should be installed, then click **Continue**.

3. **Select Software** – Choose **Entire Distribution plus OEM**.

   We do not recommend customizing the software packages.

4. **Select Disks** – Select the disks on which to install Solaris:

   a. Select disks from the Available Disks side.

   b. Click **Add** to move the disks to the Selected Disks side.

5. **Preserve Data** – If you want to preserve a file system, keep it from being overwritten, click the **Preserve** button.

   In the Preserve screen, click the box next to the disk slices you would like to preserve (e.g., `/data`).

6. **Automatically Layout file Systems** – Click **Auto Layout**.

   We suggest selecting the following file systems to automatically layout:

   - `/`
   - `swap`

   With only `root` (`/`) and `swap` selected, the other files systems in the list are collapsed into `root`. While this makes the space requirements for the `root` slice bigger, you will not be constrained by fixed-sized slices; you will have more of the disk to work with. Also, this configuration can save up to 300 MB of disk space.

   If the disk has enough space, `/export/home` is created and placed in its own slice. If not, you must create a `/export/home` directory later.

   Alternatively, you can choose a different file system layout. Be sure, however, to always have at least root (`/`) and `swap` file systems. Items that are selected are set up in separate partitions (slices). Items that are not selected are collapsed into the parent file system (they become a directory); this decreases the number of file systems but increases the size of the parent file systems. For example, we usually

select /usr but not /usr/openwin. This way, /usr/openwin becomes contained by /usr.

7. **File System and Disk Layout** – Look at the File System column. At least the root(/) and swap file systems must be present. The /export/home file system might also be present; if not, you can create a /export/home director (using mkdir) later. Click **Continue** and go to the next step.

   Figure 3 shows some example disk layouts as they appear in the Customize Disks window, which is opened by clicking **Customize** in the File System and Disk Layout window.

   If you chose to create more file systems than just root(/) and swap, compare the sizes listed under Size with the values listed in Table 5.

Table 5. Solaris 8 Disk Slice Sizes for VnmrJ or VNMR.

Slice	*Mount Point*	*Suggested Value*	*Description*
0	/	**400 MB**, at least	Contains directories and files essential for system operation; e.g., kernel, device drivers, boot programs.
1	/var	**60 MB**, at least	Contains systems files that are likely to change over the life of the system, e.g., compilations, mail files, uucp files, print spool files.
2	overlap		Spans the entire disk.
3	swap	**2 x RAM** or at least **100 MB**, whichever is more.	Space for virtual memory. Must be at least twice the amount of installed RAM or 100 MB, whichever is more. For example, for 32 MB of RAM use at least 100 MB for swap, or for 64 MB of RAM use 128 MB of swap (which is twice the RAM). Note that swap space can be added after Solaris is installed without repartitioning the disk.
4			
5	/opt	**25 MB**, at least	Contains mount points for third-party, unbundled software, and patches. If you plan to install other software, make this larger accordingly.
6	/usr	**827 MB**	Contains many of the standard UNIX programs, including OpenWindows and CDE files and programs.
7	/export/home or /data	Remainder	/export/home contains VNMR and VnmrJ user accounts. Give it as much disk space as possible.

- If the sizes in the window are the same or bigger than Table 5, click **Continue**. Note that if a /export/home file system does not exist, you must create a directory called /export/home for VnmrJ or VNMR to use.
- If the sizes in the window are smaller than Table 5, click **Customize**.

  Customize Disks – Alter the size fields on the listed disks to match the sizes in Table 5. Refer to Figure 3 for an example disk layout for a system with an external and internal hard disk.

  The first external disk is identified by **Disk: c0t0d0** and the first internal disk is identified by **Disk: c0t1d0**. Typically, the systems places as many

18 GB internal (c0t1d0) with minimum slices

	Disk: c0t1d0	17269 MB
0	/	1819
1	swap	512
2	overlap	17269
3		
4	/export/home	15938
5		
6		
7		

Capacity: 17269 MB
Allocated 17269 MB
Free: 0 MB

Boot Device: c0t3d0s0

With this layout, the directory `/export/home` must be created with the `mkdir` command before VnmrJ or VNMR can be installed.

15.6 GB external (c0t0d0) and 12 GB internal (c0t1d0) with minimum slices

	Disk: c0t0d0	13040 MB
0		
1		
2	overlap	13040
3		
4		
5		
6		
7	/export/home	13040 MB

Capacity: 13040 MB
Allocated 13040 MB
Free: 0 MB

Boot Device: c0t3d0s0

	Disk: c0t1d0	10020 MB
0	/	1819
1	swap	512
2	overlap	10020
3		
4		
5		
6	data	7689
7		

Capacity: 10020 MB
Allocated 10020 MB
Free: 0 MB

With this layout, `/export/home` is automatically created in a slice on the external hard disk (c0t0d0)

Figure 3. Example Disk Layouts for Solaris 8

files systems as possible on the internal disk and places the rest on the external disk.

Your goal should be to match the sizes shown in Table 5 while trying to give as much space as possible to `/export/home`.

On a two-disk system, you might see `/export/home0` on the external disk and `/export/home` on the internal disk. If `/export/home0` is bigger than `/export/home`, change `/export/home` on the internal disk to `/data`, and then change `/export/home0` on the external disk to `/export/home`. That way, the entire external disk is available for `/export/home`, where VnmrJ or VNMR is installed (see Figure 3).

8. **Mount Remote File Systems** – Click the **Remote Mounts** button to set up mounts to remote file systems.

   This window enables you to specify and verify remote file systems to mount from a server. While you can always add remote mounts later, adding them now might be more convenient.

9. **Profile** – Check the information in the window. Click **Begin Installation** if the information is acceptable. Click **Change** if you need to change anything.

10. Select **Auto Reboot**. The system automatically reboots after Solaris is installed.

    The system automatically reboots after Solaris is installed. Then, you are asked for a `root` password. For more information about creating a `root` password, refer to the "Choosing a Root Password," page 52.

11. After the system has rebooted, a prompt for the second CD appears. Insert the "Solaris 8 Software, 2 of 2" CD and follow the instructions.

Chapter 3. Solaris Patch Installation and Error Messages

Sections in this chapter:

- 3.1 “Patches,” this page
- 3.2 “Error Messages,” page 39

This chapter describes how to install Solaris patches and the VnmrJ software and online help.

Before installing VnmrJ, Solaris must be installed as described in the manual *Solaris Installation for NMR Host Computers* (Pub. No. 01-999223-00).

3.1 Patches

VnmrJ is installed from multiple VnmrJ CD-ROMs and a CD-ROM containing required patches for the operating system:

- *VnmrJ Operating System Patches* — patches for the workstation operating system.
- *VnmrJ*
- *VnmrJ – Manuals and VnmrJHelp*

Before you install VnmrJ, there are Solaris patches that are necessary for proper VnmrJ and VNMR operation. For example, Solaris patch 108940-24 handles core dumps. In Solaris 9 patch 111649-03 (which is not supplied with VnmrJ) might be needed for the firmware for the Toshiba SD-M1401 DVD-ROM drive. We recommend that you load patches now. Patches are on the *VnmrJ Solaris Patches* CD-ROM.

Make sure that your system has the latest Solaris patches. In the patch number, the two digits after the dash are the patch version number. Each patch has a `README` file that provides specific information about the patch.

For the latest information about Solaris patches, visit the SunSolve Online Web site:

```
http://sunsolve.sun.com
```

You can also check the Sun patch FTP server:

```
ftp://sunsolve.sun.com/pub/patches/
```

Or you can visit the Varian NMR User Page at `http:/www.varianinc.com`. Click on **NMR** in the list of *Scientific Instruments*, then click on **User Page** in the *Quick Links* section.

Preparing for Patch Installation

Before you begin installing patches, make sure the following conditions are true:

- You are logged in as single user (`boot -s`).
- VnmrJ is not running.
- No users are logged on to your system (use the `who` command to verify this condition).
- The NMR console is idle.

If you are not sure which version of Solaris you are using, enter the command **showrev**. If the command shows `SunOS 5.8`, your system is running Solaris 8; similarly, `SunOS 5.7` is Solaris 7.

Even within a given Solaris release, there can be newer versions that automatically include and install many patches, including those required by VnmrJ. To check whether a particular patch (for example, 108940-24) is already installed, use the following command (no output means that the patch is not installed):

```
showrev -p | grep 108940-24
```

In addition to individual patches, Sun also offers entire clusters of recommended patches, such as `2.8_Recommended.tar.Z` (typically files of 20 to 30 MB after compression), that permit bringing your Solaris installation to the latest level. Note, however, that the installation of patch clusters requires extra disk space. If don't have at least 36 MB of free space in `/var` (`/var/sadm`), you can handle this problem four ways:

- Install patches one by one using the `patchadd` command.
- Use the `-B` option while invoking `patchadd`. This option directs `patchadd` to save the backout data to the user-specified file system.
- Generate additional disk space by deleting files that are not needed.
- Override the saving of the old files by using the `-d` (do not save) option when running `patchadd`. This method eliminates the possibility of restoring files to previous versions.

Also see the manual page for the `patchadd` command:

```
man patchadd
```

Installation of a Solaris patch cluster takes one to two hours.

Installing Solaris Patches

To install Solaris patches, do the following:

1. Enter **su** to become root.
2. Enter **init 0**.
3. Enter **boot -s** in order to boot up in the single user mode.
4. Enter the following:

   ```
   # ps -ef | grep vold
   ```

   If no results are shown , **?????**:

   ```
   # /etc/init.d/volmgt start
   ```

5. Insert patch CD, and enter the following in a terminal window:

   ```
   # cd /cdrom/cdrom0
   # ./load.patches
   ```

6. You are asked to confirm the installation of patches. Type **y** to install all patches or **n** to skip patch installation.
   If you select **y**, the appropriate patches for Solaris are installed.

7. Eject the CD and reboot the computer:
   ```
   # eject cdrom
   # reboot
   ```

3.2 Error Messages

Device Busy

When you try to eject the CD-ROM and get the message:
`/vol/dev/rdisk/c0t6d0/...: device busy`

make sure that no window runs in which you have changed the directory to `/cdrom/cdrom0`. In such a case, do the following steps:

1. Exit the directory e.g.,
   `user1>` **`cd /`**
2. Click on **Run** windows and close all that use `/cdrom/cdrom0`.
3. Try ejecting the CD-ROM again.

"Can't connect to X11 window server"

```
Exception in thread "main"...Can't connect to X11 window
server...
```

Enter **`xhost +`** in a window owned by the local user (not `root`).

Locator Shows Only Error

If you see the message in the locator:

```
 Error      Error      Error
```

Exit from VnmrJ and, as the VnmrJ administrator (not root), enter the following commands in a shell window:
`vnmr1> /vnmr/bin/dbsetup`

Missing Fonts

When the VnmrJ `load.nmr` program (or any other Java program) starts, warning messages about missing fonts are generated. A typical message might be:

```
Font specified in font.properties not found            \
[-monotype-arial-bold-i-normal--*-%d-*-*-p-*-iso8859-1]
```

These messages are generated because a Solaris package containing these fonts has not been installed.

To install this package, insert the Solaris software CD (the first disk if there are two) and, as `root`, issue the following UNIX commands:

- For Solaris 9:
  ```
  # cd /cdrom/cdrom0/s0/Solaris_9/Product/
  # pkgadd -d . SUNWi1of
  ```
- For Solaris 8:

```
# cd /cdrom/cdrom0/s0/Solaris_8/Product/
# pkgadd -d . SUNWi1of
```

- For Solaris 7:

```
# cd /cdrom/cdrom0/s0/Solaris_2.7/Product
# pkgadd -d . SUNWi1of
```

The specific directory containing the Solaris package depends on the version of Solaris. The `pkgadd` command is the same for all versions of Solaris. The name of the package has the digit one (1) in its name and not the letter *l*.

After the fonts have been installed, eject the CD-ROM.

You do not have to reboot the computer for this package to be activated.

Chapter 4. Solaris Reference Information

Sections in this chapter:

- 4.1 “Mounting a CD-ROM,” this page
- 4.2 “Shutting Down the Sun Computer,” page 44
- 4.3 “Opening a Host or Terminal Window,” page 45
- 4.4 “System Requirements,” page 46
- 4.5 “Disabling and Reenabling a X453A Thinnet Coax Ethernet Board,” page 47
- 4.6 “Collecting System and Network Information for Solaris,” page 49
- 4.7 “Choosing a Root Password,” page 52
- 4.8 “Choosing a User Password,” page 52

This appendix contains reference information for Solaris that might be useful during the software installation.

4.1 Mounting a CD-ROM

VnmrJ and VNMR ships on a CD-ROM; therefore, you will need access to a CD-ROM drive. The CD-ROM drive can be local, attached to the computer on which you are installing the software, or remote, available over the network. In either case, make sure the CD-ROM drive is correctly connected and powered up. Insert the CD in the CD-ROM drive (if necessary use a CD caddy).

Volume Manager

If you are having trouble opening the CD-ROM (`/cdrom/cdrom0`), check to see if Volume Manager is running:

```
ps -ef | grep vold
```

If you see an entry containing `/usr/sbin/vold`, Volume Manager is running.

If Volume Manager is not running, start it by becoming `root` and entering:

```
/etc/init.d/volmgt start
```

Mounting a Local CD-ROM Drive

This procedure describes how to mount the local CD-ROM drive if it is not already mounted. The CD-ROM drive is local if it is attached to the computer on which you will be loading VnmrJ or VNMR.

1. Insert the CD-ROM into the drive.
2. Become root.

3. Check to see that the CD-ROM has been mounted by Solaris as follows:

   ```
   # cd /
   # ls cdrom
   ```

   If the `/cdrom` directory does not exist, enter the following commands:

   ```
   # mkdir /cdrom
   # mount -r -F hsfs /dev/sr0 /cdrom
   ```

   If the `/cdrom` directory does exist but shows no files when you enter `ls`, mount the CD-ROM as follows:

   ```
   # mount -r -F hsfs /dev/sr0 /cdrom
   ```

You are now ready to install the VnmrJ or VNMR software. For more information use the UNIX command `man` on `mount`, `umount`.

Mounting a Remote CD-ROM Drive

If the CD-ROM drive is attached to a remote Sun computer, you must first mount the drive locally.

Below we refer to *remotehost* as the remote host that has the CD-ROM drive attached, and we refer to the *localhost* as the local host where the VnmrJ or VNMR software is to be loaded. The procedure that needs to be followed depends on whether the *remotehost* is running Solaris or SunOS.

If the Remote Host is Running Solaris

If the remote host is running Solaris, use this procedure. If the remote host is running SunOS, skip to the next procedure.

To mount the CD-ROM drive from the *remotehost*, you need the root password and a login ID with a password for the *remotehost.*

1. Insert the CD-ROM into the drive.
2. Login to the *remotehost* as a normal user (not root, Solaris does not permit remote login as root) enter the user's password, and then become root on the *remotehost:*

   ```
   localhost.user1> rlogin remotehost -l user2
   Password:
   remotehost.user2> su
   Password:
   ```

3. The volume manager may have already mounted the CD-ROM. To check, enter `df -k` and look for an entry that looks like this: `/cdrom/vnmr_61b`. This is the entry you will use in the following steps. Otherwise, create a mount point for the CD-ROM drive and mount it on the *remotehost:*

   ```
   # mkdir /cdrom
   # mount -r -F hsfs /dev/sr0 /cdrom
   ```

   If you get the message that the device is busy, the CD-ROM drive is already mounted; you can proceed with the rest of this procedure. Solaris normally mounts local devices during bootup.

4. Check if the file `/etc/dfs/sharetab` exists:

   ```
   # cd /etc/dfs
   # ls
   dfstab    fstypes    sharetab
   ```

5. Whether `sharetab` exists or not, enter the `share` command as shown below. This command enables anyone on the network to mount the CD-ROM. If you want to restrict the systems that can mount the CD-ROM replace `ro` with `ro=hostname1:hostname2:hostname3:...`

   ```
   # share -F nfs -o ro /cdrom/cdrom0
   ```

   At this point `sharetab` exists. It tells the *remotehost* that the CD-ROM can be remotely mounted. If `sharetab` existed in step 3, the next two commands do not need to be executed. Otherwise, execute the following two commands as root:

   ```
   # /usr/lib/nfs/nfsd -a 16
   # /usr/lib/nfs/mountd
   ```

6. Double check with the `dfmounts` command:

   ```
   # dfmounts
   RESOURCE  SERVER      PATHNAME        CLIENTS
   -         remotehost  /cdrom/cdrom0   localhost
   ```

   The name `/cdrom` might also appear under `PATHNAME`.

7. Get back to the *localhost* by entering `exit` twice.

   The first `exit` will make you `user2` on *remotehost*, the second `exit` command will bring you back on *localhost*.

8. Mount the remote CD-ROM drive as follows:

   a. Become root on the *localhost.*

   b. Enter the following commands:

   ```
   # cd /
   # mkdir /cdrom
   # mount remotehost:/cdrom/cdrom0 /cdrom
   (or remotehost:/cdrom)
   ```

You are now ready to install the VnmrJ or VNMR software. For more information use the UNIX command *man* on *mount, umount.*

After VnmrJ or VNMR is loaded, you can disable the access with `unshare /cdrom` or by rebooting *remotehost.* For more information use the UNIX command `man` on `dfmounts`, `dfstab`, `mountd`, `nfsd`, `share`, `sharetab`, `unshare`.

If the Remote Host is Running SunOS 4.1.x

If the remote host is running SunOS 4.1.x, use this procedure. If the remote host is running Solaris, use to the previous procedure.

To mount the CD-ROM drive from the *remotehost*, you need the root password of the *remotehost.*

1. Login to the *remotehost* as root*:*

   ```
   localhost.user1> rlogin remotehost -l root
   Password:
   ```

2. If not already done, make a mount point for the CD-ROM drive and mount it on the *remotehost:*

   ```
   # mkdir /cdrom
   # mount -r -t hsfs /dev/sr0 /cdrom
   ```

If you get a message that the device is busy, ignore it and proceed with the next steps.

3. Using a text editor such as `vi`, edit the file `/etc/exports`. Check if the line below exists, if it does not exist add it. Note if the file is a new file.

   ```
   # vi /etc/exports
   /cdrom     -ro
   ```

   Access to specific systems only can be set by entering the line as

   ```
   /cdrom     -ro,access=localhost
   ```

4. The command below enables the mount of the CD-ROM.

   ```
   # exportfs /cdrom
   ```

5. If the file `/etc/exports` in step 3 was a new file, you must also start two daemons or reboot the *remotehost:*

   ```
   # /usr/etc/nfsd 8 &
   # /usr/etc/rpc.mountd -n
   ```

6. Get back to the *localhost* typing `exit` twice.

   The first `exit` will make you `user2` on *remotehost,* the second `exit` command will bring you back on *localhost*.

After VnmrJ or VNMR is loaded, you can disable the access with `exportfs -u /cdrom`. To permanently disable access, you must remove the line in `/etc/exports` (or remove the file `/etc/exports`). For more information use the UNIX command `man` on `exports`, `exportfs`, `mountd`, `nfsd`, `showmount`, `xtab`.

7. Mount the remote CD-ROM drive as follows:

   a. Become root on the *localhost.*

   b. Enter the following commands:

      ```
      # cd /
      # mkdir /cdrom
      # mount remotehost:/cdrom /cdrom
      ```

You are now ready to install the VnmrJ or VNMR software. For more information use the UNIX command `man` on `mount`, `umount`.

4.2 Shutting Down the Sun Computer

Using the recommended shutdown procedure is essential for preserving data and internal housekeeping information. The Solaris operating system does not necessarily write to the disk when a program tells it to. Rather, data is kept in cache memory until Solaris determines that a write to disk is needed. Data and status information could (and will) be lost if the L1-A or STOP-A key sequences are used or if the power switch is actuated. We recommend using the `init 0` command, `halt` command, or the `shutdown -y -g120 -i0` command, which, in 2 minutes, brings the computer to run level 0 where it is safe to turn off the power.

On Sun Ultra systems, the command **`init 5`** shuts down the systems power.

After you are sure you have saved the files you will need later, you are ready to shut down the system and prepare to install a new version of Solaris. The steps below offer a safe method for shutting down UNIX.

To shut down the system, you must be logged in as `root` from the `login:` prompt (and not from a shell or by entering the `su` command). If you are currently in VNMR, exit VNMR before shutting down UNIX.

1. If not already logged in as `root`, log in and enter the `root` password (if used):
   ```
   login: root
   password: root_password
   ```
2. Change to the `/etc` directory and remove the file `acqpresent`:
   ```
   varian# cd /etc
   varian# rm acqpresent
   ```
3. If you want to shut down immediately, enter the `shutdown` or `init 0` command:
   - Solaris – **`init 0`**
   - SunOS – **`/etc/shutdown -h now`**

   If you want to set a wait period to allow other users to get off the system, enter the `shutdown` command as follows:
   - Solaris – **`/etc/shutdown -y -g`*`sec`*` -i0`**
   - SunOS – **`/etc/shutdown -h`** ***`hour:min`***

   Solaris—For example, the command `/etc/shutdown -y -g30 -i0` sends messages to users warning of a shutdown in 30 seconds.

   SunOS—For example, the command `/etc/shutdown -h 15:30` sends messages to users warning of a shutdown and shuts down the system at 3:30 P.M.

   At shutdown, the system forces any information on its way to the hard disk to be written out immediately, cleans up any processes that are running, and executes an orderly shutdown of the system. The process takes about 20 seconds. When the system is safely shut down, the monitor prompt "`>`" or "`ok`" appears.

4.3 Opening a Host or Terminal Window

This section describes how to open a host window (sometimes referred to as a UNIX shell), which you can use to enter UNIX commands. These windows are the Common Desktop Environment (CDE) version of a command tool.

To open a terminal console or host window, do the following steps:

1. Move the mouse to a blank area on the screen, not on a window.
2. Press and hold the right mouse button. The Workspace popup window appears.
3. While holding the right mouse button, slide the cursor to **Hosts**.
4. In the drop-down, menu, select **This Host** and release the right mouse button.

   The window opens.

4.4 System Requirements

This section lists some Sun and software requirements.

Sun Requirements

Varian currently supports the Sun workstations shown in Table 6. This table also lists the versions of the Solaris operating system that are currently compatible with VnmrJ or VNMR software. If you purchased a Sun workstation from a source other than Varian, or plan to use an existing Sun workstation, any of the Sun workstations listed here are acceptable.

Table 6. Sun Workstations, Sun Architecture, and Solaris Versions

Sun System	*Sun Architecture*	*Solaris Version*
Blade 100 and 150	Sun-4u	9, 8 (10/00 or later)
Blade 150	Sun-4u	9, 8 (10/02 or later)
Ultra	Sun-4u	9, 8, 7

The host workstations must have at least 128 megabytes (MB) of RAM, but 256 MB is recommended. Solaris and VnmrJ require a minimum total hard disk space of 4 GB.

Monitors can be any size, monochrome or color. Graphics can be “plain” or the PGX version (the GX version provides higher performance). Base level graphics work fine with VnmrJ. Higher performance graphics configurations (e.g., Creator, Elite) are also compatible with VnmrJ.

Solaris Software Group Space Requirements

NMR spectrometers require the Entire Distribution plus OEM Support software group. Table 7 lists the space requirements of the Solaris 9 software groups. Table 8 lists the space requirements of the Solaris 8 software groups. Table 9 lists the space requirements of the Solaris 7 software groups.

Table 7. Solaris 9 Software Group Space Requirements

Software Group	*Space (MB)*
Entire distribution plus OEM	2151
Entire Distribution	2114
Developer system support	1927
End user system support	1471
Core system support	716

Table 8. Solaris 8 Software Group Space Requirements

Software Group	*Space (MB)*
Entire distribution plus OEM	1321
Entire Distribution	1296
Developer system support	1249
End user system support	936

Table 9. Solaris 7 Software Group Space Requirements

Software Group	*32-Bit Support (Mbytes)*	*64-Bit Support (Mbytes)*
Entire distribution plus OEM	801	909
Entire Distribution	787	895
Developer system support	716	837
End user system support	438	532

4.5 Disabling and Reenabling a X453A Thinnet Coax Ethernet Board

The X453A Thinnet Coax Ethernet board must be disabled before Solaris is installed and then reenabled after Solaris is installed. The appropriate steps in the Solaris installation procedures refer you to the procedures in this section.

Disabling the X453A Thinnet Coax Ethernet Board

To temporarily disable the X453A thinnet coax second Ethernet board, use these steps:

1. At the `ok` prompt enter the following commands:

   `ok` **`set-default sbus-probe-list`**

   `ok` **`show-sbus`**

   The system lists SBus slot assignments similar to the following:

   ```
   SBus slot 5 ledma le SUNW, bpp espdam esp
   SBus slot 4 aft-misc SUNW, CS4321 power management
   SBus slot 1
   SBus slot 2 le
   SBus slot 3
   Sbus slot 0
   ok
   ```

   The `le` represents the second Ethernet board. In the above example, it is installed in SBus slot 2. Write down the slot number of where your Ethernet board is installed.

2. Enter the following:

   `ok` **`printenv sbus-probe-list`**

   The system displays a series of digits.

   ```
   sbus-probe-list=541230
   ```

3. Enter the following command, with `xxxx` representing the series of digits with the slot number containing the second Ethernet board omitted:

   `ok` **`setenv sbus-probe-list xxxx`**

   For example, entering the command with the list displayed in step 4 above, it would look like `setenv sbus-probe-list 54130`, with the number "`2`" omitted.

4. Enter the following to initiate the changes:

   `ok` **`reset`**

5. Press the Stop–A (or L1-A) keys to stop the system after the Sun banner is displayed.

6. To verify that `le` is not shown next to the slot you omitted in step 6, enter:

   `ok` **`show-sbus`**

To continue with our example, the following list would display:

```
SBus slot 5 ledma le SUNW, bpp espdam esp
Sbus slot 4 aft-misc SUNW, CS4321 power management
SBus slot 1
SBus slot 2
SBus slot 3
SBus slot 0
ok
```

The system now ignores the second Ethernet board while Solaris is installed. After you install Solaris, you will reenable the second Ethernet board.

7. Boot up the Solaris CD-ROM and proceed with the Solaris installation.

To Reenable the X453A Thinnet Coax Second Ethernet Board

The steps in this section describe how to reenable the second Ethernet board. Follow the instructions in this section if you have a second Ethernet board installed and disabled it as described earlier in this manual.

1. Log in as `root` and enter the `root` password you just created:
2. At the root prompt "#," enter:

   ```
   # init 0
   ```

3. When the "`ok`" prompt appears, enter:

   ```
   ok set-default sbus-probe-list
   ```

4. Enter the `printenv` command as follows:

   ```
   ok printenv sbus-probe-list
   ```

   The system displays the following with a series of digits that now include the slot number you previously eliminated:

   ```
   sbus-probe-list 541230
   ```

5. Reset and reboot the Sun computer:

   ```
   ok reset
   ```

   Press the Stop–A (or L1–A) keys to stop the system after the Sun banner is displayed. You might have to switch back to the *new* mode.

6. Enter the `show-sbus` command and verify that the `le` entry is at the SBus slot you omitted earlier:

   ```
   ok show-sbus
   SBus slot 5 ledma le SUNW, bpp espdam esp
   SBus slot 4 aft-misc SUNW, CS4321 power management
   SBus slot 1
   SBus slot 2 le
   SBus slot 3
   SBus slot 0
   ok
   ```

7. Reboot the computer:

   ```
   ok boot
   ```

8. When the system shows the `login` prompt, log in as `root`.

4.6 Collecting System and Network Information for Solaris

The installation program asks you to supply some system and network information before installation begins. You can save time by collecting this information now, before booting from the Solaris CD-ROM.

Use the preinstallation worksheet to record your system information. Each field on the worksheet is described in this section, describing what the field means and if VnmrJ or VNMR has special requirements in that particular area.

Stand-alone, Non-networked System—If your system is not connected to a network, you need to know or create only the host name, root password, and the time zone. If your system is on a network, you need additional information that is described in this section.

Upgrade From SunOS—If your system is already running UNIX (SunOS) and you are going to install Solaris, you must get the information from your current setup as described below. Also, check your setup with your system and network administrators. Be sure you have backed up any information you wish to keep.

Host Name

A computer on a network is often called a host. Its host name is the name that uniquely identifies the computer. When choosing a host name, make sure the name you select is unique within both your local area network and, if applicable, your name service domain.

In many networks, the choice of a host name is left up to the owner of the computer (subject to the requirement of uniqueness). A host name can be up to 64 uppercase or lowercase characters. It is strongly recommended that you use all lowercase characters in the host name because some networking software that might be used in other computers on the network could require lowercase host names. Choose a name that starts with a lowercase letter, followed by any combination of lowercase letters, numbers or hyphens. The name, however, cannot end with a hyphen.

Finding the Host Name

Enter `uname -n` from within a C Shell to display host name information for a Sun computer.

VNMR Requirements

VNMR for UNITY*INOVA*, *MERCURY* series, and *GEMINI 2000* reserve the IP names `gemcon`, `inova`, `inovaauto`, and `wormhole`. Do not use these names for your host name if you are installing VNMR on one of these spectrometers. Names such as `mercury300` or `inova750`, however, can be used. We also recommend a name that is all lower-case.

Primary Network Interface

If your system has more than one Ethernet board installed, you must specify which Ethernet board (network adaptor) is the system's primary network interface.

VNMR Requirements

In order of preference, select `eri0`, `hme0` or `le0`.

IP Address

Your computer must have a unique Internet Protocol network address (IP address) if your computer is to be attached to a network.

Finding the IP Address

Ask your network or system administrator. Alternatively, you can use one of the following UNIX commands:

- If the software is being installed on a computer that is already connected to a network where NIS is present, use the command:
  `ypcat hosts | grep `uname -n``
  Note the use of back quotes (`) to enclose the last command argument.
- If your system is not connected to a network or if NIS is not present, use:
  `grep `uname -n` /etc/hosts`

VNMR Requirements

For NMR spectrometers with no second Ethernet board installed in the host computer, use 10.0.0.1 for the IP address. Otherwise, no special requirements for VNMR.

Name Service Type

The name service prompt allows choosing between NIS+, NIS, Other (DCE) or None.

Finding the Name Service Type

Ask your network administrator what name service the network uses.

VNMR Requirements

None.

Domain Name

The domain name is the name assigned to a group of computers that are administered together. All computers in the group (domain) are accessed by the same NIS or NIS+ maps.

Finding the Domain Name

Your network administrator should be able to provide the domain name. Or the domain name can be found by entering the command `domainname`.

VNMR Requirements

None.

Name Server Host Name and IP Address

If you select either NIS or NIS+ as the name service type, it is assumed that there is a different computer on the network that is the current NIS or NIS+ server. You can select how the name server will be found—allow the software to find one or specify one

explicitly. If you want to specify a name server, you must provide the host name and IP address of the server that provides the name service.

Finding the Name Server Information

On an existing computer, the server's name can be displayed by entering the command `ypwhich`. The server's IP address can be displayed by entering the command `ypcat hosts | grep` `` `ypwhich` ``. Again, note the use of back quotes (`` ` ``).

Also, during the Solaris installation, you can elect to have the system find the name server.

VNMR Requirements

None.

Enable IPv6

Consult your system administrator.

Proxy Server (Sun Blade 100 and Blade 1000)

Finding the Proxy Server Information

Use Netscape to find this information.

1. Open Netscape and select **Edit**, then **Preferences**.
2. In the Preferences window, select **Advanced**, then **Proxies**. If you know the proxy server address, enter it in either the Manual or Automatic configuration fields; otherwise consult your system administrator about network proxies.

Subnet Mask

If your system is part of a subnet, you must provide a the subnet mask number. The subnet mask is a number used to split Internet addresses into the network (Internet) and host parts.

Finding the Subnet Mask Number

If your site does not use multiple subnets, use the default number; otherwise, consult your network administrator. For a computer connected to a network, use the following command:

```
cat /etc/netmasks
```

VNMR Requirements

None.

Setting the Time Zone

Solaris software uses world time zones and automatically adjusts the system clock for daylight-savings time if appropriate. You can set the time zone by selecting geographic region, hours offset from GMT, or by selecting the name of a file in the directory `/usr/share/zoneinfo`.

4.7 Choosing a Root Password

A root password may contain any number of characters, but only the first eight characters of the password are significant. For example, if you enter `a1b2c3d4e5f6` as your root password, then `a1b2c3d4` could also be used to gain root access.

1. Enter your root password, and then enter the same password again to confirm it, as prompted. The system continues to reboot. After the message "`The system is ready`" appears, you are prompted to log in.
2. Type `root` at the `login:` prompt and press Return.
3. Enter your root password at the prompt. The root prompt (#) appears.

4.8 Choosing a User Password

Passwords help maintain system security. You can change a password with the UNIX command `passwd`. When choosing a password, consider the following:

- Select a password at least six characters long.
- Use letters or numbers or a combination of both, but do not use special characters such as `! @ # $ % ^ & * _ + | } { : ? > < \ /`. Many of these characters have special meaning to UNIX and could cause chaos if encountered in a password.
- Use either upper or lower case, but remember that since UNIX is case-sensitive, the password must be entered each time in the same case.
- Select a password that you can easily remember. If you forget your password, you cannot log into the system, and the system does not tell you, or even `root`, the password. The `root` account can, however, reset passwords.
- If extra system security is required, you might consider adding numbers or combining two dissimilar words with no space between the words.

Chapter 5. Files, Permissions, and Owners

Sections in this chapter:

- 5.1 “How to Get File Information,” this page
- 5.2 “Protection Bits,” page 53
- 5.3 “Owners and Groups,” page 55
- 5.4 “Special Permissions,” page 56
- 5.5 “Hard Links, Symbolic Links: the UNIX File System,” page 58

An important activity in UNIX is file handling. This chapter covers how to get information on files, how files are protected from unauthorized use, and how links are used to associate files with each other

5.1 How to Get File Information

The UNIX commands `ls -l` or `ls -lF` can be used for getting a rather complete set of information on a directory or a files in a directory. For example, by entering

```
ls -ldF /dev
```

the information on the directory `/dev` might appear like this:

```
drwxr-xr-x  1  root   wheel    512   Dec 14  9:30  dev/
```

This displays the following information, from left to right:

- Type of file: `d` for a directory, `-` for a plain file, `l` for a symbolic link, `c` for a character special device (usually in `/dev`), and `b` for a block special device (usually in `/dev`).
- Permissions, or protection bits (the next nine characters, `rwxr-xr-x`).
- Number of hard links to the file (`1`).
- Name of the user who owns the file (`root`).
- Name of the group that also owns the file (`wheel`).
- Length of the file, in bytes (`512`).
- Date and time of the last modification (`Dec 14  9:30`).
- File name (with file type indication, if the -F option was used, `dev/`).

Each item is described further in the following sections.

5.2 Protection Bits

The protection bits for files are organized in three groups of three characters, `rwxrwxrwx`, where `r` is read permission, `w` is write permission, and `x` is permission to execute the file.

The first group (rwx) refers to permissions available to the user listed as the owner, the second group to the users in the group listed, and the third group to all users.

For directory listings, the meaning of the protection bits is slightly different: r is still read permission (enables listing the contents of the directory); w is the permission to write something into a directory (create a subfile), change it (rename a subfile), or delete contents (subfiles) of a directory; and x is the permission to make it the working directory (change directory into it). It is possible to have directories that a user can change into, but not read, and vice versa.

The permission flags can be set individually or all at once. Setting them all together is usually easier. The command to set all flags at once has the syntax

```
chmod number filename
```

where *number* is a three-digit number. Each digit has a value from 0 to 7 based on the sum of three permission bits, rwx:

- Read permission (r) adds 4 to a digit.
- Write permission (w) adds 2 to a digit.
- Execute permission (x) adds 1 to a digit.

For example, rwx is 7(4 + 2 + 1), rw- is 6 (4 + 2), and r-x is 5 (4 + 1).

For all three digits, add up decimal values as follows:

400	read permission for the user who owns the file
200	write permission for the user who owns the file
100	execute permission for the user who owns the file
40	read permission for users who belong to the group
20	write permission for users who belong to the group
10	execute permission for users who belong to the group
4	read permission for other users
2	write permission for other users
1	execute permission for other users

For example, to change the permission flags for the file bin/shell_script to rwxrwxr-x, which means no write permission for users outside the group, use the table to find that rwxrwxr-x is 775 (400 + 200 + 100 + 40 + 20 + 10 + 4 + 1), and then enter:

```
chmod 775 bin/shell_script
```

The chmod command has another method for setting individual protection bits. chmod can use the notation u for the owner, g for a group, and o for others, and then uses a plus (+) or minus (–) sign to either add (+) or subtract (-) u, g, and o with r for read, w for write, or x for execute. If any option isn't specified, all three groups are changed. The following examples should make this clear:

chmod g+x file	Adds permission for users who belong to the group to execute the specified file or directory.
chmod +w file	Adds permission for all three groups to write to the specified file or directory.
chmod go-rw file	Removes permissions for the users who belong to the group and all other users to read from and write to the specified file or directory.

There is also a recursive option:

`chmod -R 755 file`	Set `rwxr-xr-x` permission for the specified file or directory and all of their subfiles and subdirectories.
`chmod -R o+r file`	Adds read permission for the other users to the specified file or directory and all of their subfiles and subdirectories.

Setting permissions in absolute mode (e.g., `chmod -R 755 file`) in directories other than the personal data files and libraries can be dangerous, especially with the recursive option. When recursively changing permissions, it is better to selectively add or subtract permissions: the permissions for directories are usually not the same as for plain files, and this precludes combining the `-R` option with the absolute mode.

The permissions within the `root` and `/usr` partitions have been set up to ensure UNIX system security. It is not advisable to change permissions in this area because you may inadvertently erase special permission bits (see below) and eventually make UNIX non-functional and require reloading all the software.

5.3 Owners and Groups

Every UNIX file and directory has an owner assigned to it (as well as the dates of creation and of the last modification). Normally, the owner is the user that has created the file. Only the UNIX system administrator `root` can change the ownership.

Changing File Ownership

The command to change ownership is `chown`, which has the syntax
`chown` *`new_owner filename`*

This command has a recursive option (`-R`) for changing the ownership in all files and subfiles within a directory, for example, `chown -R vnmr1 *`

Although the `ls -l` command displays the owner's user name, the directory only contains the user's ID-number. Most other commands (like `tar` and `ps` with the appropriate options) only list this ID number. Instead of the owner's name, the `chown` command can also take the user's ID-number as an argument.

When `root` is loading the VNMR software, `chown` is used to make `vnmr1` the owner of most files within `/export/home/vnmr`; however, a special problem arises when `root` copies files into some user's home directory. Because `root` has created (copied) the file, `root` still owns it. Therefore, the user may not have permission to use it and, because the file belongs to `root`, the user cannot change the ownership. Whenever `root` copies files into someone else's home directory, `chown` should be used to also transfer the ownership.

The better solution would be if the user reads the file by copying it from outside into his or her own home directory because then, as the creator of the copy, the user becomes the owner of the file. Of course, this only works if the user has read permission for the file.

Changing Group Ownership

Similar mechanisms are used in connection with the group identification. All users are organized in groups (listed in `/etc/groups`) and every user also has a primary group ID-number (defined in `/etc/passwd`). Apart from that primary group membership, any owner can share another group's permissions, in that he can be listed as member of any

number of other groups in /etc/group (of course, by opening up group memberships you are compromising in the area of security).

Users that have been defined with the command /vnmr/bin/makeuser are primary members of the group nmr, and they are all also members of the group staff, which is a standard group on most UNIX systems.

The same as the user ID, the group ID-number of a file can be manipulated. The chgrp command to change group ownership works exactly like the chown command (chgrp also has a recursive option). You must either be root or be owner of the file and member of the specified group to execute chgrp. As with chown, you can also specify the decimal ID-number instead of group name.

It is also possible to change or set the user and the group-IDs at the same time, using chown with the following syntax:
chown *username:groupname filenames*

Of course, also numeric IDs can be specified this way:
chown *userID:groupID filenames*

This syntax differs from SunOS and BSD UNIX, where a point is used instead of the colon.

5.4 Special Permissions

The permissions field can hold three more bits: the sticky bit, the set-UID bit, and the set-GID bit. These bits can be set with the chmod command similar to the other bits. Their decimal values are as follows:

4000	set-UID bit
2000	set-GID bit
1000	sticky bit

How does ls -l display these bits?

- If the sticky bit is set, the last x (execute permission for others) is replaced by a t, or to T if that last bit is not set (e.g., rwxrwxrwt).
- The set-UID bit changes the first x into s, or to S if that x is not set (e.g., rwsrwxrwx).
- The set-GID bit affects the second x the same as the set-UID bit.

The sticky bit is used in connection with executable programs. A program with the sticky bit is *not* swapped out, which can improve the performance of that specific program on systems with limited memory. On the other hand, because this forces swapping to occur with other programs, the performance of those programs may become worse. Because this may affect other users also, only root is allowed to set the sticky bit. The sticky bit is rarely if at all used today—it was created for systems with a small memory (64 MB or less) in the early days of UNIX.

If the set-UID bit is set on an executable file (a compiled program or a shell script), *whoever calls that program automatically bears the user-ID of the owner of the file, while that program is running* (and only inside that program and its child processes).

The set-GID bit works in an analogous way for the group ID-number. This feature is used a lot internally in UNIX, such as functions related to login (the login process is owned by root, after logging in all child processes are owned by the user that has logged in), and in fact many system functions need to change UID or GID.

It is quite obvious that both the set-UID and the set-GID bits are potential security holes. Therefore, these features should only be used where absolutely necessary. Special attention should be paid to programs (like shell scripts) that call child processes and may, in the case of problems, end up in an interactive shell. This may proliferate undesired permissions throughout the system. The most basic safety precaution for such files is, to *make them writable only by the owner*, particularly if they are shell scripts.

A typical example for the application of the set-UID feature is in the handling of optical media, where users should be able to mount and unmount disks but by default these are operations that can only be performed by `root`. Software for handling such removable disks (under SunOS 4.x also floppy disks) is based on shell scripts or programs that has the set-UID permission bit set. In earlier VNMR releases, shell scripts with set-UID permission were one of the possibilities to kill and restart `Acqproc` (which must be run by `root`, because it must be able to update FID files in each user's local directories).

In Solaris 2.x, removable media are automatically mounted via the `vold` volume daemon and handling the `Acqproc` process family is achieved via a special account `acqproc` (with or without a password) that has the same UID as `root` but, unlike a standard user account, does not open an interactive shell script after a login (or `su acqproc`). It only executes the shell script `/vnmr/bin/execkillacqproc`, which is owned by `root` and has the permissions `r-x------` (read and execute permission for `root` only) after that special account has been installed using `/vnmr/bin/makesuacqproc`.

We do not expect that you will ever need to set up software that uses the set-UID or set-GID protection bits; however, you should be abler to recognize files with these permissions, because they can be a serious threat to the security of your system. We will not discuss this in more detail here, but here is a list of files from `/usr/bin` and `/sbin` the use the set-UID permission bit[1]:

```
/sbin:
-r-sr-xr-x   1 root    sys       339780 May  3  1997 su


/usr/bin:
-r-s--x--x   1 root    sys       329360 Apr 26  1997 admintool
-rwsr-xr-x   1 root    sys       33260 May  3  1997 at
-rwsr-xr-x   1 root    sys       12080 May  3  1997 atq
-rwsr-xr-x   1 root     sys      10764 May  3  1997 atrm
-r-sr-xr-x   1 root    sys       19612 May  3  1997 chkey
-r-sr-xr-x   1 root    bin       14604 May  3  1997 crontab
---s--x--x   1 root    uucp      70664 May  3  1997 ct
---s--x--x   1 uucp    uucp      84524 May  3  1997 cu
-r-sr-xr-x   1 root    bin        9676 May  3  1997 eject
-r-sr-xr-x   1 root    bin       26312 May  3  1997 fdformat
-r-sr-xr-x   1 root    bin       28800 May  3  1997 login
-rwsr-xr-x   1 root    sys        9860 May  3  1997 newgrp
-r-sr-sr-x   3 root    sys       15688 May  3  1997 nispasswd
```

[1] If there are substantial differences to how the permission bits are set for these files on your system, you are probably running an earlier version of Solaris 2.x and you probably have not have installed a security patch. Check the documentation that came with the Solaris software for patch information. To find out what patches have been installed on your system, use the command `showrev -p`.

```
-r-sr-sr-x   3 root     sys        15688 May  3  1997 passwd
-r-sr-xr-x   1 root     sys        23740 May  3  1997 ps
-r-sr-xr-x   1 root     bin        18632 May  3  1997 rcp
-r-sr-xr-x   1 root     bin        64748 May  3  1997 rdist
-r-sr-xr-x   1 root     bin        14636 May  3  1997 rlogin
-r-sr-xr-x   1 root     bin         7940 May  3  1997 rsh
-r-sr-xr-x   1 root     sys        15820 May  3  1997 su
-rws--x--x   1 uucp     bin        56500 May  3  1997 tip
-r-sr-xr-x   2 root     bin        11192 May  3  1997 uptime
---s--x--x   1 uucp     uucp       66376 May  3  1997 uucp
---s--x--x   1 uucp     uucp       21428 May  3  1997 uuglist
---s--x--x   1 uucp     uucp       17772 May  3  1997 uuname
---s--x--x   1 uucp     uucp       62096 May  3  1997 uustat
---s--x--x   1 uucp     uucp       70268 May  3  1997 uux
-r-sr-xr-x   1 root     bin         4888 May  3  1997 volcheck
-r-sr-xr-x   2 root     bin        11192 May  3  1997 w
-r-sr-sr-x   3 root     sys        15688 May  3  1997 yppasswd
```

If you intend to use set-UID permissions on your system, check first whether there aren't any other solutions that are less dangerous.

5.5 Hard Links, Symbolic Links: the UNIX File System

In a simplistic view, the UNIX file system looks like a very large tree structure, but it has a number of very confusing aspects, such as different forms of links that may sometimes not be properly hierarchical and file systems mounted from various disk partitions, often including disk partitions from an other computer that are mounted locally via NFS.

Mostly, the user doesn't really care about the more complex aspects, because he or she is mostly concerned with the local file system subtree. Many users rarely look at files outside their own domain. Occasionally, the user stumbles over certain commands that sometimes refuse to work, such as when trying to move a directory (sub-)tree from one partition to an other one using the command `mv`. This section should help clarifying some of these issues.

In reality, the UNIX file system is even magnitudes more complicated than it seems from a user's point of view.

File System Tree

What is the structure of a UNIX file system tree? Most commonly, file system trees are visualized like Figure 4 (the lines are hard links, dotted lines are symbolic links).

This type of file system visualization—although simple and easy-to-read—makes it difficult to understand the difference between hard and symbolic links, and it is impossible to explain why you can have several hard links to a single file (with different names, even). Also, this scheme seems to suggest that files with real names exist on the disk, which is definitely not the case. In reality, files on a disk have no name, but they are stored under a number (the I-node number). For directories, you can assume that they all have the name "/" or "." (they are also stored under an I-node number, as we will see below).

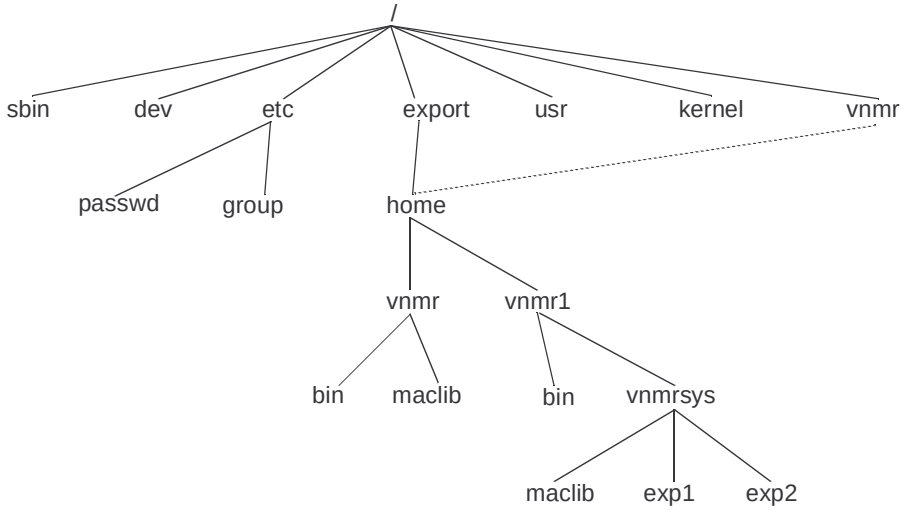

Figure 4. UNIX File System Tree Example

File Storage

Directories, plain files (text files, data files, programs), and even symbolic links are stored the same way— as a tree structure of their own (see Figure 5). At the top of the file tree is the so-called I-node (information node) containing:

- 128 bytes of information that includes the protection bits, the number of hard links to the file, user- and group-IDs, and creation and modification dates.
- 12 addresses to 8-KB data blocks (on the disk, 16 adjacent sectors of 512 bytes are connected to a single data block).
- An address of an 8-KB data block containing 2048 addresses to 8-KB data blocks (the indirect blocks).
- An address of an 8-KB data block containing 2048 addresses to 8-KB data blocks, each containing 2048 addresses to 8-KB data blocks (double indirect blocks).
- An address of an 8-KB data block containing 2048 addresses to 8-KB data blocks, each containing 2048 addresses to 8-KB data blocks, each containing 2048 addresses to 8-KB data blocks (triple indirect blocks).

This results in a maximum size of 64 terabytes (TB) for a single file (70.4×10^{12} bytes).

Of course, real files only use rudimentary parts of this structure, and small files up to 95 KB do not even use the indirect block (the double-indirect block is only used for files larger than 16 MB, and triple indirect blocks would only be used for files greater than 32 GB). To further improve the disk space usage, UNIX stores small files and remainders of big files in so-called fragments (blocks of 512, 1024, 2048 or 4096 bytes), such that the smallest file consists of an I-node (128 bytes) plus a single block (fragment) of up to 512 bytes of data.

One danger with such a file structure is fragmentation. When a disk partition gets full, data blocks are collected from all over the disk. It is for good reason that UNIX tries to keep 10 percent of each partition free. To further improve the disk performance, Berkeley UNIX organizes the disk partitions in cylinder groups and tries to keep each file within a single cylinder group. Each of these cylinder groups includes a [backup]-superblock, a summary block, and an I-node table for disk administration (to keep track of free and used I-nodes, data blocks, and data block fragments). The data blocks and I-nodes are numbered within each partition—this implies that a file cannot extend over several disk partitions[2].

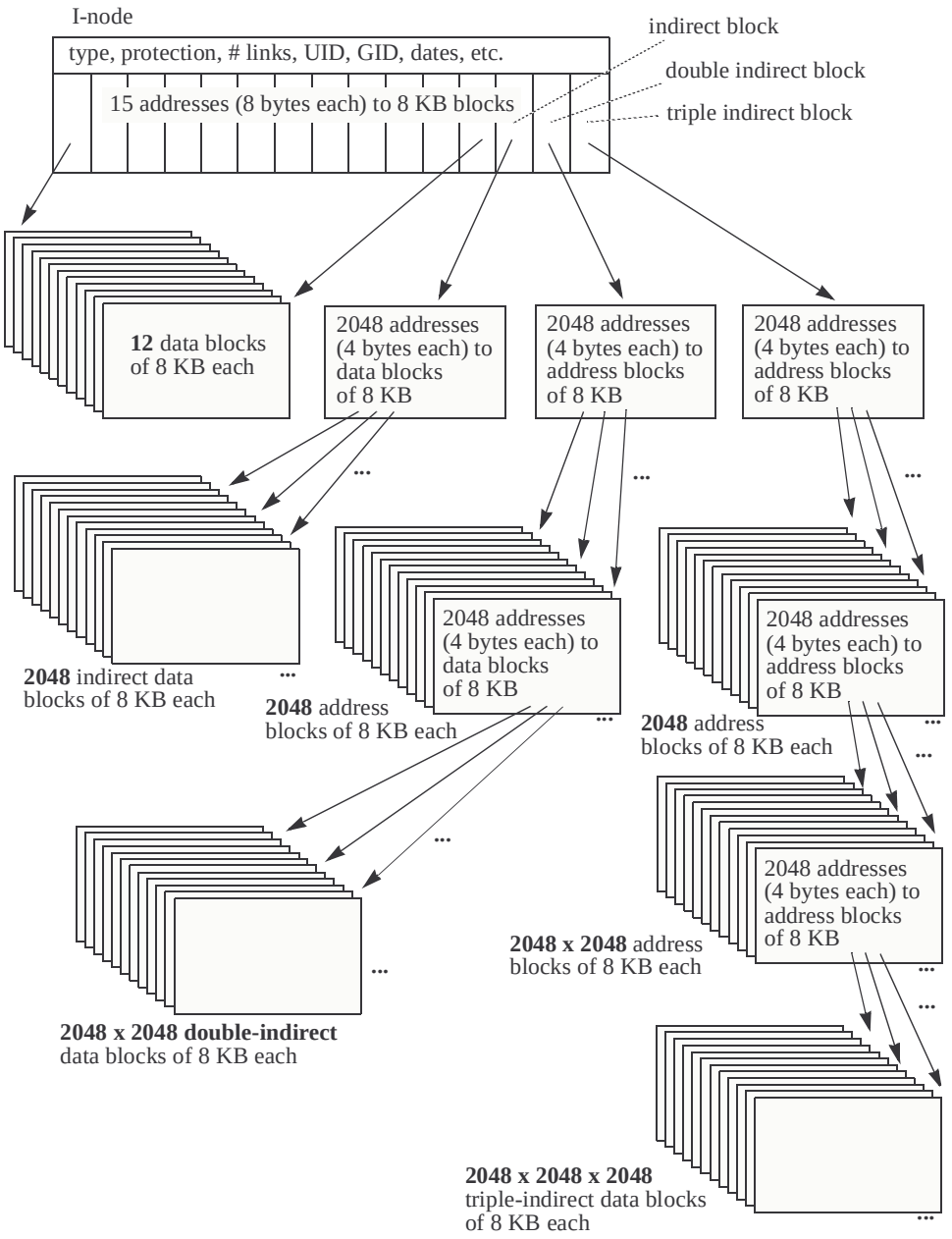

Figure 5. UNIX File Storage

Directory Files

A directory is nothing but a special type of plain file, consisting of an I-node and a variable number of data blocks or fragments. There are two kinds of information in a directory file: the I-node numbers (sometimes called the I-numbers) and the associated file names

2. There is a special software from Sun, Solstice DiskSuite, that permits combining several physical disk partitions to a single (logical) partition. This software is normally only included with larger servers or with large disk arrays, but it can be purchased for workstations, too.

(variable length, up to 255 characters each). Such a directory entry (an I-node number and a file name) is called a hard link. It links a file to a directory. As mentioned above, I-nodes are numbered separately within each logical disk partition; therefore, hard links cannot point to files in a different partition (this can only be done through symbolic links, see "Symbolic Links," page 63).

Hard Links

It now becomes obvious why and how a file can have multiple hard links to it—two directory entries can point to the same I-node. Hard links can only be made to files (I-nodes) on the same partition. The command to create *additional* hard links is `ln` (without the `-s` option), and a listing of the proper contents of a directory (I-numbers and file names) can be obtained using the `ls` command as follows:

```
# ls -ai /
    2 .              6444 devices         243 part
    2 ..             6339 etc            6413 pcfs
  258 .Xauthority    6336 export         6445 platform
 4414 .dt            6490 floppy            2 proc
  199 .dtprofile        3 home             22 sbin
  244 .fm           4242 kernel             2 tmp
 4436 .wastebasket    21 lib                2 usr
  231 TT_DB            3 lost+found        2 var
    4 bin           2128 mnt              250 vnmr
 4408 cdrom             2 net                2 vol
 2115 dev               2 opt                4 xfn
```

Additional hard links can only be made to plain files, not to directories. A link to a directory could lead to cycles in the file system structure (if such a hard link points to a parent directory). Hard links to a file can be made from anywhere within a partition, but, because that can obscure the tree structure, it is *not a generally recommended practice*.

Additional hard links can be deleted (using `rm`) without deleting the file. A file is only really destroyed when the last hard link is removed. Such extra hard links are occasionally used as backup for safety purposes, preventing the loss of data if somebody by mistake erased an important file.

There is one place where (in earlier years) where VNMR used additional hard links. Sun-3 systems were available with two kinds of floating point math hardware—the 68881 math coprocessor and the Sun FPA (floating point accelerator). In order to optimize performance, programs that used floating point math extensively (`fitspec` and the `Vnmr` module itself) were compiled in two versions: one for each of the two versions of floating point math hardware. These modules were stored in `/vnmr/bin` under the names `fitspec_68881`, `fitspec_fpa`, `Vnmr_68881`, and `Vnmr_fpa`. During the installation, appropriate hard links named `Vnmr` and `fitspec` were made to these files (through `setfloat`), and a listing of these files on a machine with Sun-FPA could have been as follows:

```
# cd /vnmr/bin; ls -il fitspec* Vnmr*
2815  -rwxr-xr-x 2 vnmr1  1064960 Oct 12  1992 Vnmr
2814  -rwxr-xr-x 1 vnmr1  1040384 Oct 12  1992 Vnmr_68881
2815  -rwxr-xr-x 2 vnmr1  1064960 Oct 12  1992 Vnmr_fpa
2934  -rwxr-xr-x 2 vnmr1    40960 Oct 12  1992 fitspec
2933  -rwxr-xr-x 1 vnmr1    32768 Oct 12  1992 fitspec_68881
2934  -rwxr-xr-x 2 vnmr1    40960 Oct 12  1992 fitspec_fpa
```

Earlier VNMR releases used yet another link (named `master`) to the `Vnmr` module. Note that the number after the permissions (in the case of plain files) indicates the number of hard links to a file. Only the `-i` option to `ls` allows us to identify which hard links point to a particular I-node.

As shown in Figure 6, we can now visualize the "real" structure of directories and subfiles.

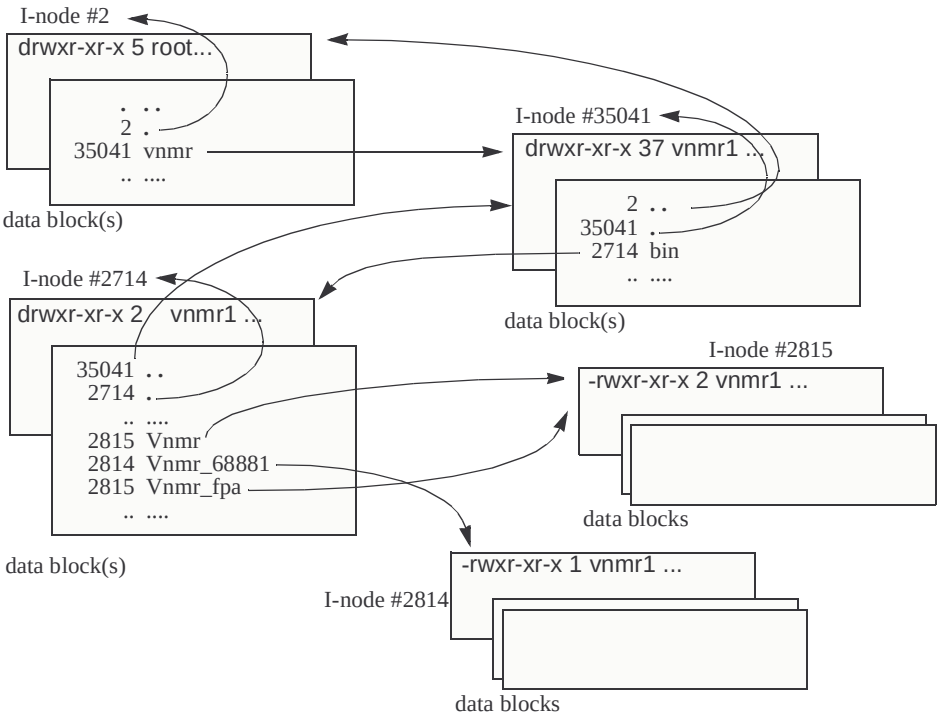

Figure 6. Structure of Directories and Subfiles

In VNMR, there are numerous cases of several macro entries in `/vnmr/maclib` or (even more so) multiple entries in `/vnmr/manual` pointing to the same data file. This is done using symbolic links (see below). In principle, it could also be achieved with extra hard links (that would save disk space), but the problem with this would be that commands such as `cp` or `tar` do not recognize that several hard links point to the same file. Therefore, after copying such a directory using `cp` or (preferably) `tar` (i.e., also after installing the software), you would end up with multiple copies of the same file.

You now can also understand the number of hard links in the output from `ls -l` in the case of a directory. There is one hard link from the parent directory, one hard link (".") from the directory itself, plus one hard link ("`..`") from each of the subdirectories.

It was stated above that a single plain file cannot extend over multiple logical disk partitions. Data blocks are numbered within each logical partition. The same is true for the I-nodes. Therefore, hard links cannot be made to a different logical partition.

It is now obvious that it is not the files themselves that carry the file name but rather the hard links. This leads to a new type of tree structure (rectangles in the diagram are files, rounded rectangles are hard links), shown in Figure 7.

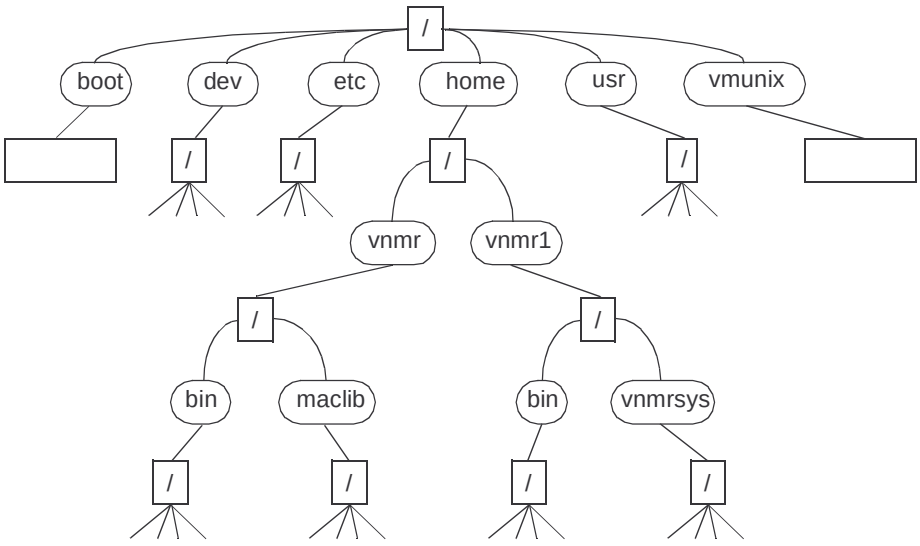

Figure 7. New Type of File Structure

Symbolic Links

As shown in Figure 8 a symbolic link is a plain file with an I-node and a data block, but the data consist of a path name (relative or absolute, see "File Names," page 75). Unlike a hard

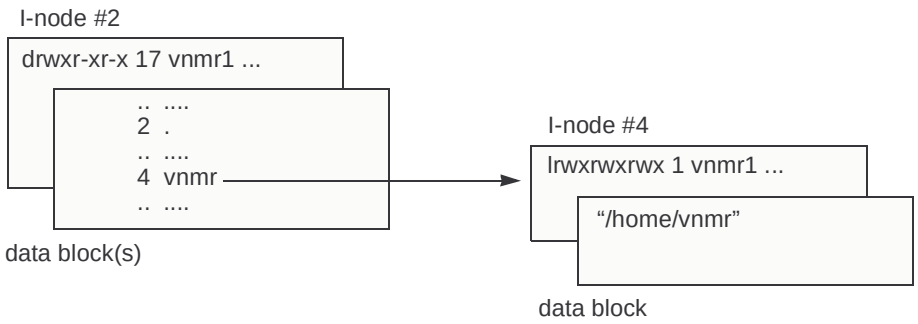

Figure 8. Symbolic Link

link, a symbolic link is an *indirect* connection to another file. Several other differences to hard links exist:

- Symbolic links can also be made between different partitions and disks.
- Symbolic links are not verified upon creation. Only when trying to access a file through a symbolic link is the existence of the target file checked. Symbolic links can be deleted without affecting the target file. If the target file is deleted, the symbolic link remains unchanged.
- Symbolic links can also be made to directories.
- Symbolic links can also be made to parent directories.

Obviously, it is possible to create file system loops if a symbolic link is pointing to its parent directory or if two symbolic links are created that point to each other. In the case of links pointing to each other, the system would get trapped in an endless loop when trying to

access that file. To avoid such cases, UNIX counts the number of symbolic links it passes, and it issues an error message “Too many levels of symbolic links” if it meets more than 20 symbolic links.

The case of a symbolic link pointing to its parent directory can have more serious consequences. The most important drawback of symbolic links is that they are not recognized as such by the `cp` command. The `cp` command copies the file a symbolic link points to, *not the symbolic link itself*.

Suppose a symbolic link inside `/export/home/vnmr` exists, pointing either to `/vnmr` (a symbolic link by itself) or to `/export/home/vnmr`. If `cp` is used to create a backup copy of the entire `vnmr` directory (`cd /export/home; cp -r vnmr vnmr.bk`), then `cp` follows the symbolic link and copies `vnmr` recursively into itself. It may ultimately fill the disk with copies of `vnmr` inside `vnmr` inside `vnmr`, etc.

Fortunately, there is an alternative for recursively copying directories containing symbolic links, involving the `tar` command (`tar` handles symbolic links properly, see “Using tar to Transfer Directories,” page 175). These recommendations should help avoid problems:

- Avoid using `cp -r` whenever symbolic links could be involved.
- Use `tar` when copying directories containing symbolic links.
- *Avoid making symbolic links pointing to a parent directory.*

Chapter 6. File Security, Repair, and Archiving

Sections in this chapter:

- 6.1 “fsck Command,” this page
- 6.2 “ufsdump Command,” page 66
- 6.3 “ufsrestore Command,” page 68
- 6.4 “tar Backups,” page 69
- 6.5 “Backups Using cron,” page 69
- 6.6 “Data Compression,” page 69

Certain non-standard operations can result in damage to the file systems that cannot be repaired, especially when the system was just about to write a file to disk. Causes include:

- Sudden power failure (e.g., switching off the power on a running UNIX system)
- Interrupting operation (e.g. pressing Stop-a or pulling cables)
- Hardware problem
- Booting the system without an automatic file system check

To prevent permanent loss of valuable data, it is essential to have backup copies of all data on magnetic tape. The optimum way of making tape backups is different for the various file systems—it basically depends on how often and how much each file system is changed.

Tape backups are time-consuming. A full backup of a complete disk on streaming magnetic tape can easily block or slow down your system for an afternoon. But this should not be taken as an excuse for not securing the data. Certainly, the backup time has to be balanced against productive time on the system and against the importance of the data on the disk.

Simply recovering from scratch, reformatting the disk and loading all the software from the original tapes may take about 4 hours, and this does not account for data loss and the time it takes to create the data again. The UNIX tool `cron` can be applied to make backups at times of low computer activities (see “Backups Using cron,” page 69).

6.1 fsck Command

Before making backup copies using the `ufsdump` or `dd` command or after a file system damage, checking the file system with `fsck` is strongly recommended. The command `fsck`, which is usually called automatically during a UNIX bootup, scans a entire file system, or even all file systems, for completeness and consistency. `fsck` tries to repair possible inconsistencies. If it cannot make a repair, it notifies the operator and asks for possible actions, such as deleting or clearing the files that are damaged. Only the superuser should run `fsck`, ideally in the single-user mode when no one else is using the system.

In the multiuser mode, there is an additional problem in that some parts of the file system might have a duplicate in memory. After running `fsck`, any `sync` (which happens

periodically due to the `update` daemon and also happens upon operator call or when shutting down the system) will revert any modification in the file system. If after running `fsck`, the message "`FILE SYSTEM HAS BEEN MODIFIED`" shows up, do not use `sync` or `shutdown`, but instead kill the `fsflush` daemon, run `fsck` again, and when `fsck` completes, stop the system by pressing Stop-a (or L1-a). To kill the `fsflush` daemon, find the process-id (enter the command `ps -ef | grep fsflush` and note the PID on the second row of the output line), and enter `kill -9 ###`, where `###` is the value of the PID.

While running `fsck`, some actions, such as answering "`y`" to the prompt "`CLEAR?`", can result in new errors. Therefore, some people suggest running `fsck` twice if any errors are found—in the first run, all prompts except "`SALVAGE?`" and "`RECONNECT?`" are answered with "`n`" in order to find out about the number of errors involved. If the number of errors is high, the original super block is probably damaged. In that case, you should try to replace the super block by one of the backups.

The first backup is in block 32 in each file system. Further block locations can be displayed by the command

```
newfs -Nv /dev/rdsk/c0t3d0s7
```

having the partition `c0t3d0s7`. The replacement is done by entering

```
fsck -F ufs -o b=32 /dev/rdsk/c0t3d0s7
```

Then try `fsck -n` again. Hopefully, the number of errors is decreased.

All you can do in the second run is to answer "`y`" to all questions, forcing the file system to be consistent again. Certain files may be lost completely, while others may be found under a different name in the `lost+found` directory of the corresponding file system.

Whenever `fsck` has modified a file system, check the file system once more to make sure that it is permanently consistent. Normally, this is done automatically during the reboot after `fsck`.

The following examples illustrate `fsck` command syntax:

`fsck /dev/rdsk/c0t0d0s0`	Standard form for interactively checking a single partition
`fsck -n /dev/rdsk/c0t0d0s0`	Run `fsck` with all questions automatically answered "no"
`fsck -y /dev/rdsk/c0t0d0s0`	Run `fsck` with all questions automatically answered "yes"
`fsck`	Interactively check all file systems (as listed in `/etc/mnttab`)

6.2 ufsdump Command

For disk partitions or file systems that are less subject to changes, such as the `/usr` partition, the `ufsdump` command is the best choice for creating backups. In case of symbolic links, the `ufsdump` command copies only the link not the data of the link target (different from the `cp` command).

`ufsdump` is also the most efficient tape command, allowing backing up tapes to their full capacity. By automatically prompting the user for a new tape, `ufsdump` allows dumping large disk partitions onto multiple tapes. A level 0 dump is recommended immediately after installing the complete software. After that, perform the less time-consuming level 1 dumps

about once a week, or even less often for the `/usr` partition. Level 1 dumps copy only files that are new or have changed since the last level 0 dump.

Dumps should be made in single-user mode, or after unmounting the partition, or at least when no other operator is using the system. Otherwise, some files might change during the dump, which can lead to possible inconsistencies. It is also recommended switching between two sets of tapes for each dump level.

For a level 0 dump of `/dev/dsk/c0t3d0s5` partition, use the command:

```
ufsdump 0ucbfs 40 /dev/rmt/0 590 /dev/dsk/c0t3d0s5
```

For a level 1 dump of `/dev/dsk/c0t3d0s5`, use

```
ufsdump 1ucbfs 40 /dev/rmt/0 590 /dev/dsk/c0t3d0s5
```

In both commands, the `40` is the blocking factor (option `b`), option `c` means cartridge tape and the `590` is the usable tape length (option `s`), in feet. The usable tape length is usually calculated as 0.95 times the real length of the tape, in feet. Table 10 gives guidelines for usable tape length.

Table 10. Usable Tape Lengths of Tape Cartridges

Tape cartridge	*Real tape length*	*Usable tape length*
QIC, 150 MB, DC-6150	620 ft	590 ft
QIC, 2.5 GB, DC-9250	900 ft	860 ft
4 mm DAT, 5 GB	90 m	280 ft
4 mm DDS-2, 8 GB	120 m	375 ft
8 mm Exabyte, 2.5 / 5 / 10 GB	112 m	350 ft
8 mm Exabyte, 14 GB	160 m	500 ft

When using standard media, both the blocking factor and the usable tape length don't need to be specified. A suitable, time-saving blocking factor is automatically used with Sun tape drives, and `ufsdump` knows about the length of standard media. This allows simplifying the two commands above to

```
ufsdump 0ucf /dev/rmt/0 /dev/dsk/c0t3d0s5
ufsdump 1ucf /dev/rmt/0 /dev/dsk/c0t3d0s5
```

The `0` option specifies the full backup (level 0). The `u` option updates the log file `/etc/dumpdates`, which is necessary for a later restoring. The `f` option specifies the target device or file.

The user partition `/usr` is virtually read-only and, in fact, is seldom altered after installing the complete software. A single backup of that partition should do for the entire lifetime of the software. Frequently used partitions should be dumped by the scheme explained in detail at the end of the Sun manual for `ufsdump` (enter `man ufsdump` in a UNIX shell). Table 11 repeats this dump scheme.

Table 11. Dump Scheme for Frequently Used Disk Partitions

	Sun	*Mon*	*Tue*	*Wed*	*Thu*	*Fri*
Week 1	0 (full)	5A	5B	5A	5B	3A
Week 2		5A	5B	5A	5B	3B
Week 3		5A	5B	5A	5B	3A
Week 4		5A	5B	5A	5B	3B

The numbers 0, 3, and 5 in this table refer to `ufsdump` levels. Such a dump scheme guarantees that any lost file can be recovered from the last day's backup, while minimizing the dump time:

- At the start of every four-week period, a full file system dump is made.
- The level 3 dumps on Friday record all changes since the last level 0 dump.
- The level 5 dumps on the other weekdays only dump files that have changed since the last lower level dump (level 0 for the first week, level 3 for weeks 2, 3, and 4).

This scheme maximizes archiving security while minimizing the dump time. The frequency of the backups can of course be adjusted, depending on how heavily a system is used, and on how important data security is regarded, but it is strongly recommended to stick to a fixed scheme; otherwise, doing the backups is easily forgotten.

The characters A and B in the table indicate tape sets. For optimum security, switching between two sets of tapes for each dump level is recommended, for the case where something happens *during* a backup. A simplified version of this scheme would be, for instance, to do only level 0 dumps on alternating tape sets every weekend.

Dumping the root partition (/) may not be so useful. On one hand, you are working with this partition while dumping it. `ufsdump` keeps a record of dumped files in the root partition, and files for the `ufsrestore` command could be damaged in case of problems with the root partition. A better solution here is to make a backup of files that have changed since installation. To find customized files in `/etc` (such as hosts, printer administration files, etc.) and to save them on a tape, enter:

```
su root
cd /
tar cvf /dev/rmt/0 'find etc -mtime -days -a -type f -print'
```

The command in back quotes is executed first. *`days`* must be replaced by the number of days since the installation. Don't forget to also save customized files such as `.rhosts`, `.profile`, `.login`, and so on. These file names can be appended to the above command. In case of a damaged root partition, you can reload UNIX from the original CD-ROM, preserving all partitions other than root, and reload all files above at the end.

6.3 ufsrestore Command

If file system errors need to be corrected, it is recommended not to restore the whole partition, but to selectively recover damaged files. To avoid causing additional damage, it is best to restore the files first into a temporary directory such as `/tmp`:

```
cd /tmp
ufsrestore xvbf 40 /dev/rmt/0 etc/inet/hosts
```

The last argument is always a relative path, because `ufsdump` stores the files relative to the mount point.

```
cp /tmp/etc/inet/hosts /etc/inet/hosts
```

To restore files selectively, using the interactive option is recommended.

```
ufsrestore bfi 40 /dev/rmt/0
```

gives the prompt "`ufsrestore>`". Enter `help` at this prompt to see the interactive command options.

The blocking factor only needs to be specified if a non-standard blocking factor was specified with the `ufsdump` command. If the default blocking factor was used, the above two `ufsrestore` calls can be simplified to

```
ufsrestore xvf /dev/rmt/0 etc/inet/hosts
```

or, for the interactive version:

```
ufsrestore fi /dev/rmt/0
```

With most disk slices you will first need to create a temporary directory, such as `/export/home/tmp`, for restoring individual data files. If you ever want to restore an entire disk slice, you must of course *not* use a temporary directory, otherwise you would duplicate the contents of the entire slice, and you would most likely run out of disk space. It should rarely be necessary to restore an entire disk slice.

6.4 tar Backups

Because files in user home directories are changed often during day-to-day operation, doing manual tape dumps would be inefficient and time-consuming for the superuser. On the other hand, users do not have the permission to use `ufsdump`. So every user will probably automatically use the `tar` program (see “tar Command,” page 171) to make individual backups (archives) of his or her NMR data and the other customized files in the user’s home directory.

6.5 Backups Using cron

In multiuser environments, frequent backups are highly recommended. The `cron` command can be quite helpful in this case. It executes jobs at a defined time according to a list in `/var/spool/cron/crontabs/user`. Each user, including `root`, has a `crontab` file named `user`. Editing this file directly has no effect. To activate entries, each user enters the `crontab -e` (edit option) command.

Sometimes an uncommon line editor is defined as the standard editor. You can change to the `vi` editor by setting the environment variable `EDITOR` to `vi` (as `root`, use `EDITOR=vi; export EDITOR`).

To make a backup of `/data` (partition `/dev/dsk/c0t3d0s7`) at 4 o’clock in the morning, from Monday through Friday, you (as `root`) have to enter:

```
0 4 * * 1-5 commands
```

where *`commands`* are UNIX commands separated by semicolons. Here the string *`commands`* is replaced by, for example,

```
cd /; ufsdump 1ucbfs 40 /dev/rmt/0 590 /dev/dsk/c0t3d0s7
```

Messages from commands that usually go to the standard output are mailed to the user, which allows checking for execution and errors. Make sure that a tape drive is connected to the system and a tape of high enough capacity is loaded. Two tapes should be used, at least exchanging them from backup to backup (see above).

6.6 Data Compression

Over the years, the capacity of disks and tapes has grown continuously. At the same time the requirements on disk space have increased and, with the advent of multidimensional NMR, the size of the FID alone can exceed the size of a single 1/4-inch tape of 150 MB. Data compression is a very helpful tool for reducing the size of files and reducing time especially for archiving on tapes.

Often the size of NMR data, especially if multidimensional or if acquired in 32-bit format, can be reduced by over 40%. Of course, the reason that data files on disks aren't compressed in general is that compressing and uncompressing data takes time and requires additional manipulations. Data compression therefore is suitable for mid-term disk storage and especially for archiving data on tape.

What Files Can Be Compressed?

Any plain file can be compressed: data files, text files, and even compiled programs. Directories cannot be compressed directly (discussed below). The amount of compression depends on the internal structure of the file. Table 12 provides some examples.

Table 12. Examples of File Compression.

File	*Kind*	*Before (KB)*	*After (KB)*	*Reduction*
`/vnmr/bin/Vnmr` (Sun-4)	compiled	1696	952	44%
`/vnmr/parlib/dept.par/procpar`	ascii (text)	14	6	57%
`/vnmr/fidlib/fid1d.fid/fid`	binary, 32 bit	128	51	60%
`/vnmr/fidlib/fid2da.fid/fid`	binary, 32 bit	1056	704	33%

The largest reductions can normally be obtained for ASCII text files and compiled programs. Other binary files such as FIDs are sometimes more difficult to compress (particularly if FIDs were acquired in 16-bit (`dp='n'`) mode or if the VNMR `compressfid` command was used). It is possible for a compressed file to be larger than the source file: for example, `/vnmr/fidlib/dept.fid/fid`). In such a case the `compress` command leaves the source file untouched (unless the `-f` option is used).

In general, FID files, together with their parameters and text files, can be reduced by 30% to 40%; therefore, it is worth the effort. Especially for archiving data, the tape capacity can be increased up to about 50%, with 35% an average reduction, and the time for compressing and uncompressing the data is normally shorter than the transfer time to and from the tape.

Note that 4-mm DDS-2 and 8-mm Exabyte tapes in high-density mode use *compression software built into the firmware of the tape drive*. As compressed files usually cannot be compressed any further, it is questionable whether it is worth compressing data files before saving them on such tapes.

Of course, compressed files cannot be used directly. They must be uncompressed first using the `uncompress` command.

Procedures for Compressed Archiving

FID files are actually directory files that contain the FID, a parameter file (`procpar`), and a text file. Because directory files cannot be compressed directly or recursively, each individual subfile in a FID file would have to be compressed and uncompressed. Because this would be tedious, most people normally combine directories into a single disk file using the `tar` command, as follows:

`tar cf` *`directory.tar directory`*

This `tar` file can now be compressed in one step. The target file name can be selected freely, but for an easier identification a suffix `.tar` is usually added to the file name. After

that, the source directory file can be deleted. This step reduces the disk space requirement and can be regarded as a first level of compression.

On what level should the `tar` command be used? There is a temptation to use the above procedure on complex directories containing lots of subdirectories and files, but that does not make the compressed data transparent enough, because subfiles are “invisible” within a `tar` file. Using `tar` on individual FID directories is recommended. The `compress` command adds a further extension `.Z` to the file name, and the source file-ID is deleted automatically. A typical procedure for compression might be the following:

```
tar cf sample.fid.tar sample.fid
rm -r sample.fid
compress sample.fid.tar
```

This results in a single, compressed disk file `sample.fid.tar.Z`. In case the `compress` command cannot reduce the size of the file, it leaves the file untouched (and not append the `.Z` extension). If the `-v` option is used with the `compress` command, it reports the percentage of compression.

The advantage of keeping the complete original file name by adding suffixes is that original file names can easily be recognized when looking for specific data. Keeping the suffix `.fid` is also recommended, because that suffix can distinguish compressed FID files from other compressed files.

The procedure for uncompressing such files is as follows:

```
uncompress sample.fid.tar
tar xf sample.fid.tar
rm sample.fid.tar
```

The `.Z` extension can be omitted with the `uncompress` command.

Due to the `tar` command, this procedure temporarily requires almost twice the disk space before the duplicate file is removed, during both compressing and uncompressing. The `compress` command itself temporarily requires space for the full (`tar`) file plus the compressed file. This means that when the disk is 100% full, it is too late for starting to compress data. In such a case, one way is to store the uncompressed file on a tape, delete the source file, read the `tar` file off the tape using `dd` into standard output, and then compress from standard input to create the compressed file directly:

```
tar cvf /dev/rmt/0 sample.fid
rm -r sample.fid
dd if=/dev/rmt/0 | compress > sample.fid.tar.Z
```

Notice that with this procedure `compress` does not generate the file name automatically.

Both the compression and the decompression can be done in a single step if the `-c` option of the `uncompress` command is used:

```
tar cf - sample.fid | compress > sample.fid.tar.Z
rm -r sample.fid
uncompress -c sample.fid.tar.Z | tar xfB -
rm sample.fid.tar.Z
```

The disk space requirement of this method is between the step-by-step method and the method with the tape. Both the original files and the compressed file coexist after the compression and decompression. The compressed archive files generated with this last method are compatible with step-by-step decompression. Selective extraction is possible with both methods.

To obtain a catalog of a compressed `tar` file, enter:

```
uncompress -c sample.fid.tar.Z | tar tfB -
```

Or even simpler, using `zcat` instead of `uncompress -c`:

```
zcat sample.fid.tar.Z | tar xfB - files
```

To selectively and specifically extract a subfile from a compressed `tar` file type

```
zcat sample.fid.tar.Z | tar xfB - files
```

Chapter 7. UNIX Commands

Sections in this chapter:

- 7.1 “Command Syntax,” this page
- 7.2 “File Names,” page 75
- 7.3 “Important UNIX Commands,” page 77
- 7.4 “Interactive Work with the C Shell,” page 80
- 7.5 “Shortcuts within Open Look and CDE,” page 81

Modern software user interfaces make administration less and less dependent on UNIX knowledge. Under OpenWindows, for example, Admintool simplifies administration of users, groups, hosts, remote and local printers, serial ports, and UNIX software. File Manager, Text Editor, Print Tool, Tape Tool, and Mail Tool are also convenient mouse-driven OpenWindows utilities.

However, there are still a number of cases where the System Administrator needs to go to the UNIX operation level. This chapter describes UNIX command syntax and file names, and lists a selection of UNIX commands, many of which you will find of importance.

7.1 Command Syntax

UNIX command names can be any reasonable length. They typically contain a combination of alphanumeric characters, including special characters such as the underscore. UNIX is case-sensitive, which means it always distinguishes between uppercase and lowercase.

The UNIX command separator is the semicolon or the Return key. If several commands are entered on the same line, they must be separated by semicolons.

Command Arguments

The UNIX argument separator is the space. Arguments are separated from the command and from each other by spaces.

Argument types are numeric, names, strings, or options. Simple strings (no spaces or special characters) can be treated like names using the space as a delimiter; complex strings must be enclosed in double straight quotation marks (“...”). Characters within strings which belong to UNIX-shell commands like a single quotation mark must be escaped with the backslash character (\) (see UNIX manuals for additional details). Options are usually a selection of characters, preceded by a minus sign (`-`). Some commands, such as `tar`, have a special syntax, where the minus sign must *not* be used for the options.

Online Command Reference

Online help on UNIX command syntax is available with the `man` command. To display help on the `tar` command, for example, enter:

```
man tar
```

This only works if the manuals were installed onto the system. The manuals are stored in several megabytes of unformatted `nroff` files. The `nroff` command slows down the manual display, but manuals can be preformatted with the `catman` command. This takes a several minutes to process. After preformatting, the original `nroff` files can be deleted with `rm -r /usr/man/man*`. This speeds up the `man` command without using up much more disk space.

The `man` command does not know about aliases. Entering `man lf` results in an error message. Use `alias` to find out about all presently defined aliases in the current C shell.

Command Grouping

Commands can be executed in a separate shell if they are enclosed in parentheses (...). This has two consequences—their action does not affect the current shell and they behave like a single command for the calling shell. For example, if the directory should be changed for only a few commands, execution in a separate shell does not change the current working directory, for example:

```
(cd /vnmr; mkdir test)
```

If several commands inside parentheses produce output, this output is concatenated, for example:

```
(echo "file listing:"; ls -l) > listing
```

As long as a command is running, it runs exclusively in the shell in which it has been submitted, and standard output, as well as any error output, is made to the same shell.

Background Execution

Commands can also be executed in background. This means that the shell may accept and execute one or more of the next commands while the previous command is running. Background calls are made by attaching an ampersand (`&`) to the command, for example opening `mailtool` in background operation:

```
mailtool&
```

Input and Output Redirection

With any command that makes standard output, this output can be rerouted into a file:

`ls`	Prints the catalog of the current working directory
`ls > listing`	Writes the catalog into the text file *listing*
`ls >> listing`	Appends the catalog to the text file *listing*

If the file does not exist yet, it is created by the command. If the file does exist, the command `ls > listing` overwrites its contents.

Commands that expect keyboard input (called standard input) can take that input from a text file instead:

`wc -w < listing`	Counts the words in the text file *listing*.

Output from one command may be directly used as input for the next command. This UNIX feature is called a pipe and is represented by a vertical bar (|):

`ls -l | more` Displays a long listing with the `more` command.

Multiple pipes can be chained together, for example:

```
ls -lR | grep root | more
```

Command substitution can be used to insert the output of a command into a text string:

```
set prompt="`uname -n`:`who am i` \!>"
```

This inserts the output of the commands `uname` and `who am i` into the prompt string. Commands in back straight quotes (`...`) are executed first, and their output substitutes for the quoted strings before the command line is interpreted completely.

7.2 File Names

UNIX file names can be almost any length (up to 256 characters) and can contain both uppercase and lowercase characters, numbers, and some special characters (including the underscore, dot, hash sign, and plus sign).

File Name Suffix

Some commands and shell scripts expect file names with a specific suffix, such as `filename.c` for C programs), but generally a suffix has no function under UNIX. Therefore, users are free to make up own set of suffixes, perhaps using `filename.old` for annotating old versions of files, `filename.bk` for backup copies, and so on.

In fact, the suffix can be used as a standard file-name extension. VNMR uses the `filename.fid` suffix for FIDs, `filename.par` for parameters, and various others. Only one suffix per file name is recommended. If the file name is very long and needs some structuring, you can use underscores or uppercase characters.

Dot Files

A special class of files, called *dot files*, has a file name that starts with a period (e.g., `.login`). Dot files are not displayed in standard file listings using the command `ls`. To include dot files in the file listing display, enter `ls -a`. See Chapter 10, "UNIX System Customization," for more information on system dot files.

File System Hierarchy

The slash (/) is used to separate directory levels. Under UNIX, a file name, whether simple or complex, is also called a path because the name describes a path to a specific file. There are two classes of paths:

- Absolute paths start at the root (/) and lead all the way down to the file, for example, `/export/home/vnmr1/vnmrsys/exp1/acqfil/fid.`
- Relative paths reference the current directory the user is in, for example, `vnmrsys/exp1/acqfil/fid`.

The current directory of the user is commonly called the *working directory*. The path to this directory can be displayed with the command `pwd`. The working directory also has a file name: the dot (.). Therefore, the relative paths `vnmrsys` and `./vnmrsys` are identical.

The directory that automatically becomes the working directory after login is called the *home directory*. From within a C shell (`csh`), a user can reference his or her home directory with a tilde "`~`". Another user's home directory can be referred to with the file name `~username`. Note that within a Bourne shell (see "Bourne Shell Scripts," page 103) these abbreviations are not valid.

The directory above the working directory is called the *parent*. This directory has two dots (`..`) as the file name (e.g., `cd ..` changes the working directory to the parent directory).

Wildcard Characters

Most UNIX commands accept wildcard characters in file names:

- An asterisk (`*`) stands for any number of characters.
- A question mark (`?`) stands for a single nonblank character.

By using square brackets `[...]`, restrictions for single character positions are also possible:

- `[a-z]` stands for any lowercase character.
- `[abc]` stands for either a or b or c.

These wildcards can also be combined, such as `??*` stands for "at least two characters". The following are examples of wildcard characters in use:

`rm *.o`	Removes all files with `.o` suffix
`ls a*`	Lists all file names starting with `a`
`rm -r par[2456]00`	Removes files `par200`, `par400`, `par500`, `par600`
`ls .??*`	Lists all dot files with at least 2 characters after the dot.

Keyboard Shortcuts for Entering File Names

If you are using the C-shell and the `filec` option is set by entering

```
set filec
```

(or by including `set filec` in your `~/.cshrc` file for permanent setting, undo it by the `unset` command), the Esc and Ctrl-d keys are redefined in certain situations to display file names in the path you are entering. For example, enter

```
ls /vnmr/n
```

and press the Esc key. Upon pressing Esc, the system completes the command line to

```
ls /vnmr/nuctables
```

You can now press Return to execute the line.

For another example, enter

```
ls /vnmr/a
```

and press Esc. The system fills in two characters

```
ls /vnmr/acq
```

and then beeps, because `/vnmr` contains several files that start with the character `a`—`acq` is the common part of all files starting with `a`. The beeping can be switched off with a line

```
set nobeep
```

in your `~/.cshrc` file. You could now continue by typing one or two more characters, and then press Esc again, and so on.

You can also reduce the number of files that the Esc option includes. For example, including

```
set fignore .o
```

in your `~/.cshrc` file causes it to disregard files with `.o` suffix.

Before or after using Esc (in fact anywhere within a command line), you can press Ctrl-d. The system displays a list of files that match the characters in the last token (such as `ls a*` or `ls acq*`) and then repeats the prompt and the previous (unfinished) command line, allowing you to make a selection. For example:

```
> ls /vnmr/user [Ctrl-d]
user_templates/ userlib/
> ls /vnmr/userlib
```

This is a special use of Ctrl-d. Under other circumstances, Ctrl-d works differently.

Note that with the `filec` option, Ctrl-d always has this functionality in a C shell. As soon as the first character on a command line has been typed, it interprets the last token on the line as a file name and tries to complete it. Therefore, it is impossible to kill a window or log out with Ctrl-d inside a command line. It may be necessary to make the system forget the partial command line by pressing Ctrl-u first.

7.3 Important UNIX Commands

Some of the more useful and important UNIX commands are listed in this section. Commands listed below in bold are important for standard UNIX users; the other commands are often found in shell scripts or mainly important for UNIX administrators. Refer to Sun manuals for complete information on UNIX commands.

Directory Commands

pwd	Displays working directory.
cd	Changes to home directory.
cd *directory*	Changes working directory to *directory*.
ls	Lists files in working directory.
ls *directory*	Lists files in *directory*.
lf	Alias of `ls -F` (defined in the file `.cshrc` distributed with VNMR) that is like `ls` but with file type annotation (`/` marks a directory, `@` a symbolic link, and `*` an executable file).
ll	Alias of `ls -l` (defined in the `.cshrc` file distributed with VNMR) that is like `ls` but with additional file details.
mkdir *directory*	Creates new directory with the name *directory*.
rmdir *directory*	Removes directory with name *directory*, provided *directory* is empty (i.e., has no files).

File Handling Commands

cp *file1 file2*	Copies file named *file1* to file named *file2*. If the name *file2* exists, it is overwritten with no warning message (unless the `-i` option is used). `cp` works only on simple files, not on directories.

cp -r *file1 file2*	Makes a recursive copy for a directory and all subfiles. It does not recognize symbolic links (use `tar` instead). Beware of recursively copying a directory into itself—this can quickly fill a file system, and the resulting infinitely stacked directories is difficult to clear.
mv *file1 file2*	Moves one or more files (*file1*) into another directory (if *file2* is a directory). If *file1* has new names, it also renames the files. If *file2* already exists and is not a directory, the file is lost without warning message (unless the `-i` option is used). Directories cannot be moved between different file systems (partitions); use `cp` instead.
rm *files*	Removes one or more files (*files*), accepts wildcards (be careful), but does not work for directories, which use `rmdir` or `rm -r` instead.
rm -r *directory*	Removes directory named *directory* recursively with all of its subfiles.
tar	Reads and writes files and directories (see Sun manuals and Chapter 15, "Using Tapes and Floppy Disks," for details). `tar` may also be used to copy directories recursively across a network.
`ln -s` *file1 file2*	Creates symbolic link for *file1* to destination path specified by *file2*.
`touch` *files*	Creates or updates directory entry for one or more files (*files*) If file does not exists, creates a file of size zero; if file exists, updates date of last modification.
`find`	Searches in a given directory, downward through all directory levels, trying to the find one or more files. As shown with the next few examples, `find` has various options: where to search, by what criterion to find files (for example, by name, owner, size, permission, access date, modification date, owner, and number of links), what to do with the files found, and what to do with the information (print it or execute some other command).
`find /home -name "*.old" -user vnmr1 -print`	Finds all files with `.old` suffix within directory `/home` that belong to user `vnmr1`, and print the paths to the files.
`find / -name core -exec rm {} \;`	Finds all files named `core` and removes them. Note the notation for the arguments to the `rm` command, and the backslash for the semicolon (the semicolon should not be interpreted by the shell as command separator, but as a terminator for the arguments of the `rm` command)
`find / -atime -3 -print`	Finds all files that have been accessed within less than 3 days and prints their paths.
`find / -mtime +30 -print`	Finds all files that have last been modified more than 30 days ago and prints their paths.
`chmod`	Change permissions (see Chapter 5, "Files, Permissions, and Owners," for further information).
`chown`	Change owner (see Chapter 5, "Files, Permissions, and Owners," for further information).

Text Commands

cat	Displays text from text files or writes it to standard output.
head	Displays text from the start to a specified point in a text file or from standard input.
tail	Displays text from a specified point to the end of a text file or from standard input.
cut	Cut out the specified character or token column(s) from a file or from standard input.
fold	Displays text from a text file or standard input, wrapping around (folding over) long lines.
more	Displays text from a text file or standard input in pages with the following keyboard commands: press Spacebar to scroll to the next page, Ctrl-b to scroll back one page, Return to scroll down one line, and q to quit. When more is used on a text file, v switches to the vi editor (after quitting vi, more becomes active again).
page	Similar to more, but erases and redraws screen (window). page is somewhat faster than more and uses the same keyboard commands as more.
vi	Edits a text file (see "Text Editor vi," page 164)
textedit	Edits a text file (see "Text Editors dtpad and textedit," page 163)
sed	Edits a stream of text.
diff *file1 file2*	Prints the differences between two files.
diff -r *dir1 dir2*	Recursively compares two directories and their subfiles.
grep *pattern files*	Searches for a pattern within one or more text files.
grep -l *pattern files*	Returns the names of files that contain the specified pattern.
awk	Scans and processes text file line by line.
tr	Translates characters, for example, converting upper case to lower case.
fmt	Format ASCII text: wrap lines in paragraphs, does not split words (unlike fold).
sort	Sorts lines in a text file.
nroff	Formats text for character-oriented terminals.
troff	Formats text for pixel-oriented terminals.
echo	Displays the text string that is given as an argument.

Process Management and System Administration Commands

df -k	Displays percentage of free disk space in each (or specified) partition.
du -k	Displays size in kilobytes of files in the current working directory and all subfiles.
du -k *file*	Displays size in kilobytes of all subfiles of file *file*, including their subfiles.
du -sk *files*	Displays total size in kilobytes of one or more files (*files*), including all its subfiles.
du -sk	Displays total size in kilobytes of the working directory.
su	Requests changing user-ID to become root. If passwords are implemented, su asks for the root password.

`su - user`	Request changing user-ID to `user`. The first argument [-] changes the environment to what would be expected if the user actually logged in as the specified user. If passwords are implemented, `su` asks for the `user` password.
`su root -c 'kill #'`	Kills a process with PID # as a temporary superuser. Asks for the superuser (`root`) password.
Ctrl-d	Logs user out. Exits the current window or subshell.
`ps -ef`	Displays all process-IDs and status of current processes.
`kill`	Sends signals to specified processes (requires process-IDs); the default signal is 15 (TERM) and causes most processes to terminate. If the TERM signal fails to kill a process, signal 9 (KILL) should always do the job: `kill -9 PID`.
`sync`	Updates (synchronize) the system disk for a consistent file system. The `update` daemon does a periodic `sync`.
`prtvtoc /dev/rdsk/#`	Displays information on format and partitions of disk #. To identify #, use the `df -k` command (see above).
`date`	Displays system date and time.
`who`	Displays a list of users currently logged in.
`who am i`	Displays current user name, the terminal, and the login time.
`hostname`	Displays the name of the current host.
`uname -n`	Displays the name of the current user.
`uname -a`	Displays all information about the current system.
`id -a`	Displays user and group ID.
`compress` *`file`*	Compresses a (plain) file.
`uncompress` *`file`*	Uncompresses a compressed file.

7.4 Interactive Work with the C Shell

Commands in bold are important for standard UNIX users; the other commands are often found in shell scripts or mainly important for UNIX administrators. Refer to Sun manuals for complete information on UNIX commands.

`alias`	Displays currently defined aliases.
`alias lf ls -F`	Makes the command `lf` an alias of `ls -F`.
`unalias lf`	Removes alias so that alias `lf` no longer valid.
`history`	Displays last command lines (recommended alias: `h`)
`!!`	Reexecutes last command line.
`!!`***`str`***	Reexecutes last command line with string *`str`* added.
`!`***`n`***	Reexecutes command line number *`n`*.
`!-`***`n`***	Reexecutes command line *`n`* lines back.
`!`***`str`***	Reexecutes command line that starts with *`str`*.
`!$`	Reexecutes last argument (token) from last command line.
`^`*`str1`*`^`***`str2`***	Replaces *`str1`* by *`str2`* in last command line and reexecutes.
Ctrl-c	Cancels execution of the foreground command.
Ctrl-s	Stops execution of the foreground command (no command entry possible, except after Ctrl-q or Ctrl-c).

Ctrl-q	Resumes stopped or suspended command.
Ctrl-z	Suspends execution of the foreground command (command entry possible).
`bg`	Resumes execution of suspended command in background.
`jobs`	Displays information on background jobs.
`fg`	Brings background process to foreground.

7.5 Shortcuts within Open Look and CDE

Window moving and resizing is accomplished in Open Look with the left mouse button.

- To move a window, grab the window frame at any point and drag it to the new position. If necessary, windows can pass the edge of the screen.
- To resize a window, grab one of the resize corners and move it

For transferring text within or into `vi`, only the Copy and Paste keys (L6 and L8, or the corresponding window menus) can be used.

Within a shell (`csh`) or when working with the Open Look version of `textedit`, you can use the powerful drag-and-drop utility (apart from the Cut, Copy and Paste keys on the keyboard):

1. Highlight the text to be transferred with the left or middle mouse button, and then move the mouse over the highlighted text and press the left mouse button. A small rectangle with an arrow in the upper-left corner appears
2. Move the cursor over to the position where you want to insert that text (when editing) or over the window with the interactive shell into which you want to drop the text.
3. Release the mouse button. The text is transferred.

Within `textedit`, when you release the button over a character, the text is inserted before that character

Drag-and-drop does not work from and into nonscrolling windows (i.e., also scrolling windows after disabling the scrolling) and from and into `vi`. This is a limitation of the current version of Open Look, not a bug. Drag-and-drop also does not work within a remote shell and from and into the VnmrX windows.

With the new UNIX windows environment, CDE (Common Desktop Environment), most of the restrictions in Open Look are overcome. It is a next step towards a windows-driven interface as it has been used on personal computers for many years. Because CDE has many similarities to the personal computers and is easy to explore by trial and error, no further descriptions are given here, except that to copy text within the same window in CDE (e.g.: into the command line), highlight the text that you want to copy, and then press the middle mouse button. This pastes the highlighted text at the current input location (e.g.: the command line).

Chapter 8. Software Operating Levels

Sections in this chapter:

- 8.1 “Solaris Run Levels,” this page
- 8.2 “Login Shell,” page 89
- 8.3 “Windowless Multiuser Mode,” page 90
- 8.4 “Graphical User Interface,” page 90
- 8.5 “Shutting Down the System,” page 93

UNIX is a complex and powerful operating system. The complexity not only is in the large number of files and directories (Chapter 11, “Disk Organization,” covers this topic), but also in the large number of processes involved (many running simultaneously, in a time-shared mode), in the way these processes are started up, and in their interaction and interdependency. In this chapter, we discuss how to control (i.e., switch between) the various operating levels, trying to give you some insight on what happens when Solaris 2.x starts up and shuts down:

8.1 Solaris Run Levels

There are five different run levels in the Solaris environment:

level 0	Runs at the PROM-based software level on Sun workstations.
level 1	Single-user mode, for diagnostics and system maintenance.
level 2	Multiuser mode, but without networking facilities (processes and daemons relating to networking are not started).
level 3	Normal Solaris multiuser operation, including all networking activities as defined in the administration files.
level 4	This is available as an alternate multiuser operating mode; currently unused.

Level 2 did not exist in SunOS 4.x. It was introduced with the transition to Solaris 2.x (i.e., System V UNIX).

Run Level 0—Monitor Mode

The monitor mode (run level 0) is PROM-based software that serves three main purposes:

- Booting UNIX.
- Performing basic hardware diagnostics.
- Setting the boot parameters (default boot device and options), hardware diagnostics parameters (e.g., defining what is tested automatically upon bootup and how), and basic system setup parameters (character screen dimensions, system startup logo, etc.).

The monitor mode is passed quickly when the system automatically boots up after switching on. This mode is reached either by shutting down UNIX (for rebooting or switching off the workstation) or by interrupting UNIX by pressing Stop-a (pressing the Stop key and A key simultaneously). The monitor mode runs in a white ASCII screen, with no windows, and is characterized with the `ok` prompt. The most relevant commands of this mode are shown below (commands considered more important are shown in bold):

`boot`	Boot up to default run level, from the default boot device
`boot -r`	Boot up from default boot device, check what devices and peripherals are present, add device entries to `/dev` where necessary (used after adding peripherals)
`boot -s`	Boot up from default boot device, into single-user mode only
`boot disk`	Boot up from disk (default)
`boot cdrom`	Boot up from CD-ROM
`boot cdrom - browser`	Boot up from CD-ROM and install Solaris using a Web browser (*only for a system with Solaris 2.6 and higher, at least 48 MB of RAM, and a boot disk with at least 1 GB*)
`boot net`	Boot up via Ethernet (diskless systems only)
`go`	Continue after pressing Stop-a
`help`	Show help on the monitor mode
`help diag`	Show help on PROM-based diagnostics
`help setenv`	Show help on the `setenv` monitor command
`printenv`	Display EEPROM settings (you can also get this information in a shell, when running Solaris, using the `eeprom` command).
`reset`	Reset all hardware, self-test, reboot
`setenv <args>`	Change EEPROM settings
`sync`	Sync disk and reboot after Stop-a

Pressing Stop-a should only be used in emergency cases (e.g., when the user interface is hung and cannot be activated by Ctrl-c, Ctrl-d, pressing the Stop key alone, killing windows, or killing processes from an other system using a remote login). If Stop-a was pressed by mistake, you can type `go` in monitor mode to resume the previous activities (you may need to repair the screen using the Refresh option from the background menu).

CAUTION: **If it was really necessary to use Stop-a, it is strongly recommended that you enter the `sync` monitor command to continue. `sync` attempts to synchronize the disk prior to rebooting and may help avoid unnecessary file system inconsistencies and lost or corrupted files.**

Earlier Sun workstations had a different monitor mode that used the "`>`" prompt and had a very terse syntax (most commands were a single character), but essentially offered the same capabilities as the new mode. The main reason Sun created the new monitor mode was to increase safety. In the old mode it was relatively easy to incidentally change EEPROM locations with "arbitrary" keyboard sequences. The new mode uses much longer commands that make it unlikely that commands are typed by mistake.

Current workstations still have a rudimentary subset of the old monitor mode implemented, but we recommend you use the new mode exclusively if possible. If you happen to get the "`>`" prompt, you can type `n` to switch to the new monitor mode. The command `old-mode` is used to change from the new mode to the old mode.

CAUTION: Only if you have reached the monitor mode using the proper procedures (`init 0` or `shutdown`, see below) is it safe to switch off the system using the on/off switch in the back of the CPU. Switching off the system otherwise can corrupt data.

Run Level 1—Single-User Mode

The single-user mode is the proper operating level for checking disk partitions with `fsck`, for dumping or restoring disk partitions without user interaction, etc. In single-user mode, all daemons that could alter disk files are switched off—there is no real multitasking, no swapping, no network activities, no printing, no logins (other than for `root`), and, of course, no acquisition on spectrometer hosts.

As of Solaris 2.x, the `root` password must be entered in order to gain access to the single-user mode[1]. A C shell is opened after logging in, with a hash sign "#" as default prompt. At this point, you can perform system maintenance tasks such as file system checks, backups, and repairs (see Chapter 6, "File Security, Repair, and Archiving," for further information).

Keep in mind that the single-user mode is meant for system administration only. You may experience certain restrictions in this environment (e.g., the command path includes a limited number of directories so that some commands may require an absolute path).

Once you are finished with your system administration task, you should either shut the system down or continue the bootup to multiuser mode:

- Press Ctrl-d to let the system boot up to its default run level (usually 3).
- Enter `init 6` to reboot the system completely (again, up to its default run level)
- Enter `init 0` to shut the system down to monitor mode (e.g., for switching it off)
- When (and only when) instructed by the `fsck` command, you may interrupt the system by pressing Stop-a to bring it down to monitor level. *Use the with caution: pressing Stop-a can corrupt data on a disk.*

Run Level 2—UNIX Without Networking

In this run level, daemons and processes operating networking activities are not started. One can use this run level to operate the system temporarily as a standalone workstation, for system maintenance tasks (such as backing up disk slices on tape) for which remote activities (such as `rsh`, `rcp`, `rlogin`, `ftp`, `telnet`, etc.), and interference by other users should be suppressed, in order to obtain consistent file system dumps.

For spectrometer hosts (at the very least for UNITY*INOVA*, *MERCURY*-Series, and *GEMINI 2000* systems) at this run level, acquisitions are not possible, the same as in single-user mode. Therefore this run level is of little use for most VNMR users and spectrometer system administrators.

---

1. This is an essential security feature. Under SunOS, any user could shut the system down by pressing Stop-a, boot into single-user mode, and the break into the system, remove passwords, etc. On the other hand, you must now not forget the `root` password if you don't want to risk losing system administration access.

Run Level 3—Networking Included

This is the standard, default UNIX and Solaris operating mode; logins by other users are possible both locally as well as remotely. When reaching that run level, a login process is started on the main terminal. With CDE installed, this login shell is normally superseded by an `xdm` graphical login session that gives access to either OpenWindows or CDE.

init and shutdown Commands

The `init` command is used to switch between the various operating levels:

`init 0`	Run system down to monitor mode (run level 0).
`init S` **`init s`**	Switch to single-user mode (run level 1), for diagnostics and system maintenance (`S` and `s` are identical).
`init 2`	Switch to multiuser mode without networking facilities (run level 2).
`init 3`	Switch to multiuser mode (run level 3).
`init 4`	Switch to alternate multiuser mode (run level 4, not implemented).
`init 5`	Run system down completely, power off (where possible).
`init 6`	Run system down to monitor mode, and reboot it to its default run level (usually run level 3).

The most frequently used `init` options are `init 0` to run the system down to monitor mode (`halt` in SunOS 4.x) or `init 5` to for switching it off completely and automatically powering it down where possible (most users just use `init 0`), `init 6` for rebooting the system (`reboot` in SunOS 4.x), and occasionally `init s` for entering the single-user mode.

`init` is also used by the `shutdown` command, which can be used to shut down the system more gracefully to the users. `shutdown` first warns all users about the upcoming system shutdown, then, after a “grace period,” the system is shut down using the `init` command. The default grace period is 60 seconds, and by default, `shutdown` runs the system down to single-user mode

`shutdown`	Run system down to single user mode.
`shutdown -g300`	Same, but with a 5-minute grace period.
`shutdown -g300 "maintenance shutdown"`	Same, with message

With the `-i` option, `shutdown` can be used to reboot the system or to run it down completely:

`shutdown -i6`	Reboot system, with 1-minute grace period.
`shutdown -g300 -i6`	Same, but with a 5-minute grace period.
`shutdown -g300 -i0 "maintenance shutdown"`	Run system down completely, with warning message for users and 5-minute grace period.

On a multiuser system, the command `shutdown -i0` is preferred for shutting down a system. On a system with only a single user, you can also safely use the `init 0` command.

The SunOS 4.x `fasthalt` and `fastboot` commands have been eliminated. They are no longer required, because Solaris 2.x only checks file systems at bootup, when it is really necessary.

How Does the UNIX Bootup and Shutdown Work?

Under SunOS, the system startup was done by a small number of shell scripts: `/etc/rc`, `/etc/rc.boot`, `/etc/rc.local`, `rc.ip`, and `rc.single`. In Solaris 2.x, the task of booting up the system and shutting it down is done with several shell scripts, depending on the target run level. The principal scripts (`rc0`, `rc1`, `rc2`, `rc3`, `rc5`, `rc6`, and `rcS`) are stored in `/sbin`. In order to understand what these shell scripts do, let us look at the central part of `/sbin/rc3`:

```
if [ -d /etc/rc3.d ]; then
    for f in /etc/rc3.d/K*
    {
        if [ -s ${f} ]; then
            case ${f} in
                *.sh)   .        ${f} ;;        # source it
                *)      /sbin/sh ${f} stop ;;   # sub shell
            esac
        fi
    }
    for f in /etc/rc3.d/S*
    {
        if [ -s ${f} ]; then
            case ${f} in
                *.sh)   .        ${f} ;;        # source it
                *)      /sbin/sh ${f} start ;;  # sub shell
            esac
        fi
    }
fi
```

The `rc3` script looks at the files in `/etc/rc3.d` (each of the run levels has its associated directory in `/etc`: `rc0.d`, `rc1.d`, `rc2.d`, `rc3.d`, `rcS.d`). In a first `for` loop, `rc3` looks at files starting with *K*; each of these files is executed in turn, *in alphabetical order.* The way the scripts are called depends on the name of the script. If it has a `.sh` extension, it is simply executed; otherwise, it is called with an argument `stop`.

In a second `for` loop, `rc3` does the same thing with all files starting with an *S* in the same directory, except that entries without `.sh` extension are now called with an argument `start`. `rc3.d` contains just five entries:

```
S15nfs.server
S31smc.init
S31smc.init
S77dmi
S99TAS
```

There are no *K* files, just five *S* (startup) files.

After the first character, which determines whether a script is a `start` or a `kill` script, the name contains a two-digit number, followed by some descriptive part that indicates what that script deals with. The numeric part determines the sequence in which the scripts are called.

Many things in a bootup sequence must happen in a certain order (e.g., certain programs can only be started once some specific other programs have been launched), and that order

is controlled simply by the numeric part of the script name. To change the order of execution, the script must be renamed appropriately.

Obviously, these five scripts are not the entire story for starting up multiuser mode. Prior to executing `rc3`, the script `rc2` is called, which operates on the files in `rc2.d`. It turns out that `rc3.d` is the smallest of the `rc*.d` directories. All the others contain many more files: 19 scripts in `rc0.d` and `rc1.d`, 40 in `rc2.d`, 11 in `rcS.d`. The scripts `rc0`, `rc5`, and `rc6` are all identical and use the files in `rc0.d`.

Which one of the scripts that is called by which `init` option is defined in the file `/etc/inittab`. `init` does more than just to call the `rc*` scripts—see `man inittab` for more information.

Overall, there seems to be a confusing number of shell scripts involved in the system startup and shutdown; however, most of the files in the `rc*.d` directories are just *hard links* to files in a directory `/etc/init.d`. Here is a listing of that directory:

```
ab2mgr                  inetsvc                 RMTMPFILES
acct                    init.dmi                rootusr
ANNOUNCE                init.snmpdx             rpc
asppp                   initpcmcia              rtvc-config
audit                   keymap                  sendmail
autofs                  leoconfig               skipes
autoinstall             lp                      skipkey
buildmnttab             mcd                     spc
buttons_n_dials-setup   mkdtab                  standardmounts
cachefs.daemon          MOUNTFSYS               sysetup
cachefs.root            networker               sysid.net
cacheos                 nfs.client              sysid.sys
cacheos.finish          nfs.server              syslog
cron                    nscd                    TAS
devlinks                pcmcia                  ufs_quota
dhcp                    perf                    utmpd
drvconfig               power                   uucp
dtlogin                 PRESERVE                volmgt
inetinit                README                  xntpd
```

Each of these scripts deals with a specific task in the operating system. Let's just pick one example: `init.d/volmgt` handles the volume management (automatic recognition and mounting of floppy and CD-ROM). There are two extra hard links to this file: `rc0.d/K73volmgt` and `rc2.d/S92volmgt`. Because `K73volmgt` has a name starting with *K*, it is run with the argument `stop`. Whereas the `S92volmgt` script in `rc2.d` has a name starting with *S*, so it is called with the argument `start`. In other words, `init.d/volmgt` is a shell script that starts *and* stops the volume management:

`init.d/volmgt start`	Starts volume management
`init.d/volmgt stop`	Stops volume management

By placing hard links (these could also be symbolic links) with the appropriate names in the specific `rc*.d` directories, we can determine for which run level volume management must be started, where it must be stopped, and where exactly in the bootup sequence.

Similarly (just to give you a second example), the script `init.d/lp` has three extra links: `rc0.d/K20lp`, `rc1.d/K20lp`, `rc2.d/S80lp`. In other words, the printer software

is stopped both for run level 0 and for run level 1 (single-user mode), and started in run level 2 (the first step in starting up multiuser mode or run level 3).

Overall, these shell scripts perform much the same as the `rc*` scripts in SunOS 4.x or BSD UNIX for starting up the operating system. The main difference is that in Solaris 2.x not only the system startup is performed in an orderly and controlled manner, but also the system shutdown is performed in a sequence that avoids unwanted side effects (e.g., by stopping the processes in random or uncontrolled sequence and killing resources that might still be used by some other program to terminate properly).

8.2 Login Shell

Once the multiuser mode is reached, there is a process `ttymon` that watches out for active terminals, primarily the main keyboard and user interface. For each active terminal, `ttymon` starts a `login` process (on serial terminals, a return character must be given to make `ttymon` start the `login` process).

The login process asks for the user name. If a password is defined, it asks for the password, and then compares the user entry with the encrypted password in `/etc/shadow`:

- If the password matches, the login process puts the user in his or her home directory, defined in `/etc/passwd`, and starts up the initial shell, also defined in `/etc/passwd`. The shell is normally a C shell, `/etc/csh`. In this case, *`~username`*`/.cshrc` is executed, as well as *`~username`*`/.login`.
- If the login process does not find the user name in `/etc/passwd`, it asks for a password anyway, to make it more difficult to outsiders to find out defined user names.

In Solaris 2.x, a login process is also started in single-user mode. This blocks a security hole in SunOS 4.x, where a system could simply be aborted and brought up into single-user mode, in which passwords could be removed.

It is strongly recommended to have passwords defined for every user, particularly in networked or open-access situations. A good password should not just contain alphabetic characters:

- It should be mixed (lower and upper) case and include numeric characters.
- The length should be 7 or 8 characters (extra characters beyond that are disregarded).
- It should not be a birthday or a telephone number, and should not be found in any dictionary for any language.
- It should be changed at regular intervals.

Solaris checks the password when it is changed by the user and refuses to accept simple character permutations (reshuffling) of the old password.

Primarily, the login process is started in a white, non-graphic screen. If CDE is loaded, that login screen is superseded by `dtlogin`, a graphical login screen. This not only looks nicer than the traditional text-based login, it also gives the user a choice:

- Log into the CDE environment, see “Common Desktop Environment,” page 90.
- Log into a “fail-safe environment” (a single `xterm` window that cannot be used for running VNMR).
- Select “Command Line Login”, the window-less (white, non-graphic) multiuser mode, see “Windowless Multiuser Mode,” next.

8.3 Windowless Multiuser Mode

On systems without CDE, the initial environment after the login is the windowless multiuser mode. This offers a single shell interface—typically, a C shell for most users or a Bourne shell for `root`. This mode resembles early, non-graphical UNIX interface. Background processes can be started in order to allow for multiple, simultaneous tasks, but this only makes sense as long as the background processes don't generate substantial amounts of output.

Today, this mode is used for simple system administration tasks or for shutting down the system. Both activities must be done as `root`, and both can also be done in a graphical environment.

For regular users after a login, the shell script `~/.login` is executed, which automatically starts the OpenWindows graphical user interface. The windowless multiuser mode is only reached again after quitting the OpenWindows interface. If the `~/.login` script is modified accordingly, it may even log the user out again automatically, such that the windowless multiuser mode never is active for these users. In order to do this, add the three lines

```
clear          # get rid of cursor rectangle
echo 'Automatically logging out ... '
logout
```

after the line

```
openwin -noauth
```

in the `~/.login` script.

If CDE is loaded, the windowless multiuser login becomes one of the startup options. If selected on a system running VNMR, this option causes the `~/.login` script to be executed, which then automatically starts up OpenWindows.

8.4 Graphical User Interface

All modern desktop computers and graphics workstations use a graphical user interface (GUI). Solaris 7 and higher use the CDE (Common Desktop Environment) X-based graphical user interface.

The GUI is the main platform for user interaction with UNIX. Because the GUI features multiple windows, it permits running many (dozens, at least) of applications at the same time and on the same user interface. This is made possible both by the UNIX multitasking and time sharing capabilities, as well as by the high-performance CPUs used in modern workstations.

The graphical user interface primarily is a software layer, on top of which many processes can run in parallel (i.e., at the same level). In addition to that, new processes and graphical utilities can also be started *within* any of the utilities (primarily shells) that run on the GUI layer. This also allows building *hierarchical* process families on top of the GUI layer. VNMR is one example of software that forms its own hierarchy of processes, see "VNMR/VnmrJ," page 92.

Common Desktop Environment

The Common Desktop Environment (CDE) is a standardized graphical user interface that was defined and adopted by Sun, HP, IBM, and Novell. CDE is simple and intuitive to use.

Because it is based on the Motif X Window system standard, it resembles the Windows user interface used on PCs. Although there are fundamental differences between OpenWindows and CDE, for the user the primary difference between the two user interfaces is in the way windows and their utilities (buttons, menus, scroll bars, etc.) look and work.

The primary interaction with CDE is done through the CDE toolbar at the bottom of the screen, as well as through a background menu, similar to OpenWindows. It is simple for the user to configure the CDE environment. Even the CDE toolbar can be reconfigured easily, and just by using the mouse. Similarly, the screen background and the window frame colors can be set by the user, through mouse clicks

There is little if any need to work with text configuration files for CDE (most configuration files reside in `~/.dt`). Therefore, we will not discuss these here.

Unlike OpenWindows, CDE also provides a way to properly terminate running applications when logging out. Whatever utilities are running at logout, they are restarted when entering CDE the next time. As of version 6.1A, VNMR is fully CDE compliant. You can leave VNMR running when exiting CDE, and it will restart, even in the same experiment, when you re-enter CDE.

CDE has a full set of desktop utilities (file manager, calendar manager, clock, performance meter, shell tools, etc.), including a CDE shell tool, called `dtterm`, and a full-screen text editor, `dtpad` (see also “Text Editing,” page 163).

A prominent new feature compared to OpenWindows is that CDE allows switching between any number of desktops. The default is four desktops. The advantage here is that several full-screen applications can be running simultaneously, such as VNMR, Pulsetool, the online manuals, a Web browser, and so on, that would otherwise create considerable window overlap or would require the user to temporarily iconify those applications not needed at the moment.

This last CDE feature may also create problems. Most workstations have 8-bit graphics, which allows for up to 256 colors at any given time. Because of the temptation to open several applications simultaneously, color flashing problems may become more frequent when switching to CDE. This is particularly true for VNMR users, because VNMR uses a fairly large color table. The following hints may help in solving this problem:

- Start VNMR first. This way, it has the correct colors.
- Web browsers are color hogs. Don’t use Web browsers (particularly with color images) extensively while running VNMR, but rather call them on a case-to-case basis (maybe when not temporarily running VNMR).
- Limit the color cube size with Adobe Acrobat Reader for VNMR Online, see “Color Flashing Problem with Adobe Acrobat Reader,” page 214.
- Open the Style Manager from the CDE toolbar, then select “Color”, click on “Number of Colors” and select “More Colors for Applications”.

Of course, you can still interact with UNIX through a shell and command line interface, see also “Shell Tool, Command Tool, Dtterm, and Xterm,” page 91.

Shell Tool, Command Tool, Dtterm, and Xterm

Even with a modern, graphical user interface like CDE users (particularly system administrators) may still want, at least occasionally, to interact with UNIX through a command line interface, i.e., through a UNIX shell (usually a C shell). From within each of these shell interfaces, new processes can be started, including new shell tools, forming a hierarchical process family. Note that if these processes run in foreground, and you end the

parent process (shell) or window (shell tool), you also end the child process (or the entire family of child processes attached to the one you terminate.

Under CDE, most users use dtterm, the proper CDE shell window. However, users of the `vi` editor (see "Text Editing," page 163) may find that (especially when running `vi` remotely, and in particular when running `vi` under CDE, but off a SunOS 4.x system), that the OpenWindows command tool (`cmdtool`) is more compatible with `vi`, see also "Problems with Running vi," page 167).

Shell windows come in different flavors. Each has slightly different properties. Most are scrolling (dtpad, xterm, and command tool), but shell tool is not. They all behave differently at UNIX and text terminals. Therefore, they all require a different setting of the `term` variable (`TERM` in the Bourne shell), as shown in Table 13.

The OpenWindows shell tool and command tool are actually one and the same program. You can toggle between the two by enabling and disabling the scrolling. Even with this, you must still make sure that the `term` variable is set correctly after starting up the tool. Note that particularly when doing a remote login, the `term` variable is usually incorrect (`xterm` instead of `dtterm` or `sun-cmd` for standard scrolling windows). If you want to use `vi` remotely, you should first set this environment variable, e.g.:

```
set term=sun-cmd
```

To quit from a shell window, you terminate the shell that runs in that window by using the command `exit` or by typing Ctrl-d.

`xterm` offers somewhat less comfort than the other shell windows. It is primarily used in an X terminal environment, particularly from PC and Macintosh X servers, see also "Using a Personal Computer as an X Server," page 195.

Table 13. Shell Tools and Their Properties

Tool	*Command Path*	*Setting of `term`*
dtterm	`/usr/dt/bin/dtterm`	`dtterm`
command tool	`/usr/openwin/bin/cmdtool`	`sun-cmd`
shell tool	`/usr/openwin/bin/shelltool`	`sun`
xterm	`/usr/openwin/bin/xterm`	`xterm`

VNMR/VnmrJ

Within the graphical user interface, VnmrJ or VNMR are examples of several active processes, with their own set of windows and graphical utilities. VnmrJ and VNMR are actually entire families of programs and includes several user interfaces. Most of the user interaction with this family of programs is through menu buttons and graphical input, but VNMR and VnmrJ also have their own command line interface and command ("VNMR shell") language, MAGICAL.

In CDE, VnmrJ and VNMR can be started from the CDE toolbar at the bottom of the screen. Unlike OpenWindows, it is possible to exit CDE while VNMR is still running (starting with VNMR 6.1 only). When CDE is started again next time, VNMR will then automatically start up again (it will even rejoin the experiment that was used before logging out).

8.5 Shutting Down the System

At times, it may be necessary to shut down the entire system or the acquisition cabinet.

To Shut Down the Entire System

Shutting down the entire system should not happen frequently, except when the main power has to be switched off for service operations or other reasons. Use the following procedure:

1. Stop acquisition and then exit VnmrJ or VNMR.
2. Exit the windowing environment and log out.
3. Log in as `root`.
4. Shut down the host with the command **`shutdown -i0`**.
5. When the monitor level is reached, shut down external peripherals (disks, tapes, and CD-ROM).
6. Shut down the Sun computer and screen.
7. Shut down the spectrometer console.

To Shut Down Acquisition Computer or Acquisition Cabinet

Shutting down the acquisition computer or the acquisition cabinet may be necessary when boards have to be removed or exchanged from the acquisition computer or when service operations require removing rf boards.

1. Use one of the following procedures to stop the acquisition daemon:
   - If `makesuacqproc` was run during installation, in a UNIX window, enter:
     `su acqproc`
   - If this does not work, open up a UNIX shell window and enter:
     `su -c /vnmr/acqbin/killacqproc -f`
2. If the host computer is to be used for awhile without acquisition, it is best to remove the acquisition flag by entering:
   `rm /etc/acqpresent`

To restart the acquisition computer, you need to get the acquisition daemon running again.

1. Use one of the following procedures:
   - If `makesuacqproc` was run during installation, in a UNIX window, enter:
     `su acqproc`
   - Otherwise, enter:
     `su -c /vnmr/acqbin/startacqproc`
2. If the file `/etc/acqpresent` has been removed, create it again by entering:
   `touch /etc/acqpresent`

Pressing Stop-a (or L1-a on older keyboards) is a hard interrupt or crash. These key combinations should be used only in an emergency and with caution. If you have to use such a hard interrupt, you should afterwards enter `sync` if in new monitor mode or enter `g0` (g-zero) if in the old monitor mode. For details, see “Run Level 0—Monitor Mode,” page 83.

CAUTION: **Never have the acquisition daemon running when the acquisition computer is not properly operating. Never switch off or reset the acquisition computer without killing the acquisition daemon first. If such a situation occurs, the acquisition process can severely block or slow down all system activities.**

Chapter 9. Shell Scripts

Sections in this chapter:

- 9.1 “C Shell Scripts,” this page
- 9.2 “Bourne Shell Scripts,” page 103
- 9.3 “How to Write Shell Scripts,” page 110

Shell scripts are text files containing UNIX commands, mostly interspersed with programming tools (branching, loops etc.), and comment lines. They actually serve as new commands and are executed like commands, by calling them by their name. Solaris 2.x and newer, SunOS 4.x, and most other UNIX implementations have two built-in command interpreters; therefore, two different kinds of scripts exist: Bourne shell scripts and C shell scripts. Each script is covered in this chapter:

9.1 C Shell Scripts

The C shell is the best selection for interactive work (like standard UNIX command entry). It is therefore the default interactive shell. But it is less suitable for elaborate shell script. The programming control structures (e.g., `case` statements) are rather complex, and C shell scripts use more resources and run slower than Bourne shell scripts. For people who know C, however, C shell scripting is probably the easiest to learn, because its syntax is very much like the C programming language (hence its name). For help, enter `man csh` or refer to Sun documentation or any book on BSD 4.2 UNIX.

C Shell Selection, Comments

Irrespective of the calling shell, both command interpreters (Bourne and C shells) are always available (on 4.2BSD UNIX). Therefore, at the beginning of the shell script a selection must be made. The first line in a C shell script must begin with a number, or hash, (#) sign. Also comment lines must start with a number sign.

The most convenient way to write C shell scripts is to start with comment lines (or a descriptive title). This is a good practice and it selects the C shell at the same time. A safer way of defining C shell scripts is to include the following comment line in front of the first command (somewhere in the starting comment, ideally on the first line):

```
#!/bin/csh
```

This calls the C shell command interpreter explicitly. It is possible to speed up the execution with a fast call:

```
#!/bin/csh -f
```

This version does *not* run `~/.cshrc` before starting, which means that the standard aliases are not available. This should be no problem, because shell scripts normally don’t use command aliases.

Set-Variables

Any number of new variables can be generated and initialized with the following statement:
`set name=value`

where *name* is any alphanumeric name. The standard convention is that normal variables are written in lowercase and environment variables in uppercase. The value can be a name (no quotes necessary), a numeric value, a string, or a word list (or name list see below). Any reference to that new variable is made by a dollar-sign notation:
`$name`

Example for the use of a variable:
`echo $name`

A set of standard variables are predefined and globally accessible in the C shell, for example:

`history`	Length of history, e.g., `32`
`prompt`	Prompt for input, e.g., `` `hostname`-`who am i` # ``
`term`	Terminal, e.g., `vt100`
`user`	User name

A list of all so-called *set-variables* can be obtained with the command `set` (no argument).

Environment Variables

Global to both command interpreters is a set of environment variables that are transferred when the two command interpreters call each other. By convention, these variables have uppercase names. A listing of all currently defined environment variables can be obtained with the command `setenv`, for example:

```
VarianNMR:vnmr1 29>setenv

MANPATH=/usr/dt/man:/usr/man:/usr/openwin/share/man

DTDATABASESEARCHPATH=/export/home/vnmr1/.dt/types,/usr/dt/ap
pconfig/types/%L,/usr/dt/appconfig/types/C

DTXSERVERLOCATION=local

LANG=C

HELPPATH=/usr/openwin/lib/locale:/usr/openwin/lib/help

DTSOURCEPROFILE=true

PATH=.:/usr/openwin/bin:/usr/dt/bin:/usr/ccs/bin:/vnmr/gnu/b
in:/vnmr/jre/bin:/vnmr/pgsql/bin:/usr/bin:.:/usr/openwin/bin
:/bin:/usr/ucb:/usr/sbin:/export/home/vnmr1/bin:/etc:/vnmr/b
in

AB_CARDCATALOG=/usr/dt/share/answerbooks/C/ab_cardcatalog

DTUSERSESSION=vnmr1-VarianNMR-0

XMICONBMSEARCHPATH=/export/home/vnmr1/.dt/icons/%B%M.bm:/exp
ort/home/vnmr1/.dt/icons/%B%M.pm:/export/home/vnmr1/.dt/icon
s/%B:/usr/dt/appconfig/icons/%L/%B%M.bm:/usr/dt/appconfig/ic
ons/%L/%B%M.pm:/usr/dt/appconfig/icons/%L/%B:/usr/dt/appconf
ig/icons/C/%B%M.bm:/usr/dt/appconfig/icons/C/%B%M.pm:/usr/dt
/appconfig/icons/C/%B
```

```
DTDEVROOT=

SESSION_SVR=VarianNMR

OPENWINHOME=/usr/openwin

EDITOR=/usr/dt/bin/dtpad

LOGNAME=vnmr1

DTSCREENSAVERLIST=StartDtscreenSwarm StartDtscreenQix
StartDtscreenFlame StartDtscreenHop StartDtscreenImage
StartDtscreenLife     StartDtscreenRotor StartDtscreenPyro
StartDtscreenWorm StartDtscreenBlank

MAIL=/var/mail/vnmr1

USER=vnmr1

DISPLAY=:0.0

SHELL=/bin/csh

DTAPPSEARCHPATH=/export/home/vnmr1/.dt/appmanager:/usr/dt/ap
pconfig/appmanager/%L:/usr/dt/appconfig/appmanager/C

HOME=/export/home/vnmr1

XFILESEARCHPATH=/export/home/vnmr1/%T/%N%S:/vnmr/%T/%N%S:/us
r/openwin/lib/%T/%N%S

XMICONSEARCHPATH=/export/home/vnmr1/.dt/icons/%B%M.pm:/expor
t/home/vnmr1/.dt/icons/%B%M.bm:/export/home/vnmr1/.dt/icons/
%B:/usr/dt/appconfig/icons/%L/%B%M.pm:/usr/dt/appconfig/icon
s/%L/%B%M.bm:/usr/dt/appconfig/icons/%L/%B:/usr/dt/appconfig
/icons/C/%B%M.pm:/usr/dt/appconfig/icons/C/%B%M.bm:/usr/dt/a
ppconfig/icons/C/%B

TERM=dtterm

dtstart_sessionlogfile=/dev/null

TZ=US/Pacific

DTHELPSEARCHPATH=/export/home/vnmr1/.dt/help/vnmr1-VarianNMR
-0/%H:/export/home/vnmr1/.dt/help/vnmr1-VarianNMR-0/%H.sdl:/
export/home/vnmr1/.dt/help/vnmr1-VarianNMR-0/%H.hv:/export/h
ome/vnmr1/.dt/help/%H:/export/home/vnmr1/.dt/help/%H.sdl:/ex
port/home/vnmr1/.dt/help/%H.hv:/usr/dt/appconfig/help/%L/%H:
/usr/dt/appconfig/help/%L/%H.sdl:/usr/dt/appconfig/help/%L/%
H.hv:/usr/dt/appconfig/help/C/%H:/usr/dt/appconfig/help/C/%H
.sdl:/usr/dt/appconfig/help/C/%H.hv

XMBINDDIR=/usr/dt/lib/bindings

vnmruser=/export/home/vnmr1/vnmrsys

vnmrsystem=/vnmr

vnmreditor=vi

memsize=16

vnmrmenu=/vnmr/glide/vnmrmenu

FASTMAP=/export/home/vnmr1/vnmrsys/fastmap
```

```
PGLIB=/vnmr/pgsql/lib
PGDATA=/vnmr/pgsql/data
PGDATABASE=vnmr
PGPORT=5432
PGHOST=localhost
CLASSPATH=/vnmr/jpsg/lib/Jpsg.jar:/vnmr/jpsg/lib:/vnmr/jre
BROWSERDIR=/export/home/vnmr1/vnmrsys/ib_initdir
INITDIR=/export/home/vnmr1/vnmrsys/aip_initdir
AIPDIR=/export/home/vnmr1/vnmrsys/aip_initdir
CSIDIR=/export/home/vnmr1/vnmrsys/csi_initdir
JREHOME=/vnmr/jre
DELEGATE_PATH=/vnmr/user_templates/ImageMagick
LD_LIBRARY_PATH=/vnmr/jre/lib:/usr/openwin/lib:/vnmr/java:/v
nmr/pgsql/lib
GNUDIR=/vnmr/gnu
GCC_EXEC_PREFIX=/vnmr/gnu/lib/gcc-lib/
TCL_LIBRARY=/vnmr/tcl/tcllibrary
TK_LIBRARY=/vnmr/tcl/tklibrary
TIX_LIBRARY=/vnmr/tcl/tixlibrary
BLT_LIBRARY=/vnmr/tcl/bltlibrary
vnmrtext=/vnmr/tcl/bin/dg
graphics=sun
WINDOWID=3670025
TERMINAL_EMULATOR=dtterm
PWD=/export/home/vnmr1
VarianNMR:vnmr1 30>
```

Environment variables can be changed with the same command `setenv`:
`setenv` *`name value`*

For example:
`setenv TERM dtterm`

Arithmetics with Variables

Simple integer math is possible within the C shell. Arithmetic lines must start with an at symbol (@), followed by a space:

```
@ count = 0              Assign 0 to the variable count
@ count = $num + 3
@ count = $num - 3
@ count = $num * 3
```

`@ count = $num / 3`	
`@ count = $num % 3`	Modulo function
`@ count += 2`	count = count + 2
`@ count -= 2`	count = count - 2
`@ count *= 3`	count = count * 3
`@ count /= 3`	count = count / 3 (truncation of result)
`@ count++`	count = count + 1
`@ count--`	count = count - 1

The operators `+`, `-`, `*`, `/`, and `%` can also be combined to form complex mathematical expressions. Multiplication, division, and modulo operators are executed before addition and subtraction. Parentheses can be used to change the precedence.

A number of bit operators (right-shift, left-shift, complement, logical negation, bitwise exclusive and inclusive OR, and bitwise AND) are also available (see C manuals).

Flow Control

A number of C-like flow controls have been built into the C shell to allow for conditional execution and both conditional and unconditional looping. Conditional execution and looping need a logical function that is either true (numeric value 1) or false (numeric value 0). This is either a `test` function (e.g. for the existence of a file) or a logical expression.

`test` Function

The `test` function works like a UNIX command with options and an argument. It can be written in two forms, which are basically equivalent:

```
test -f filename
( -f filename )
```

For reasons of consistency with C, the second form is usually preferred. The example above tests whether the file *`filename`* exists and is not a directory. The following options are possible:

`r`	Read access
`w`	Write access
`x`	Executable
`e`	Existence
`o`	Ownership
`z`	Zero length
`f`	Plain file, not directory
`d`	Directory

Numeric comparison operators are also available:

`==`	Equal to
`!=`	Not equal to
`>`	Greater than
`>=`	Greater than or equal to
`<`	Less than
`<=`	Less than or equal to

For example:

```
( $num == 3 )
```

Also, string comparison is possible (do *not* put the pattern in quotation marks):

`==`	Equal to
`!=`	Not equal to
`=~`	Matches (with wildcard characters)
`!~`	Does not match (with wildcard characters)

For example:

`( $str == xyz )`	True if `$str` is `xyz`
`( $str =~ [abc]* )`	True if `$str` starts with `a`, `b`, or `c`
`( $str !~ *.c )`	False if `$str` ends with `.c`

These logical functions can be combined with logical operators:

`&&`	Logical AND
`||`	Logical OR

For example:

```
(( $num == 5 ) && ( -f filename ))
```

The entire expression must be enclosed in parentheses, and again parentheses can be used to group such functions. The logical AND has precedence over the logical OR (and is therefore evaluated first), but for clarity it is recommended to always use parentheses.

Conditional Execution

Conditional execution is accomplished with the `if` statement:

```
if ( <logical_function> ) then
    <commands>
else
    <commands>
endif
```

Of course, the `else` branch can be left out if it is not required. The `then` must be on the same line as the `if`, and both the `else` and the `endif` should be on separate lines. Multiple `if` statements can also be linked, but in this case there is only one `endif`:

```
if ( <logical_function> ) then
    <commands>
else if ( <logical_function> ) then
    <commands>
else
    <commands>
endif
```

Conditional Looping

Conditional looping is accomplished with the `while` construct:

```
while ( <logical_function> )
    <commands>
end
```

Endless loops can be constructed as follows:

```
while (1)
```

```
    <commands>
end
```

Because true has the numeric value 1, this `while` loop executes forever, unless there is a `break` statement inside the loop (see below), or until an external termination (abort) signal is given.

Unconditional Looping

Unconditional looping is used to work through a number of items in a list, using a list variable, for example:

```
foreach suffix(a b c)
    date > time.$suffix
    sleep 3
end
```

This generates the files `time.a`, `time.b`, and `time.c`. The items in the list can also be numbers (rather "number names"). `suffix` is a list variable as it would be set with the following statement:

```
set suffix=(a b c)
```

The individual elements of such a list variable can be accessed as follows:

```
$suffix[2]
```

The numbering starts at 1. The number of list elements (which is 3 in the above example) is accessible as

```
$#suffix
```

A `break` statement allows jumping to the next command after the `end` from within a `while` or a `foreach` loop:

```
while ( <logical_function> )
    <commands>
if ( <logical_function> ) break
    <commands>
end
<more_commands>
```

Note that there is no `endif` after the `break` command.

`case` Construction

Multiple `if` statements can be avoided with a C-like `case` construction:

```
switch (<string variable>)
    case <value1>:
        <commands>
        breaksw
    case <value2>:
        <commands>
        breaksw
    <more_cases>
    default:
        <commands>
        breaksw
        <commands>
endsw
```

The `default` branch can be left out if it is not required, and also the last `breaksw` is not necessary, but a good practice, because then it isn't left out when a further `case` is added).

The program flow is the same as in C. Whenever the interpreter finds a match, it executes the commands up to the next `breaksw` or `endsw`.

exit Statement

The `exit` statement can be used to quit the execution of a shell script at any point. It is a good practice to return with some status information:

`exit 0`	Exit with good status
`exit 1`	Exit on error (error status)

Arguments

All arguments given to a shell script command are accessible within the script under the list variable `argv`:

`$#argv`	Number of arguments
`$argv[1]`	First argument
`$argv[2]`	Second argument
`...`	
`$argv[$#argv]`	Last argument

The command `shift` can be used to left-shift all arguments (argument 1 is deleted, `$#argv` is decremented by 1). Shift without an argument operates on `argv`; any other list variable can be given as argument.

The simpler argument syntax using `$1`, `$2`, ... and `$#` from the Bourne shell (see the section "Arguments," page 102) has also been made available within the C shell.

Interactive Input

Instead of setting a variable directly, it can be interactively filled with input from standard input (the terminal), for example:

```
echo -n "please enter filename: "
set name = $<
```

This script regards the entire input line as a single word and puts it into the variable `name`. A way around this is the generation of a list variable:

```
echo -n "please enter filenames: "
set line = $<
set names = ($line)
```

or, in a complex script:

```
echo -n "please enter filenames: "
set line = $<
foreach file ($line)
    <commands>
end
```

Another way to generate a list variable is with the `gets` command, which generates it in one step:

```
echo -n "please enter filenames: "
```

```
set names = 'gets'
```

Here Documents

Commands that work with standard input (such as `sed`, `awk`, `grep`, and `ed`) can also take their input from the script itself, using a *here document*. The syntax is as follows:

```
command_name << end_marker
lines of data in the here document
end-marker
```

The command often is followed by arguments. The `<<` notation means that it takes its input from the here document. The *`end_marker`* can be any special character that does not appear in the here document, including `+`, `?`, `EOF`, `!`, and `%`. Here is an example:

```
ed filename <<%
g/string1/s//string2/g
w
%
```

This sequence calls the `ed` editor on the file *`filename`* and substitutes all occurrences of *`string1`* by *`string2`*. If sequences with a dollar sign should not be interpreted as variables in the here document, the dollar sign must be escaped with a backslash:

```
ed filename <<%
g/\$s/s//string/g
w
%
```

If substitution should be avoided in the whole document, the end-marker can be quoted out. This is the safer way to avoid substitution:

```
ed filename <<´%´
g/$s/s//$t/g
w
%
```

9.2 Bourne Shell Scripts

Bourne shell scripts have simpler control structures for branching and looping, and they run faster because they use less system resources. Therefore, most longer shell scripts use the Bourne shell. For help, enter `man sh`, check Sun documentation, or look up the topic in almost any book on UNIX.

Bourne Selection, Comments

At the beginning of the shell script, a selection must be made, the same as for the C shell. The lines before the first command call must *not* begin with a number sign (#). A safer way of defining Bourne shell scripts is to include the following comment line in front of the first command, somewhere in the starting comment, ideally on the first line:

```
#!/bin/sh
```

This calls the Bourne shell command interpreter explicitly and allows starting the shell script with comment. Unlike the C shell, there is no fast call for the Bourne shell.

The number sign is used to write comments into the script. Anything after a # sign (up to the end of the line) is regarded as a comment

Variables

Any number of new name variables can be generated and initialized in a simple way:
name=value

where *name* is any alphanumeric name. The standard convention is that normal variables are written in lowercase and environment variables in uppercase. The value can only be a name (no quotes necessary). Any reference to that new variable is made by dollar sign notation:
$name

An example for the use of a variable is the following:
`echo $name`

There are no real numeric variables; however, variables can be numeric strings.

Environment Variables

C shell environment variables are also known and recognized within a Bourne shell. However, the commands `set` and `setenv` don't exist within the Bourne shell, where such variables are defined just as normal shell variables:
`GNUDIR=$vnmrsystem/gnu`

To make such variables known to subshells (this is what distinguishes environment variables from normal shell variables), you must export them:
`export GNUDIR`

In VNMR, such environment variables are used to transfer variables (information) from the `seqgen` and `psggen` Bourne shell scripts to their `make` utilities.

Arithmetic with Variables

No direct arithmetic operations are supported directly within the Bourne shell, but the `expr` command can be used to do calculations with numeric strings. This command puts the result of its calculation into standard output. Therefore, to set the value of a variable, you use command substitution. The following examples should be self-explanatory:

`val3=`expr $val1 + $val2``	Add numeric (string) values
`val3=`expr $val1 - $val2``	Subtract numeric (string) variables
`val3=`expr $val1 * $val2``	Multiply numeric (string) variables (note that `*` has to be escaped by `\*`)
`val3=`expr $val1 / $val2``	Divide numeric variables
`val3=`expr $val1 % $val2``	Take modulo function of a numeric variable

Flow Control

Various flow controls are built into the Bourne shell to allow for conditional execution and both conditional and unconditional looping. Conditional execution and looping need a test function that is either true (numeric value 1) or false (numeric value 0).

`test` Function

The `test` function works like a UNIX command with options and an argument. It can be written in two forms that are basically equivalent (note that the brackets must be surrounded by spaces):

```
test -f filename
```

or

```
[ -f filename ]
```

Each of these examples test whether the file *filename* exists and is not a directory. The possible options for test are the following:

`r`	Existence and read access
`w`	Existence and write access
`f`	Existence and plain file, not directory
`d`	Existence and directory
`s`	Existence and size greater than zero

The `test` function can also be used to check the size of strings (name variables), for example:

```
test -z $name
```

This example tests whether the string *name* has size zero. The other option is

```
test -n $name
```

This example tests whether the length of the string in the variable *name* is nonzero.

String comparison is also possible:

`test $name1 = $name2`	Check if two strings are equal
`test $name1 != $name2`	Check if two strings are not equal
`test $name1`	Check if a strings is not the null string

For comparison of partial strings, the commands `basename` and `awk` can be useful (see UNIX manuals for more details)—for example:

`base=‘basename $name‘`	Strips off directory part of file name stored in variable `$name`
`base=‘basename $name .c‘`	Strips off directory part of file name stored in variable `$name` and removes suffix `.c` (if present)
`len=‘echo $base | awk ’{print length}’‘`	Sets `$len` to length of string `$base`
`name_end=‘echo $base | awk ’{print substr($0,length-3,4)}’‘`	Sets `$name_end` to the last four characters of string `$base`

Integer (numeric string) comparison is also available:

`test $num1 -eq $num2`	Equal
`test $num1 -ne $num2`	Not equal
`test $num1 -gt $num2`	Greater than
`test $num1 -ge $num2`	Greater than or equal
`test $num1 -lt $num2`	Less than
`test $num1 -le $num2`	Less than or equal

These functions can be combined with logical operators (for more information about this feature see UNIX manuals):

`-a`	Logical AND
`-o`	Logical OR
`!`	Logical negation

Parentheses can be used to group such functions. The logical AND has precedence over the logical OR (and is therefore evaluated first), but for clarity it is recommended to always use parentheses. Note that parentheses have a specific meaning for the shell and must be escaped with a backslash.

For example,

```
if test $# -ge 3 -a -f $1 -a -d $2 -a -d $3; then ...
```

tests whether there were at least three arguments, whether the first argument is a plain file, and whether arguments 2 and 3 are directories. Only if all four conditions are fulfilled, the statements after the `then` are executed.

The `test` command can also be written in a short form. The last example above then reads as follows:

```
if [ $# -ge 3 -a -f $1 -a -d $2 -a -d $3 ] ; then ...
```

Note: You *must* have spaces inside the brackets.

Parentheses are not required in the above examples, because only logical ANDs are used. An example of a combination of OR and AND is the following:

```
if [ $# -ge 3 -a \( -f $1 -o -d $2 -o -d $3 \) ] ; then ...
```

If you have any doubts about logical precedences between OR and AND, it is better to define them explicitly using parentheses!

Conditional Execution

Conditional execution is accomplished with the `if` statement:

```
if <commands1>
then <commands2>
else <commands3>
fi
```

Of course, the `else` branch can be left out if it is not required. Note that `if`, `then`, `else`, `elif`, and `fi` must be on separate lines (or semicolons must be used to mark the lines). It would, therefore, be possible to write the above sequence as follows:

```
if <commands1>; then <commands2>; else <commands3>; fi
```

Multiple `if` statements can also be linked:

```
if <command1>
then <commands2>
elif <command3>
then <commands4>
else <commands5>
fi
```

Note that in this case there is only one `fi`.

There is also a shorthand version for the `if` statement: `&&` works like a logical AND, and `||` is a logical OR that can be used to link commands:

```
<command1> && <command2>
```

means that *command2* is executed only if *command1* does not fail. This is the same as:

```
if <command1>
then <command2>
fi
```

For example,

```
[ -f <file1> ] && [ -f <file2> ]
```

On the other hand,

```
<command1> || <command2>
```

means the *command2* is only executed when *command1* fails.

Looping

Conditional looping is accomplished with the `while` construct:

```
while <commands1>
do <commands2>
done
```

Note again that `while`, `do`, and `done` are on separate lines.

Unconditional looping is used to work through a number of items in a list, for example:

```
for <name> in <value1> <value2> <value3>
do <commands>
done
```

A `for` loop can also be used to scan through the argument variables, see below.

`case` Construction

Multiple `if` statements can be avoided with a `case` construction:

```
case <variable> in
    <pattern1>) <commands1> ;;
    <pattern2>) <commands2> ;;
    <pattern3>) <commands3> ;;
    < ... >
    *) <commands4> ;;
esac
```

The patterns can also contain wildcard characters. Only the commands from the first match to the next double-semicolon (`;;`) are executed. Because the wildcard `*` matches all values, it takes the default cases, and the command therein is only be executed if no other match was found. Multiple patterns can be combined, for example:

```
case <variable> in
    <pattern1>|<pattern2>) <commands1> ;;
    *) <commands2> ;;
esac
```

Special character in a pattern—such as a question mark or parentheses—have to be escaped with the backslash (\).

The `exit` statement can be used to quit the execution of a shell script at any point. It is a good practice to return some status information:

`exit 0`	Exit with good status
`exit 1`	Exit on error (error status)

An `exit` with no argument exits with the status of the last command before the `exit`.

Arguments

The command name, the process-ID, and arguments given to a shell script command are accessible within the script:

`$$`	process-ID of the current shell script
`$#`	Number of arguments
`$0`	name under which the script was called (complete path)
`$1`	First argument
`$2`	Second argument
`...`	

The same as C shells, the command `shift` can be used to left-shift all arguments (argument 1 is deleted, the number of arguments, `$#`, is decremented by 1).

The argument variables can be scanned easily with a `for` loop:

```
for i
do >$i
done
```

This script creates an empty file for each of the input arguments. The notation `>filename` creates or clears (if it already exists) a text file. Another example:

```
for i
do case $i in
    -[abc]) <commands1>;;
    -*)     <commands2>;;
    *.[oc]) <commands3>;;
    *)      <commands4>;;
done
```

In this script:

- *commands1* is executed for the options `-a`, `-b`, and `-c`
- *commands2* is executed for all other options
- *commands3* is executed for arguments that are not options and have a `.o` or `.c` suffix
- *commands4* is executed for the rest of the arguments.

Only in this case, where the `in` part of the `for` construction is left, can the `do` stand on the same line as the `for`:

```
for i do >$i; done
```

If the value of a variable should be built into a string, its name must be separated from the rest of the string, for example:

```
tmp=~/bin/data; cat source > ${tmp}1
```

This puts the output of the `cat` command into a file `~/bin/data1`, whereas

```
cat source > $tmp1
```

puts it into a variable `$tmp1`.

If a variable name has not been set, `$name` or `${name}` represents a null string. The braces also allow you to define default values for the case where a variable is not set.

Interactive Input

Instead of setting a variable directly, it can be filled with interactive input from the terminal (actually from standard input, which can also be a pipe). For example:

```
echo -n "please enter filenames: "
read name1 name2 name3 ...
```

This reads a single line from standard input (normally the keyboard). The words of the input line are assigned to the variables *name1*, *name2*, *name3*, etc. sequentially. The last variable takes all the remaining words of the input line. If there are less words than variables, these variables are be set. Variable defaults can be used to define default input:

`${1-'*'}`	Returns `*` if `$1` is not defined
`${2-$1}`	Returns `$1` if `$2` is not defined
`${1='*'}`	Sets `$1` to `*` if `$1` is not defined, then returns `$1`
`${2=$1}`	Sets `$2` equal to `$1` if `$2` is not defined, then returns `$2`.

Here Documents

The basic syntax of *here documents* for Bourne shell scripts is the same as for the C shell (which was described in “Here Documents,” page 103), except that the way to quote out the end-marker to avoid substitution in the whole document is accomplished with the backslash in the Bourne shell:

```
ed filename <<\%
g/$s/s//$t/g
w
%
```

Procedures

Shell scripts can be called within shell scripts (with arguments, of course), but apart from that, procedures can be defined within a Bourne shell script. The main purpose of using procedures is to simplify shell scripts by reducing repetitive tasks to procedure calls. Similar to many programming languages, such procedures must be defined before they can be called.The following is a simplified example from a Sun installation script:

```
LOGFILE=sample.log
...
stdin_log() {
    read $1 < /dev/tty
    eval echo "\$$1" >> $LOGFILE
}
...
stdin_log reply1
...
stdin_log reply2
...
```

The main task of this procedure is to capture the keyboard input for a log file (with the `read` command, the keyboard input is not part of the standard output and can therefore not be rerouted into a log file). Note that within the procedure `$1` refers to an argument to the procedure, not to the shell script.

9.3 How to Write Shell Scripts

Any method for generating a text file is adequate for creating shell scripts. Scripts should be placed somewhere in the C shell path; otherwise, the script can only be called with a full (absolute or relative) path. The path is normally set in `.login` (or in `.cshrc`) and includes `/bin`, `/usr/bin`, `/home/vnmr/bin`, `~/bin`, and other command directories. Usually scripts are placed in the personal `bin` file `~/bin`, sometimes in the directory in which they are going to be used (especially if the shell script includes relative pathnames). If a shell script should be made available to all users, it might be placed in `/usr/bin` (change the ownership to `root`). If only the NMR users should have access, it can be placed in `/home/vnmr/bin` (change to ownership to `vnmr1`).

Unless a script is generating by duplicating and then modifying an existing shell script, the new file does not have the execution permission flag set. Enter

```
chmod 755 filename
```

to achieve this (read and execution permission for all users, write permission for the owner only). If only people from the own group should execute the script, enter:

```
chmod ug+x filename
```

Still the system will not find this new command (unless the user has logged out and logged in again) because the C shell reads the command directories only when it is started up and does not know about new additions unless it is notified accordingly. The `rehash` command forces the C shell to update its internal command reference list. (This is not necessary when working with the Bourne shell.)

Chapter 10. UNIX System Customization

Sections in this chapter:

- 10.1 “General UNIX Customization,” this page
- 10.2 “X Window System Configuration Files,” page 113
- 10.3 “CDE Customization Files,” page 114
- 10.4 “VNMR and Other Application Configuration Files,” page 114

The entire user interface can be customized by editing a series of files, such as `.cshrc` and `.login`, in each user’s home directory. This chapter describes these files.

VNMR stores a set of default customization files in `/vnmr/user_templates`, from which they are transferred into the various home directories by the `makeuser` command when installing VnmrJ and VNMR or adding new users. Note that most of these customization files have names that start with a dot, hence the name *dot files*. Dot files can not be seen with the `ls` command unless the `-a` or the `-A` option is selected or if you are `root`. For more information on customization files, refer to the Sun manuals.

10.1 General UNIX Customization

The primary dot file for every user is the personal login start-up file `.login`. Apart from that, many UNIX commands and utilities have dedicated customization files, for example, `mail` uses a file `.mailrc` and the C shell (`csh`) uses `.cshrc`. The “`rc`” in some of these file names stand for “remote control”.

.login File

The file `.login` is a shell script that runs automatically after every log in. The main task for `.login` is to define environment variables. There are several UNIX global environment variables, such as `path`, `MANPATH`, `PRINTER`, and `TAPE`. Additionally, there are a number of variables used in connection with VNMR, including `vnmruser`, `vnmrsystem`, `vnmreditor`, `memsize`, `vnmrmenu`, `vnmrtext`, and VnmrJ and VNMR-specific path additions (`/vnmr/bin`, `~/bin`). Additional variables are used for the GNU C compiler (such as `GNUDIR`, `GCC_EXEC_PREFIX`, and GNU path additions) and for the execution of Tcl/Tk utilities (`TCL_LIBRARY` and `TK_LIBRARY`).

Next, the `.login` script (it is a C shell script) determines the environment in which it is run. If it is run on the system console (the nongraphical login screen), it starts up the CDE environment (after setting the `DISPLAY` and `graphics` environment variables and adding `/usr/openwin/bin` and `/usr/dt/bin` to the command path), sets the `graphics` environment variable (used by VnmrJ and VNMR), and the rest of the startup is done by startup scripts specific to the selected environment.

If run via a remote terminal or on a virtual terminal (remote network login), `.login` prompts the user to specify the terminal type:

- `g` for GraphOn terminal
- `d` for a dumb terminal
- `t` for Tektronics terminal (the user is also prompted for the exact terminal type)
- `s` for Sun terminal (typically used with remote logins from a Sun window when no remote X session is to be started)
- `x` for a remote X Window system server.

In the case of the remote X server, the user is also prompted for the host name of the X server (a numeric IP address can be entered instead). With many X servers (such as most UNIX workstations), running remote X displays requires the user to first enter

```
xhost +
```

or

```
xhost remote_host
```

prior to starting the remote login, in order to allow remote systems to display information on the local screen via X. The `xhost +` option gives this permission to any remote host, the `xhost` *`remote_host`* option is specific for one system only.

For remote terminals, `.login` also sets the `term` and `graphics` environment variables that allow VnmrJ and VNMR to run properly on the respective terminals. It is also possible to run VNMR in non-graphical mode if `d` (dumb terminal) is specified. The same can be achieved by entering

```
setenv graphics none
```

prior to starting VNMR using `vn`. This is of interest in the case of very slow connections. All VNMR commands and macros can be run from the command line with no graphical display, but any text output from the text window or from line 3 (and error messages) is shown.

.cshrc File

The file `.cshrc` is for generating command line aliases—such as `h` as the command for `history`, `m` for `more`, `la` for `ls -a`, and `ll` for `ls -al`—and to set certain variables such as the length of the history or the format of the prompt.

The Sun manual *Getting Started with UNIX* lists a large number of items that can be implemented in `.cshrc`, including features that try to increase data safety by enabling the interactive mode for the `rm` and `cp` commands or features that avoid unintended destruction of windows by disabling Ctrl-d. While some of these features may be acceptable for beginners, they can make life hard for others working on the system.

Some alias definitions mentioned in the Sun manuals may even interfere with the execution of certain C shell scripts (note, however, that traditionally most shell scripts are Bourne shell scripts). If you would like to use these features, make sure they are valid in the interactive mode only and not when running C shell scripts. It is perfectly possible to have parts of `.cshrc` only executed during shell scripts and other parts only during interactive mode. See the Sun manuals for more information—the version distributed with the standard VNMR software has a global part only.

C shell scripts can also be written such that `.cshrc` is not called— start them with the line

```
#!/bin/csh -f
```

.exrc File

The file `.exrc` is specific to the `ex` and `vi` file editors and can be used to individually customize for any directory some of the variables for the `vi` text editor, for example, enable the line number display flag. This file is not normally used on VNMR systems.

.profile File

The file `.profile` is specific to the Bourne shell. It is used to customize the Bourne shell environment, which in SunOS is normally used only for shell scripts. This file is not normally used on VNMR systems.

.mailrc File

The file `.mailrc` is a configuration file for the mail software. The file can be generated and modified through the `defaultsedit` tool (SunView) or through a separate, window-based tool started from within Mail Tool in Open Look.

The file `.mailrc` contains lines specific to the `mail` command, to the `mailtool` utility, or to both. It contains a series of set statements and defines variables within `mail` and `mailtool`. Some variables have values—such as bell, flash, and interval—that define the number of beeps and flashes when new mail arrives (very useful when using `mailtool`), and the interval (in seconds) at which mailtool checks for new mail (default is 300 seconds).

If the variable `askcc` is set, `mail` and `mailtool` ask for carbon copy (cc) recipients. If `askbcc` is set, `mailtool` also asks for blind carbon copy (Bcc) recipients. See the Sun manuals for more information about mail software.

.indent.pro File

The `.indent.pro` file is a configuration file (profile) for the `indent` command, which serves to standardize the format of C source files (programs). Refer to the *User Programming* manual for information on C source files. Normally, this file is not placed in the user's home directory, but rather in the directory that contains C source files. `indent`, unfortunately, is not included with standard Solaris 2.x software.

10.2 X Window System Configuration Files

The files `.xinitrc`, `.Xdefaults`, and `.xlogin` are among the dot files used in the X Window system software environment.

.xinitrc File

The file `.xinitrc` is for the Open Look environment only. It installs the Open Look defaults from the `.Xdefaults` file (see below), starts up the Open Look window manager program `olwm`, calls `.openwin-init` (see below), and defines the Open Look screen background (the file `/vnmr/varian.xicon` in standard VNMR).

.Xdefaults File

The file `.Xdefaults` is for the OpenWindows environment only. It contains the default values that define the behavior (outlook, color, fonts, etc.) of the generic X Window system

aspects of the user interface. The contents of `.Xdefaults` are installed through the file `.xinitrc` (see above).

.xlogin File

The `.xlogin` shell script is intended for use (explicit calls) with remote X terminals, in case some of the environment variables required for Open Look have not been defined properly (`OPENWINHOME`, `LD_LIBRARY_PATH`, `path`, etc.). `.xlogin` can often be used in shell tools, instead of logging out and in again. This file is not executable (not is it called automatically). It must be called with `source .xlogin`.

10.3 CDE Customization Files

The CDE customization and definition files are all placed in the local directory `~/.dt`. There is usually no need for the user to modify these files directly. The CDE graphical user interface can be defined, modified, and customized directly and interactively. For further information, consult the Sun documentation or the online *AnswerBook*.

Many CDE applications use a configuration files in `/usr/dt/app-defaults` and `/usr/openwin/app-defaults`. A user can customize CDE environment by editing the file `.dtprofile` located in the user's home directory.

10.4 VNMR and Other Application Configuration Files

In newer VnmrJ and VNMR configuration definitions are located in application-specific files in `/vnmr/app-defaults` (`app-defaults/Vnmr`, `app-defaults/Acqi`, etc.).

To customize any of the applications that have files in `/vnmr/app-defaults`, any user can create a local directory `~/app-defaults` and then have local, customized copies of any or all of the files in `/vnmr/app-defaults`. For other Solaris (CDE) applications, there are similar defaults files in `/usr/dt/app-defaults`.

Most changes in either `.Xdefaults` or `~/app-defaults` concern either colors or fonts. For a list of possible fonts under OpenWindows, use the command

```
xlsfonts | more
```

(don't be scared off by the complex X font names—there are some simpler font names further down the output).

Colors can be specified three different ways:

- Specifying the intensities for red, green and blue as three decimal numbers in the range of `0` to `255`, for example:
- Specifying a six-digit hexadecimal number with two digits (`00` up to `ff`) per color, for example:
- Specifying color names, such as `white`, `black`, `RoyalBlue`, `SteelBlue`, etc. A list of possible color names is found in `/usr/openwin/lib/rgb.txt`.

Chapter 11. Disk Organization

Sections in this chapter:

- 11.1 “Disk Partitioning,” this page
- 11.2 “Solaris Directory Structure,” page 119
- 11.3 “Multiuser Setup for VnmrJ,” page 126

This chapter describes partitioning and the file system organization for system hard disks

11.1 Disk Partitioning

UNIX allows hard disks to be divided into slices (partitions) that have different functions. To a large degree, these disk slices are treated like different hard disk drives—they are a kind of “logical disk drive.” Disk partitioning allows adjusting the size of the various slices to fit user’s needs. This partitioning is to a large extent defined when installing Solaris at the initial system setup, or after adding or reformatting a disk (for more information, refer to “Solaris 9 Installation,” page 17 or “Solaris 8 Installation,” page 29).

Naming Disk Slices

Every disk slice is treated as a separate device. Thus, each of the slices in `/dev` has a device entry. Typical slice names are `/dev/rdsk/c0t3d0s0`, `/dev/rdsk/c0t3d0s6`, and `/dev/dsk/c0t0d0s2`. The subdirectory name indicates whether the disk is to be addressed as character device or raw device (`/dev/rdsk`) or rather as block device (`/dev/dsk`)[1]

The naming of these device entries reflects the hierarchical organization of the SCSI disks, and includes the following numbering scheme:

- *controller (`c`) ID* – 0 is the built-in (SCSI or EIDE) controller, additional (SCSI) controllers (SBus or PCI cards) receive higher IDs (1, 2, 3, etc.).
- *target (`t`) address or ID* – Typical target IDs are 0, 1, 2, and 3 for standard SCSI or SCSI-2 disks. Higher ID numbers can be used if wide SCSI-2 (SCSI-3) or UltraSCSI controllers are used.

---

1. The difference between raw and block devices is in the way a program reads or writes from (or to) these devices. With some devices this is dictated by the properties of the device, but mostly it is just a question of what is best for a program. When addressing such devices, a user must know which option to take (both options are defined for hard disk drives) or can find it out by trial-and-error (fortunately, disk device names rarely need to be specified by the user). Some examples: `fsck` requires a raw device to be specified, but with `mount` a block device must be specified.

- *disk (*`d`*) ID* – In theory, a disk controller can control multiple disks; however, all modern SCSI disks come with a built-in (embedded) disk controller. Therefore, in virtually all cases, the disk-ID is 0, because there is only one disk per controller.
- *Slice (*`s`*) number* – Disks can be split into up to 8 slices, numbered 0 to 7.

The disk or slice addresses seem complicated at first—but in reality it is not difficult to correlate devices and their entries in `/dev`. The main complication is with the internal hard disks, for which the SCSI-ID cannot be seen.

- On SPARCstation 4, SPARCstation 5, SPARCstation 20, and Ultra workstations, the SCSI-ID for internal drives is encoded in the internal SCSI connector.
- On the SPARCstation 4, the internal drive is automatically set to SCSI target 3 (`c0t3d0`).
- On SPARCstation 5, SPARCstation 20, and Ultra workstations, the lower of the internal drives is at SCSI-ID 3 (`c0t3d0`), the upper one at SCSI-ID 2 (`c0t2d0`).

The software driver for the standard SCSI bus is usually configured for four disks at SCSI targets 0, 1, 2, and 3. External disk drives on the built-in, standard SCSI bus on SPARCstation 4, SPARCstation 5, SPARCstation 20, and Ultra 1/140 systems can use SCSI-ID 0 (`c0t0d0`) or SCSI-ID 1 (`c0t1d0`). On a second SCSI bus, you should use SCSI-IDs 0, 1, 2, or 3 for external disk drives (`c1t0d0`, `c1t1d0`, `c1t2d0`, `c1t3d0`). The fast, wide SCSI-2 (SCSI-3) or the UltraSCSI, such as provided with Ultra 1/170E systems with Creator graphics or newer Ultra (Ultra 30, Ultra 60) systems, allows for a total of 16 devices including, of course, more than four disks[2].

Disk slices are composed of a group of successive cylinder groups. A slice can also cover an entire disk. All partitions on Sun disks feature the fast Berkeley file system called ufs (UNIX file system) under Solaris 2.x. The ufs system is a form of disk organization that not only is reportedly faster than the (more fragmented) System V UNIX file system, but it does not lose as much performance when the disk gets nearly full. The Berkeley file system also is much less subject to aging. Typically, a System V file system gets slower over time, because the files tend to be more and more fragmented (many System V UNIX systems have file system maintenance utilities to clean up and defragment disk partitions).

The fast Berkeley file system performs better mainly because:

- The ufs (BSD) file system uses a bigger block size (8 KB as opposed to 0.5 KB with the System V UNIX file system) combined with variable size fragments (4, 2, 1, and 0.5 KB). A typical file structure consists of fewer elements (but still, small files can be stored efficiently using fragments).
- BSD file systems are organized in cylinder groups (see below), and whenever possible, a file consists of data blocks from the same cylinder group. This not only minimizes the movements of the disk heads (minimizing access times), but it prevents (or at least reduces) scattering of the data blocks of a given file over an entire partition (fragmentation, typical for System V file systems when the disk is nearly full).

For obvious reasons, disk slices cannot be overlapping, but there is an exception —while slices 0, 1, and 3 to 7 in a standard partition map are usually set up as a sequence of consecutive partitions of flexible size, slice 2 is usually laid out a single partition that covers the entire disk and overlaps with all other disk slices. This allows the system administrator to either use several slices out of 0, 1, and 3 up to 7 (in which case slice 2 becomes unusable) or (as often done with second, third, etc. disks) define an unpartitioned disk by

[2] There are also SCSI-3 (fast, wide SCSI-2) SBus expansion cards for most Sun systems, and PCI bus UltraSCSI expansion cards for newer, PCI-based Ultra workstations.

using slice 2 without having to alter the partition map (of course, in this case all other slices are unusable)[3].

With standard Solaris 2.x, disk slices cannot be extended over multiple disks. The slice size is limited by the size of the hard disk. However, optional software from Sun called DiskSuite (bundled with the software for bigger server systems) permits having disk slices that extend over several physical disk drives. This software also allows dynamically increasing the size of a file system when more disk space becomes available.

A Typical Solaris System Disk Layout

The disk layout on Solaris 2.x systems depends heavily on the number and the size of the available disks. The user is reasonably free in deciding which parts of the UNIX file system to put on separate slices and on which disk .The entire UNIX file system can be placed on a single, big disk partition (not recommended) or the standard recommendations as provided by the Solaris installation software can be used. The Solaris installation program proposes a layout for the installed disks, but the user can still decide to modify the proposed layout. For instance, it is up to the user to decide whether `/usr/openwin` should be made part of the `/usr` partition or whether it should be a separate disk partition.

The disk slices on a typical Solaris installation include the following file systems:

- Root partition, "/" (typically on `/dev/rdsk/c0t3d0s0`)
- Swap space, combined with `/tmp` (typically on `/dev/rdsk/c0t3d0s1`)
- `/var` file system (typically on `/dev/rdsk/c0t3d0s3`)
- `/opt` file system (typically on `/dev/rdsk/c0t3d0s4`)
- `/usr` file system (typically on `/dev/rdsk/c0t3d0s5`)
- `/export/home` file system (typically on `/dev/rdsk/c0t3d0s6`, sometimes also on a separate disk, such as `/dev/rdsk/c0t2d0s2`)
- Other file systems for extra disks, such as `/data` (e.g., on `/dev/rdsk/c0t2d0s2`)

These file systems are discussed in "Solaris Directory Structure," page 119.

How is a Disk Organized Internally?

Virtually all hard disk drives used today consist of multiple disk platters, typically rotating at 5400 or 7200 rpm (some even faster models are now becoming available). The disk platters are used on both sides. There is a fork-like arm that carries read/write heads for all of the platters (two per platter), and all disk heads are moved in parallel or synchronously.

Different from the old analog audio records, the information is not stored in a single, long, spiral track, but in concentric circles. The circle that is covered by one disk head (on one surface) in one disk turn is called a track. Each track is split into multiple data fields called sectors. Typically, the sector size is 512 bytes. To maximize information density and to distribute the information density more evenly over the disk (the outer tracks are much longer than the innermost ones), some Sun disks define more sectors on the outer tracks than on the inside of the disk.

All tracks that can be accessed simultaneously without moving the heads are called a cylinder[4]. As head movements are mechanical and cost time (seek time, in the order of

3. Before file systems are installed in the various slices, partitioning consists only of a partition map, which was part of the disk header information (disk label) in the first sectors of the disk.

several milliseconds), Solaris tries to write consecutive data into sectors and tracks within the same cylinder. Also, because head movements take longer if they occur between distant cylinders, the Berkeley (ufs) file system tries keeping the information from a single file within groups of adjacent cylinders: ufs disks are organized in cylinder groups. A cylinder group typically is 16 cylinders.

When reading a specific sector on a disk, there is an additional delay from the time it takes until that sector is at the position of the heads. This delay is called rotational latency. Modern drives reduce the rotational latency by using faster rotation speeds. One type of drive, called a zero latency drive, start reading a track immediately after positioning the heads, storing the other sectors of a track in an intermediate buffer. When several consecutive blocks are read from disk, this additional information can be retrieved from a fast RAM buffer.

Zero latency cannot be used for writing to disk. The data for the sector currently under the write head may not be available yet. Instead, Solaris 2.x writes such information into a "virtual file system" (cache file system, `cachefs`) and updates the disk at a later point in time. For the user, writing to the disk this way takes almost no time at all. Also, information that resides in the cache file system can also be retrieved very quickly.

ufs File System

A typical ufs disk layout (header and first disk slice) contains the following elements:

- *Block (sector) 0* – Contains the disk label (disk type and characteristics, such as number of bytes per sector, sectors per track, tracks per cylinder, sectors per cylinder, total number of cylinders, and accessible cylinders) and the partition map (VTOC, unallocated disk space, partition numbers, size in sectors, starting sector, read-only and/or "unmountability" flags, mount points). The contents of the partition map can be read (as `root` only), using, for example:

  ```
  prtvtoc /dev/rdsk/c0t3d0s0
  ```

- *Sectors 1 to 15 on a boot disk* – Contain the boot blocks, a small program that is loaded into RAM at boot time by the firmware (PROM Monitor) and which then loads the kernel into memory. Only a disk with boot blocks can be used as boot disk. The boot blocks are installed (on the first disk only) by the installation software. A second disk could be made bootable (e.g., as a backup boot disk) by manually installing boot blocks using the command `installboot` (see Sun documentation for details).
- *Sector 16* – Contains the first superblock, which includes the fundamental parameters of a file system, such as the number of data blocks, and the number of I-nodes. Because the information in the superblock is so important for any file system, backup superblocks are stored in the first sector of every cylinder group. All superblocks within a disk slice are identical. If the first superblock is corrupted, the file system can be reconstructed using one of the backup superblocks, for example:

  ```
  fsck -F ufs -o b=32 /dev/rdsk/c0t3d0s5
  ```

  By convention, the first backup superblock is always stored at sector 32 of every slice.
- *Sector following every superblock* – Contains a summary block or cylinder group map, containing information about free data blocks and fragmentation in the cylinder group.
- *Sectors following a summary block* – In each cylinder group, the summary block is followed by the I-node table: a database (linked list) of available I-nodes for that cylinder group. The size of the I-node table depends on the number of I-nodes.

---

[4] The first cylinder is usually the most innermost, i.e., slices re filled starting with the inner cylinders.

- *Following sectors in each cylinder group* – Contain I-nodes and data blocks. The number of I-nodes per cylinder group is usually adjusted such that there is one I-node per 2 KB in data blocks (i.e., actual disk space). The number of data blocks in a slice determines the total disk space, the number of I-nodes determines the maximum number of files (plain files, directories, symbolic links). For performance reasons, the usable disk space is usually adjusted to 90% of the total disk space (files filling the last 10% would be too fragmented).

Detailed information on the role of I-nodes and the structure of a UNIX file in the context of I-nodes and data blocks is given in "Hard Links, Symbolic Links: the UNIX File System," page 58.

Superblocks, summary blocks, the I-node library and the I-nodes themselves all take up disk space. Typically, about 6% of the physical space on a disk are used for such file system overhead. In addition to that, Solaris normally only fills a file system to 90% (i.e., if a slice is 90% full, the `df` command reports "100% full"). Overall, as shown in Table 14, this permits using about 85% of the physical disk size for data storage[5].

Table 14. Physical Disk Size Compared to Usable Disk Space

Model	*Size*	*ufs disk space*	*Usable disk space*
535 MB	521640 KB	498389 KB (95.5%)	448550 KB (86.0%)
1.05 GB	1026144 KB	963662 KB (93.9%)	867296 KB (84.5%)
2.1 GB	2077080 KB	1952573 KB (94.0%)	1757316 KB (84.6%)

11.2 Solaris Directory Structure

The next sections discuss what files are stored on the various disk slices. Although for most users it is not necessary to know which file is on what partition, it certainly helps for the system administrator, who is responsible for making sure that disk partitions are not (maybe inadvertently) overfilled. This not only promotes data integrity, but it also helps optimizing performance by avoiding operating with nearly-full file systems.

To see which disk slice your working directory is in, use the `df` command:

```
> df -k .
Filesystem         kbytes  used   avail  capacity Mounted on
/dev/dsk/c0t3d0s6  411582  340552  29880    92%     /usr
```

To find out on which disk slice any other directory resides, enter:

`df -k` *`directory_name`*

[5] It is possible to alter the ufs file system parameters (e.g., using the `tunefs` command or using `newfs` to generate a file system). The available disk space can be increased by reducing the number of I-nodes (but this may limit the number of files that can be stored on a slice) or by reducing the `minfree` parameter, i.e., the percentage of disk space that is usually kept free for performance reasons (the default is 10%, less may slow down the file system when it gets full). In general, altering the file system parameters is not recommended because the gain in disk space is very limited and normally not worth the effort.

Files in the Root Slice

The top of the UNIX file system is root. There is also a primary (ufs) file system mounted on /, typically called the root slice or root partition. When starting up, the boot software first reads the boot block on the first disk sectors, and then it mounts the root slice that contains the UNIX kernel and software needed to continue the boot process. The directories in the root slice are described below. Many directories seem to be part of the root slice, but they actually are mount points for other file systems (disk slices and remote file systems).

`/bin` — Symbolic link to `/usr/bin`.

`/dev` — Pointers to devices, the definitions of the available hardware, including disk slices (e.g., `/dev/dsk/c0t3d0s0`, `/dev/rdsk/c0t3d0s2`), I/O ports (e.g., `/dev/ttya`, `/dev/term/a`), the console (`/dev/console`), pseudo-I/O ports that are used for windows and remote logins (e.g., `/dev/ptyp0`), the screen frame buffer (`/dev/fb`), and so on. To maintain compatibility with older software, many of the SunOS definitions are still present as symbolic links to the corresponding Solaris 2.x definitions (e.g, the SunOS names for disk partitions, such as `/dev/sd0a` and `/dev/rsd1b`. A special entry in `/dev` is `/dev/null`, the "bit bucket", i.e., a port for output that is to be discarded.

`/devices` — The software port names in `/dev` are meant for the user, i.e., they should be both easy to understand and easy to memorize. For the software, however, accessing a port requires considerable extra information, and some of this may even be hardware-specific (i.e., it may depend on the workstation model). The port information for all ports is stored in the file name or path inside `/devices`, and the entries in `/dev` are symbolic links into the corresponding entries in `/devices`. When adding devices (peripherals, expansion boards, etc.) to the workstation configuration, appropriate new entries in `/dev` and `/devices` are generated by the software when booting up with the reconfigure option, i.e., with the `boot -r` command.

Some selected examples:

```
console -> ../devices/pseudo/cn@0:console
dsk/c0t2d0s0 ->
   ../../devices/iommu@0,10000000/sbus@0,10001000/
   espdma@4,8400000/esp@4,8800000/sd@2,0:a
dsk/c0t2d0s1 ->
   ../../devices/iommu@0,10000000/sbus@0,10001000/
   espdma@4,8400000/esp@4,8800000/sd@2,0:b

hme -> ../devices/pseudo/clone@0:hme
ie -> ../devices/pseudo/clone@0:ie
le -> ../devices/pseudo/clone@0:le
log -> ../devices/pseudo/log@0:log
mem -> ../devices/pseudo/mm@0:mem
null -> ../devices/pseudo/mm@0:null

openprom -> ../devices/pseudo/openeepr@0:openprom
pts/0 -> ../../devices/pseudo/pts@0:0
pts/1 -> ../../devices/pseudo/pts@0:1
rdsk/c0t2d0s0 ->
   ../../devices/iommu@0,10000000/sbus@0,10001000/
   espdma@4,8400000/esp@4,8800000/sd@2,0:a,raw

rdsk/c0t2d0s1 ->
   ../../devices/iommu@0,10000000/sbus@0,10001000/
   espdma@4,8400000/esp@4,8800000/sd@2,0:b,raw
term/a -> ../../devices/obio/zs@0,100000:a
term/b -> ../../devices/obio/zs@0,100000:b
ttya -> term/a
ttyb -> term/b
```

	In most cases, the user needs only deal with the simpler entries in `/dev`, but there are exceptions—certain applications reserve a port by changing its permissions and ownership. For example, after setting up a printer on a serial port, the corresponding entry in `/devices` is owned by the user (and the group) `lp` and can only be accessed by that user (the symbolic links in `/dev` all have permission `rwxrwxrwx`). Occasionally, a port may be blocked for no good reason. In this case, the system administrator must fix the permissions and ownership of the corresponding port entries in `/devices`, not in `/dev`.
`/etc`	Important UNIX and network administration files, such as `passwd` and `hosts`, as well as the system start-up scripts in the various `rc` directories (see "Run Level 1—Single-User Mode," page 85). In early SunOS releases, `/etc` used to also contain the system commands required for system administration and start-up. These have been moved into other directories (mostly `/sbin` and `/usr/sbin`), and symbolic links have been placed in `/etc` instead. Important files are stored in subdirectories within `/etc`, such as `/etc/dfs` (file system sharing administration), `/etc/lp` (printer administration files), `/etc/mail` (mail administration files), and others.
`/export`	Disk slices and file systems to be shared with other systems via NFS mounting. On most systems, `/export` contains a single directory, `/export/home`, which is the mount point for the `home` directory slice (see "Home Directory Slice, /export/home," page 124).
`/home`	In SunOS 4.x, the `home` directory slice was mounted on `/home`. In Solaris 2.x, this was changed to `/export/home`, and `/home` was kept as a directory for automounted home directories. *Do not use* `/home` *for standard home directories.* This will not work properly (this would happen in the `root` partition) and will confuse the automounter software. Typically, to improve performance in the VNMR environment, `home` directories are local, not automounted.
`/kernel`	Different from SunOS 4.x, where the UNIX kernel (the part of the UNIX operating system always resident in memory) was a single file `/vmunix` that was often regenerated and modified by the user, the Solaris 2.x kernel is now dynamic and usually doesn't need to be modified for the addition of new device drivers. Such drivers can be added dynamically at any time, without even restarting UNIX. The contents of `/kernel` are used to build a kernel (rarely an issue). The actual kernel is stored as `/platform/<arch>/kernel/unix`.
`/lib`	Link to `/usr/lib` (directory with libraries and compilation files).
`/lost+found`	Directory that takes up eventual unreferenced I-nodes when fixing the root file system using the `fsck` command.
`/net`	Main automounting directory for file systems other than `home` directories. All accessible exported file systems on accessible remote hosts (as set by `/etc/hosts`) can be accessed as `/net/`*`hostname`*`/`*`filesystem`*. To check if a remote host has exported file systems and what the exported file systems are, enter `cd /net/`*`hostname`*` ls`
`/platform`	Contains the actual UNIX kernel in a subdirectory named after the system architecture (kernels for different architectures can be stored on the same system). On a SPARCstation, the kernel is typically `/platform/sun4m/kernel/unix`.
`/proc`	Pseudo-directory (doesn't take up any disk space) containing a list of active system processes (file names correspond to the process-IDs).

/sbin	Important commands required during startup (before /usr is accessible) and during system recovery. This includes commands such as ifconfig, init, mount, mountall, sh, su, swapadd, sync, umount, umountall, and uname, plus the shell scripts rc0, rc1, rc2, rc3, rc5, rc6, and rcS that execute system start-up and shutdown scripts in /etc/rc*.d. See also "Run Level 1—Single-User Mode," page 85.
/vol	Used by the volume manager (vold) for removable media, such as CD-ROMs and floppy disks.
/tftpboot	Installed on UNITY*INOVA* spectrometer hosts as part of VNMR. It contains the acquisition computer operating system (VxWorks) and the Magnet and Sample Regulation (MSR) board, which is uploaded to these CPUs after rebooting the spectrometer console.
/xfn	Used in connection with the Federated Naming Service (FNS, as proposed by X/Open) for enterprise-wide system administration (used on top of NIS+). Not used in a typical VNMR environment.

Typically, the root slice contains also empty directories as the mount points for ufs mounting of local disk slices (/var, /tmp, /usr, /opt, /export/home, /data, etc.) for removable media, such as the CD-ROM (/cdrom) and floppy disks (/floppy), and for NFS mounting of remote file systems (often containing the file server host name).

/tmp Slice

The /tmp file system serves two purposes: /tmp contains temporary files (buffers for editing and scrolling windows and the like) that *are cleared automatically with every bootup*. Also, unlike SunOS 4.x, /tmp contains the primary swap space. The user cannot see as a directory listing any swap files inside /tmp. Swap usage is only displayed in the output of the swap -s command.

/var Slice

The /var slice contains files that are modified by every user's activities, such as spooling, system logging, and mail. The most important subdirectories include:

/var/adm	UNIX system administration files, statistics files, messages, logs.
/var/cron	Log file for some cron activities (typically uucp-related)
/var/log	Recipient for system log files. Mostly the syslog file and older, backed up versions (syslog.0, syslog.1, etc.) of the same file, which contains a detailed log about sendmail activities.
/var/lost+found	Takes up eventual unreferenced I-nodes when fixing the /var file system using the fsck command.
/var/lp	Logs (mainly error logs) from the printing services (lp).
/var/mail	Mail data for every user.
/var/sadm	Logs about the system software installation and the installation of Solaris patches.
/var/spool	Temporary storage for spooled data during the data transfer, for example, to a printer (/var/spool/lp) or from a mail server (/var/spool/mqueue).
/var/tmp	Under CDE, the user typically doesn't have an open console window. Console messages are stored in temporary log files with unique names such as /var/tmp/wsconAAAa0037E:0.0. It is not unusual to find a large collection of such console log files, using up lots of disk space.

Insufficient Space in /var

Installation of patch clusters requires extra disk space. If don't have at least 36 MB of free space in `/var` (`/var/sadm`), the following error message appears:

```
Insufficient space in /var/sadm/patch to save old files
```

You can handle this problem three ways:

- Use the `-B` option while invoking `patchadd`. This option directs `patchadd` to save the backout data to the user-specified file system.
- Generate additional disk space by deleting files that are not needed.
- Override the saving of the old files by using the `-d` (do not save) option when running `patchadd`.

Also, see the manual page for `patchadd`:

```
man patchadd
```

/opt Slice

The `/opt` disk slice is meant for optional software. The purpose is to separate this software from user data. Software such as Wabi (Sun emulation software for running Microsoft Windows under X), Web browsers, and third-party compilers are typically installed in `/opt`. This directory is usually 10 to 60 MB, often more.

/usr File System

The `/usr` file system contains the bulk of the UNIX software: commands, manuals, run-time library files, and so on. Its size (usually 100 MB to 400 MB or more) mostly depends on what software selections are made during the Solaris installation. This slice is meant to be used solely by UNIX and should not hold other data. During the installation, it may be necessary to leave free space in `/usr` for the installation of additional software that might be loaded later. The files in `/usr` can be regarded as read-only by the normal user because only `root` can modify them—an important safety feature.

The most important directories and files in `/usr` are the following:

`/usr/bin`	Main directory for 400+ UNIX commands.
`/usr/ccs`	Directory with two subdirectories (`/usr/ccs/bin` and `/usr/ccs/lib`) containing commands (such as `as`, `make`, `lex`, `yacc`, and many others) and library files used with compilers and related software. Even though Solaris 2.x does not include a C compiler, the contents of `/usr/ccs` (which are only installed as part of the developer configuration) are required for third-party C compilers, such as GNU C (as used by VNMR).
`/usr/dt`	Directory for the CDE (Common Desktop Environment) software.
`/usr/include`	Directory with (Solaris- and system-specific) `include` files for C or C++ source files (installed as part of developer configuration).
`/usr/lib`	Large directory with object libraries (archives), runtime libraries, templates, and default configuration files for various software.
`/usr /lost+found`	Directory that takes up eventual unreferenced I-nodes when fixing the `/usr` file system using the `fsck` command.
`/usr/openwin`	Directory for the OpenWindows software. Depending on the available disk sizes and the disk partitioning, this may be on a separate disk slice.

/usr/sbin	Directory with system administration commands that are used by the operating systems (i.e., commands and daemons launched automatically when booting up Solaris) or are used only by root.
/usr/share	Directory with data that can easily be shared between different Solaris systems, even if they have different architectures. Includes /usr/share/lib (terminal information database, keyboard layouts, time zone definitions, etc.) and /usr/share/man, the ASCII UNIX manuals (actually nroff source files).
/usr/ucb	Directory with BSD UNIX commands (from SunOS 4.x) that are either not part of SVR4 UNIX or behave differently from their SVR4 counterparts. These files are kept for compatibility reasons only—it is better not to use them under Solaris 2.x (eventually they may no longer be supplied). Related data is stored in /usr/ucblib (runtime and object libraries for compiling and running commands under the BSD compatibility package) and /usr/ucbinclude (include files for C programs that differ between BSD and SVR4). For reasons of compatibility between SunOS 4.x and other UNIX flavors, /usr also contains a number of symbolic links pointing to new file system locations: /usr/5bin (/usr/bin), /usr/man (/usr/share/man), /usr/preserve (/var/preserve), /usr/pub (/usr/share/lib/pub) and others, plus the subdirectories /usr/adm, /usr/mail, /usr/spool, /usr/tmp, which have all been moved to /var.

Home Directory Slice, /export/home

In a standard installation, /export/home contains both the VnmrJ software directory, /export/home/vnmrj, as well as the home directory for every user (except root), such as /export/home/vnmr1, but all these can be installed wherever it is convenient from an operational point-of-view (e.g., on a big disk, such as /data). The VnmrJ software and the home directories can also reside on different slices. When processing large amounts of data, or when acquiring or processing multidimensional spectra, you should preferably place the home directories in a large slice or disk, with plenty of free disk space.

VnmrJ Software Directory

It is often advantageous to store the VnmrJ software directory (50 to 190 MB, depending on the spectrometer type and on the installed software options) in a place where two or more versions can coexist. Because the VnmrJ software directory is accessed either through a symbolic link with /vnmr or through an environment variable vnmrsystem defined for every user, VnmrJ can be stored in directories indicating the software revision, for example, /export/home/vnmrj1.1C, /export/home/vnmrj1.1D.

If the environment variable vnmrsystem points to /vnmr, it is only necessary to change the symbolic link /vnmr to change between different VnmrJ releases, for example:

```
rm /vnmr
ln -s /export/home/vnmrj1.1D /vnmr
```

Home Directories for Users

Home directories for VnmrJ users can contain numerous files:

- Files for customization and setup of the personal user interface, such as the directories ib_initdir (setup files for the image browser), app-defaults (customization files for VNMR-related programs and possibly also other applications), and .dt (setup files for CDE).

- Any number of personal data files and subdirectories.
- Optionally a local `bin` directory (e.g., `/export/home/vnmr1/bin`) for personal executables (compiled programs, shell scripts).
- Local `vnmrsys` subdirectory (one for each user), containing local, personal VnmrJ files (macros, pulse sequences and related files, menus, experiments, global parameters, etc.).

The `vnmrsys` directory has several subdirectories that also exist in `/vnmr`, such as `help`, `maclib`, `manual`, `menulib`, `parlib`, `psglib`, `seqlib`, `shapelib`, `shims`, and `tablib` (plus some optional subdirectories, such as `psg`). The convention is that the contents of these directories take precedence over files with the same name in a subdirectory of `/vnmr`. This way, every user can customize the VnmrJ software without having to touch the files in `/vnmr`.

vnmrsys Directory

The `vnmrsys` directory also contains the user's VnmrJ workspaces, `exp1` to `exp9`, as well as the user's global parameters (`vnmrsys/global`). These workspaces and the global parameters are standard UNIX files and directory trees; however, for speed reasons (disk access is very slow compared to memory-based operations), VnmrJ keeps the global parameters and the current experiment in memory, rather than working with the disk files.

The current experiment on the disk is only updated when joining a different experiment, upon exiting VnmrJ or when calling the VnmrJ command `flush`. The global parameters are only updated upon exiting VnmrJ or when calling the VnmrJ command `flush`. *Therefore, you must therefore not alter* `vnmrsys/global` *while VnmrJ is running.* Upon terminating, VnmrJ writes over the modified disk file with the version that it kept in memory. *For the same reason, you must not touch any current experiment*.

Experiment Locking

Considerable confusion could result if VnmrJ is running in multiple copies (e.g., via remote X terminals) and if several of these copies would point to the same experiment. To avoid this, VnmrJ uses a special experiment locking scheme.

Whenever VnmrJ is running, it generates an ASCII file `vnmrsys/lock_#.primary` (where # is the number of the current experiment). Therefore, the file `lock_1.primary` indicates that `exp1` is locked[6]. This file contains three items:

- *Lock type* – An integer number (see Table 15) indicating the type of lock.

Table 15. Lock Types

#	*Lock type*	*Remarks*
1	acquisition	`Acqproc`/`Expproc` related
2	background	`vbg`, or `Vnmr -mback`
3	foreground	normal, interactive VNMR operation
4	automation	special locking for automation mode

- *Host name* – The name of the host system on which VnmrJ is running
- *Process-ID* – The ID of the process (Vnmr) that is locking the experiment.

6. In the process of establishing experiment locking, a secondary lock file is generated temporarily.

When joining a new experiment (through `jexp(#)`, `jexp#`, or upon starting up), VnmrJ first checks for the existence of a lock file for the target experiment. If such a file is found, VnmrJ then verifies whether on the indicated host (i.e., locally) there is a process with the process-ID found in the lock file. If the process exists, VnmrJ refuses to join that experiment. If the process is not found, that probably indicates that the copy of VnmrJ that generated the lock file somehow aborted or was terminated irregularly, such that it was not able to remove the lock file (free up the experiment) as normally when terminating via `exit` or when leaving an experiment via `jexp(#)` or `jexp#`.

Normally, the lock file is created and deleted automatically. Eventually, such a file can be left in `vnmrsys` by mistake (perhaps due to a software or computer crash). In most cases, the VnmrJ command `unlock(n)` will remove it. If the lock happens to be an acquisition lock and is present by mistake, the lock file belongs to `root` (which runs `Expproc` or `Acqproc`), and `unlock` refuses to remove the lock file. However, the user can still remove that lock file using the `rm` command[7].

Files on Other Disks or Disk Slices

Systems with two or more disks usually have extra disk slices for data storage (the name `/data` is suggested). On systems with a small first disk and a big second disk, `/export/home` should be installed on the second disk, to allow for more flexibility with big experiments. The remainder of the first disk can then be mounted as `/data`.

Home directories can also be located in directories other than `/export/home`. Certain parts of VnmrJ (or customized macros) may assume that all home directories are in `/export/home`; therefore, a file in `/data` would not be found. This problem is bypassed by creating a symbolic link `/export/home/xx` pointing to `/data/xx`, such that for the software the file still seems to be in `/export/home`. Use the Solaris `admintool` to create new users. This way, it is assured that the password file `/etc/passwd` contains the proper information about the location of the home directory.

11.3 Multiuser Setup for VnmrJ

The directory `/export/home/vnmr` contains all the files that are (and must be) globally available. Because `/export/home` might contain various VnmrJ directories (e.g., `vnmrj1.1D`), to simplify access to the VnmrJ software directory, a symbolic link `/vnmr` is established, pointing to the currently active software directory. Usually, when addressing `/vnmr`, we actually don't mean the symbolic link, but the directory that link points to.

All users can read the data within `/vnmr` and also execute any macros, pulse sequences, shell scripts, etc. within this directory. In the default setup, only the NMR system administrator `vnmr1` can make modifications to files in `/vnmr`.

Every user has and can use their own (local) pulse sequences, macros, menus, parameter sets, shell scripts, etc. Each user can also modify files from `/vnmr` by creating a local copy and modifying that file (which would then be located in his own `psglib`, `maclib`, `menulib`, `seqlib`, `parlib`—all in the user's `vnmrsys` directory or within a local `bin` file for shell scripts).

If the user calls a macro that exists under the same name both in his local `maclib` and in the global `maclib` (`/vnmr/maclib`), the system executes the local macro, i.e., the local directories are looked up first. This is useful feature, but it can potentially create problems

[7] The ability to remove a plain text file is not controlled by the permissions and the ownership of the file itself, but rather by the permissions of the directory in which the file resides.

because the system always uses a local pulse sequence, while the global file (with the same name) may have been improved or upgraded.

The general idea is to keep the system software clean and to modify it as little as possible. If `vnmr1` wants to make changes available to all users, he or she may decide to put something into `/vnmr` on a case-to-case basis (make sure you keep a log of what modifications are done in `/vnmr`—this will be helpful when upgrading to the next VNMR release).

For macros, an additional directory can be searched automatically if this directory is specified in a (global) VnmrJ parameter `maclibpath`. This additional directory would be searched between the local and the VnmrJ `maclib`. This allows other users to also access the `maclib` directory of `vnmr1`, for example.

Chapter 12. Networking via Ethernet

Sections in this chapter:

- 12.1 “Ethernet Configurations,” this page
- 12.2 “Basic Ethernet Network Access,” page 133
- 12.3 “Network Access and Security,” page 136
- 12.4 “Checking the Network,” page 138
- 12.5 “Disabling the Ethernet,” page 139
- 12.6 “Remote Login, Remote Shells, Remote Copy,” page 140
- 12.7 “Remote Printing and Plotting,” page 140
- 12.8 “Advanced Networking: Connecting to the Internet,” page 141
- 12.9 “Mounting File Systems (mount, /etc/dfs/dfstab),” page 142
- 12.10 “Automatic Mounting (/etc/vfstab),” page 146
- 12.11 “Automounter,” page 148
- 12.12 “Using FTP,” page 149
- 12.13 “NIS and NIS+,” page 153
- 12.14 “Heterogeneous Networks,” page 153
- 12.15 “Remote Home Directories,” page 154
- 12.16 “Gateway Machines,” page 155
- 12.17 “Diskless Systems,” page 156

Networking is such a widespread topic with UNIX in general, and especially with Sun computers, that only selected areas are discussed in this chapter. Further details can be found in the sections covering networking in Sun manuals.

12.1 Ethernet Configurations

In the past, an Ethernet network was constructed from 50-ohm cable, which comes in two different qualities: thick Ethernet and Thinnet. Recently, a new standard for low-cost Ethernet connections, twisted-pair Ethernet, has emerged. It provides for Ethernet communication through cheaper cabling and with more flexibility in the network topology.

Standard Ethernet and Thinnet

The basic Ethernet structure consists of a backbone, a linear coaxial cable with 50-ohm impedance that must be terminated with 50-ohm resistors at both ends. Branching in the backbone is strictly forbidden with Ethernet. The maximum length of the backbone depends on the physical layout of the cable.

There are two standards for Ethernet cable: the original, expensive thick cable, which can be extended over 500 m, and the Thinnet or thin Ethernet, which is based on normal 50-ohm BNC cable and can be extended over 180 m. Up to five such cable segments can be linked together with signal repeaters (as recommended by Sun), which results in the maximum total length of 2500 m for thick Ethernet or 900 m for Thinnet, in a single Ethernet network (often called a local-area network, or LAN).

Several such networks can be linked together via gateway stations, which are basically computers with more than one Ethernet interface. Standard Ethernet cable is equipped with high-quality N-type connectors so that shorter pieces can be linked together. In theory, Thinnet and standard Ethernet cables can be linked together, but this is not recommended.

The device that transmits and receives data is called the transceiver (short name for transmitter-receiver). The transceiver consists of the transceiver hardware and of a tap kit that connects the transceiver (normally a small box) with the backbone cable. Note that Varian field service does *not* connect systems into existing networks; this is the responsibility of the user or the network administrator.

Insertion points are marked every 2.5 meters on the standard Ethernet cable. Standard cable lengths should be used for Thinnet. Be sure to use Thinnet cables, not just any BNC cable. Three types of tap kits are in use:

- *N-type inline tap kit* – It has two N-type connectors and can be used as a junction between two standard Ethernet cable segments (not as a repeater). This is the safest connection, but the Ethernet has to be broken for connecting this tap kit, resulting in a temporary communications loss in the entire Ethernet segment.
- *Vampire tap kit* – At one of the insertion marks a hole is drilled into the standard Ethernet cable, which allows connecting a new transceiver without interrupting communication. Special tools are needed.
- *Thin Ethernet or standard BNC tap kit* – Although it has two BNC type connectors, some thin Ethernet tap kits only have a single BNC connector. In such a case, a T junction has to be used immediately at the tap kit (no branching is allowed with Ethernet) and the two segments of the Thinnet must be attached to it. If the tap kit is located at the end of the network, a T junction is still required with connections to the Ethernet, the tap kit and to a 50-ohm resistor. Also here, the backbone has to be broken up for the installation, causing an temporary loss of communication in the entire Ethernet segment, unless there are spare (unused) T-junctions in the Thinnet backbone.

The Ethernet controller is connected to the transceiver via a branch cable, which uses 15-pin connectors and can be up to 50 m in length. A computer connected to Ethernet (computer, controller, transceiver, and tap kit) is also called Ethernet node. Up to 100 nodes are allowed per each 500 m standard Ethernet segment.

Most Sun computers are equipped with an Ethernet controller with branch cable connector, either directly on the CPU board or through a special interface cable., newer models are equipped with a 10/100BaseT connector, see below.

Connecting to the Network

Once the software is installed and set up to run the Ethernet network, the network hardware can be connected. You may perform the installation yourself, but if you have purchased a Point-to-Point Thinnet Network (Varian Part No. 00-992489-00) or an additional Thinnet Node (Varian Part No. 00-992491-00), a Varian service engineer will perform the installation, subject to the following limitations:

- The network consists of Thinnet products listed above.
- The installation takes place at the time of the installation of the spectrometer.

- The computers at both ends of the network were purchased from Varian.
- The cabling can be routed on the floor and does not involve walls or ceilings.
- The connection does not involve connecting to an existing Ethernet network.

Thinnet is a network composed of thin Ethernet coaxial cable (RG/58U). Each end of the cable connects to a transceiver via a BNC T-connector, and the transceiver connects to an Ethernet port on the rear of the computer. A 50-ohm terminator must be placed into any BNC T-connector port not attached to the transceiver or a cable. The configuration of the Point-to-Point Thinnet Network is shown in Figure 9. It consists of two transceivers (called µtransceivers due to their small size), two BNC T-connectors, two 50-ohm terminators, and a length of thin Ethernet.

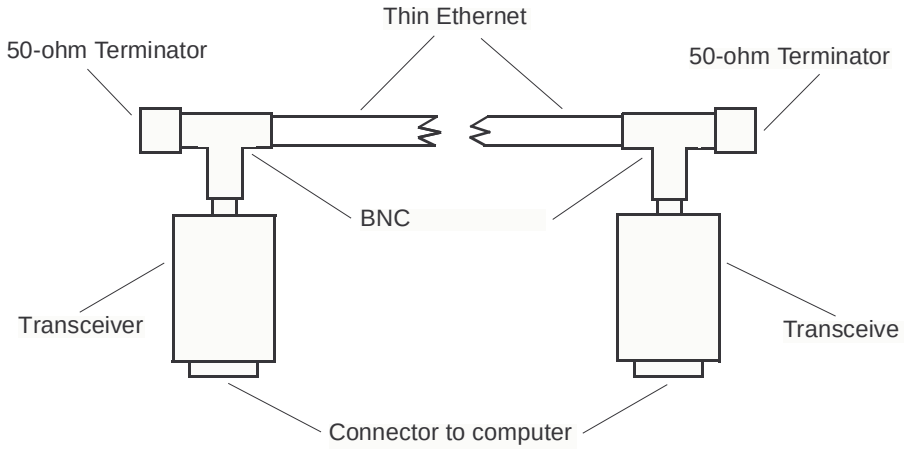

Figure 9. Point-to-Point Thinnet Network

Because of the nature of thick Ethernet cabling and the connections made to it, Varian policy is to not install thick Ethernet. This includes installing the cables, transceivers, and checking the installation. The instrument owner is responsible for correct completion of the installation of a network using thick Ethernet cabling.

If you decide to perform the installation yourself, the following instructions apply.

1. Follow the instructions in the "Shutting Down and Restarting UNIX," page 202 of this manual to shut down the system and power down the workstation.

CAUTION: **Do not install Ethernet cable with power on. Blown fuses can result.**

2. If installing a Thinnet (thin Ethernet) network, connect a transceiver to the Ethernet port on each workstation, attach a BNC T-connector to each transceiver, and connect thin Ethernet cable between the transceivers. Attach a 50-ohm terminator to each BNC connector without a cable.

   If installing thick Ethernet, connect the cable from the workstation to the nearest Ethernet transceiver (from the InLine Tap Kit, Part No. 00-968492-00). Refer to the Sun Microsystems Hardware Installation Manual for the procedure. On some Sun systems, a jumper must be changed.

3. Follow the instructions in the section "Restarting UNIX," page 204 to turn the power on to the workstation and reboot the system.

4. Log in as `root`, enter the `root` password (if implemented), and then confirm network operation running `ping`, where you replace the *`hostname`* argument with the name of your host system:

   `# ping hostname`

   You should get the message below:

   `hostname is alive`

Details of the remote log in operation and other networking features are available by entering `man rlogin` or by referring to the Sun Microsystems documentation.

Disconnecting from the Network

If your system is going to be (or has been) removed from the Ethernet network, enter the `setnoether` command and reboot the system, as follows, before running an acquisition.

1. Log in as `root`, and enter the `root` password (if implemented).
2. Enter the following commands:

   `# cd /vnmr/bin`
   `# ./setnoether`

3. Reboot the system:

   `# init 6`

The `reboot` command does a shutdown and a boot on Solaris.

If Ethernet is disconnected and `setnoether` is not run, the display in Figure 10 may appear if a performance meter is running on the system. This display indicates the CPU is busy looking for a non-existent Ethernet and cannot do acquisitions, lock, or shim.

Figure 10. Performance Meter with Nonexistent Ethernet

Twisted-Pair Ethernet

Twisted-pair Ethernet (TPE) uses a completely different topology All systems connect with a central hub in a star-shaped configuration, and several hubs can be connected again, using twisted pair cabling. Converters allow connecting TPE networks with conventional (BNC or Thick Ethernet) networks. The cabling is TPE networks is done with either shielded or unshielded twisted pair cables with RJ-45 connectors (same as used for telephone connections). In NMR laboratories, shielded cables are strongly recommended. TPE networks operate at 10 Mbps (megabits per second) for a 10-MHz base modulation rate.

Recently, 100-Mbps *fast Ethernet* has become available. Fast Ethernet uses the same TPE cabling standard, but high-quality shielded cables specified for 100 Mbps are highly recommended.

Most or all fast Ethernet ports support *both* 10 Mbps (10baseT) *and* 100 Mbps (100baseT) communication, even on the same network. The Ethernet hardware automatically adjusts the transmission speed to that of the current communication partner. The Ultra 1 models with Creator graphics controller (Ultra 1/170E, Ultra 1/200E) and the newer Ultra (Ultra 5, Ultra 10, Ultra 30, Ultra 60)workstations are equipped with a fast 10/100baseT Ethernet port. For earlier workstations, fast Ethernet ports are available through an SBus expansion board.

12.2 Basic Ethernet Network Access

After computers have been linked via Ethernet, their communication software needs to be activated; otherwise, the systems don't know about Ethernet or other computers. The Ethernet communication is a complex multilayer process. In this manual, we only look at those parts of the software that are essential for the user.

The best and safest way to define the network parameters and set up a workstation for network access is not to start manipulating the network definition files by hand, but to either start entering the proper networking parameters upon loading Solaris or (if Solaris is loaded already) to enter

```
sys-unconfig
```

which removes all local information from the networking definition files in `/etc` and reboots the system. Upon booting, the user is then prompted for the necessary information, such as the local host name, the IP address, and netmask (see Chapter 1, "Solaris 9 Installation," or Chapter 2, "Solaris 8 Installation," and Chapter 4, "Solaris Reference Information,"). Other hosts on the network can then be added using the Solaris AdminTool. However, for debugging purposes it is often useful to be able to check the networking information directly in the administration file. Therefore, we provide this information as if you were modifying these files by hand.

Ethernet Hardware Address

The first thing to consider is that each of the systems must have a unique address for identification in the network. Each Ethernet device has a *hardware address*, a PROM-based sequence of hexadecimal numbers that is unique worldwide (license and distribution by Xerox). A hardware address might be, for example:

```
8:0:20:1:6:4a
```

The user does *not* need to know this address on standard systems (it is read out from the PROM when booting up UNIX), unless the limNET program is being used.

Numeric IP Address

More important than the hardware address is the *Internet address*, which is user-definable. The software looks up the Internet address in an internal library `/etc/hosts`. If the Ethernet is not activated, this file, created by the `suninstall` program, consists of a single line (apart from comments):

```
127.0.0.1      unity localhost loghost
```

This defines `unity` as local host and also as loghost. The number `127.0.0.1` is the default Internet address for systems without active Ethernet. It defines `unity` as host number 1 (0.0.1) on network number 127

The Internet address consists of two parts: the host number and the network number. The address has a total length of 4 bytes (with a value of 0 to 255 each). In a simple network, to communicate, *all hosts communicate must use the same network number, and each host must have a unique host number.*

Three classes of Internet addresses exist:

- *Class A* – The first number between 0 and 127. The last 3 bytes define the host address. This allows defining 128 networks with up to 16,777,214 hosts each (x.0.0.1 – x.255.255.254).

- *Class B* – The first number between 128 and 191. The network and the host address. are defined by 2 bytes. This gives 16,384 network numbers (128.0 – 191.255) with up to 65,534 host numbers each (x.x.0.1 – x.x.255.254).
- *Class C* – The first number between 192 and 255. The network number uses 3 bytes but only 1 byte as the host number, resulting in 4,194,304 different network numbers (192.0.0 – 255.255.255) with up to 254 host numbers each (x.x.x.1 – x.x.x.254).

Do not use host numbers that have either all bits set or all bits unset (0x00 and 0xff, or 0 and 255 for a class C address), because either one of the two (depending on the network protocol) is reserved as the *broadcast address*, through which all hosts on a network can be reached.

For a purely local stand-alone network, the choice of the network class and network number is arbitrary. Most people choose to stay with Sun's default class C network number (192.9.200). The choice of host number is completely up to the local network manager (within the restriction mentioned above).

For a linked (internetworked) network (through gateways or through attachment to the Internet network), the network number cannot be freely selected. It must be ensured that the same network number is used on all systems that should be reached directly (without going through a gateway system). If you are hooking up to an existing network, you are given both the host and the network numbers from the network administrator. If you intend to hook a new network onto existing other networks, you have to talk to the networking facilities manager to obtain a new network number or you should obtain an official network number from your Internet service provider (ISP).

Current production spectrometers (UNITY*INOVA* and *MERCURY*-Vx) the acquisition computer is connected to the host through Ethernet. On this network, network address 10 is used, a class address officially reserved for internal networks. Therefore, do not select this network address on your local network. The term "internal network" implies that this network will never be visible to the public (e.g., there is no Internet packet routing to the acquisition computer). If you have a local network with the network address 10, you can certainly expect problems when connecting to Internet, because you are not using a network address that is officially assigned to your site.

Netmasks, /etc/netmasks

Larger institutions often posses a class B network number for the entire facility, because there may be far more than 255 hosts. For security, however, it may be undesirable to let everybody talk to everybody else throughout the institution. Also, distributing the network over several subnets, instead of using a single large backbone, reduces the network load. For these reasons, the facility manager may decide to divide the facility network into several subnets that behave like different networks. You may receive a 3-byte class B IP address, such as 135.24.68. For you, this is the same as a class C network number, because byte 3 of the address has been declared part of the network number. The mechanism which allows doing this is called a *netmask*.

The netmask defines which bits within an Internet address define the network number, in cases where a higher class network (typically class B) should be split into a series of subnetworks. The default netmask is 255.0.0.0 for class A networks, 255.255.0.0 for class B networks, and 255.255.255.0 for class C networks.

In the example given above, the netmask is probably 255.255.255.0. This is the most common nondefault netmask, although in theory any number of bits in the host part of an Internet address can be declared part of the network number. For example, a small institution with a class C network could create four subnets (up to 62 hosts each) using a

netmask of 255.255.255.192. A network number and the associated netmask are defined in a file `/etc/netmasks` (on systems without subnetting this file can be left untouched). For example, in this `netmasks` file,

```
135.24.6.0    255.255.255.0
```

the network `135.24.6.0` is associated with the netmask `255.255.255.0`:

/etc/hosts File

If you are going public with your network (e.g., through public e-mail access), you need to obtain an IP network address that is *unique worldwide*. In the United Stated, contact a Internet service provider (ISP). For Europe this authority has been delegated to the national branches of EUnet (the European Internet)[1]. You need a good justification (over 255 hosts within 5 years) for a class B network number. Class A addresses are not available for those who don't have one already. It is easier to obtain a class C address.

What does the `/etc/hosts` file look like on a system within a network? Here is an example:

```
192.9.200.1      inova500 loghost
192.9.200.2      unity400
192.9.200.3      inova300
192.9.200.4      sun
127.0.0.1        localhost
```

The list is generally self-explanatory. The address `192.9.200` is the default (class C) network address given by the `suninstall` program. When loading UNIX, this number can be modified. The host number is set on the Workstation page, and the network number on the Defaults page. The `suninstall` program creates a correct file `/etc/hosts`. Other systems, however, have to be added by editing `/etc/hosts` (be sure to make a backup copy of the original file before starting to edit it). Remember, do *not* use the host numbers 0 and 255 for any of the four fields.

Several separated networks (or subnets), even several network classes, can coexist on the same cable segment. Only gateway machines with two Ethernet controllers can access two networks (or more, with more controllers) simultaneously. They also have one host address and one host name per network. The gateway machine is the only possibility for hosts on different networks to talk to each other.

On UNITY*INOVA* spectrometer hosts, `/etc/hosts` also contains entries for the communication with the acquisition console. The network IP address for this internal network (i.e., networks that are not intended for communication with the rest of the world) is 172. These spectrometer entries might have the following addresses

Host	*IP address*	*Notes*
<NAME>	172.16.0.11	
Router	172.16.0.1	router connected to primary ethernet card
INOVA	172.16.0.2	acquisition CPU host name
INOVAAUTO	172.16.0.4	(MSR) board host name
Sun Workstation	172.16.0.11	Router is connected to primary ethernet card

[1] For Germany, the e-mail address is hostmaster@deins.Informatik.Uni-Dortmund.de; for Switzerland, it is hostmaster@eunet.ch.

Host	*IP address*	*Notes*
Slim	172.16.0.43	
LC/MS Workstation	172.16.0.50	

In this example, `wormhole` is the name of the spectrometer host (on the acquisition network only), `inova` is the host name for the acquisition CPU, and `inovaauto` is the host name for the Magnet and Sample Regulation (MSR) board. *This makes it obvious why* `inova` *must not be chosen as hostname on spectrometer hosts.*

On *MERCURY*-Series and *GEMINI 2000* spectrometers, there are two entries for the acquisition in `/etc/hosts`:

```
172.16.0.1      wormhole
172.16.0.21     gemcon
```

/etc/hostname.*, /etc/nodename Files

Apart from `/etc/hosts`, the local host name must also be stored in `/etc/nodename` (this file contains just the local host name, nothing else). The UNIX start-up scripts in `/etc/rc.d` check for a file named `/etc/hostname.*`. On a stand-alone system, this file is named `/etc/hostname.xx0` or `/etc/hostname.lo0` and contains the local host name (same as `/etc/nodename`). If one of these two files is found, the Ethernet hardware is not activated at bootup time.

On networked systems, there is one file `/etc/hostname.*` per network interface (two or more on gateway machines), and it contains the local host name for the corresponding network. The file name extension is the name of the network:

`hostname.le0`	Standard Ethernet, first (standard) port
`hostname.le1`	Standard Ethernet, second port (typically on an SBus card)
`hostname.mce0`	10/100baseT Ethernet, first (standard) port
`hostname.mce1`	10/100baseT Ethernet, second port

The `le` in the above port names stands for Lance Ethernet, named after the manufacturer of the Ethernet interface chip. On some older systems, different (Intel) Ethernet hardware was used, and the corresponding port names then were `ie0`, `ie1`, etc.

12.3 Network Access and Security

On a stand-alone system, the password mechanism provides a reasonable level of security (in connection with the file permissions). In a network, system security becomes more complex:

- It is undesirable to always have to type the password when working across a network (especially in your own account on a remote system).
- Certain remote commands (`rsh` and `rcp`, see "Remote Login, Remote Shells, Remote Copy," page 140) do not work if a password has to be specified.

Global Access Control, /etc/hosts.equiv

The most important file for access control is `/etc/hosts.equiv`. For the above example, a possible `hosts.equiv` file would be the lines:

```
inova500
```

```
unity400
inova300
sun
```

This file is a list of *trusted hosts*, the host names of systems allowed to have direct access, such as via `rlogin` or `rsh`, on that system. If a user (the name, not the ID counts) is defined on both systems and does a remote login into the account with the same name, no password is checked if the remote system is listed in `/etc/hosts.equiv`. Remote shells (`rsh`) and remote copying (`rcp`) are only possible if the user or the host is trusted through `hosts.equiv`. Note that `hosts.equiv` is only valid for users other than `root` (for `root`, the file `/.rhosts` provides the same functionality, see below).

In the example above of `hosts.equiv`, every host name is on a single line. The above form would give any user from the listed hosts remote access (the local host name doesn't need to be listed, but including it simplifies the administration because the same file can be used throughout a network). If the file consists of a single plus (`+`) sign (the default after loading SunOS), every host is trusted. Host names that are preceded by a minus sign (`-`) are explicitly not trusted (users from such a host would have to use passwords when executing a remote login, and remote shells or remote copying is not allowed).

If only the host name is listed in `hosts.equiv`, every user from that host is a trusted user. It is also possible to specifically exclude certain users from a given host from the `rsh` and `rcp` commands (for remote logins, these users always have to use a password):

```
inova500 -vnmr2 -vnmr3
```

We can also deny access to the local system to a specific user from *any* host:

```
+ -vnmr2 -vnmr3
```

On the other hand, it is also possible to further open up the access for a remote user:

```
inova500 vnmr1
```

This allows `vnmr1` from `inova500` to do a remote login into any local account (except for `root`) without having to supply a password. Remote shells are also possible using any account except `root`. For `rcp`, the remote user name determines the local write permission. It is obvious that this feature should be used very restrictively (if at all); otherwise, the system security could be severely affected. The same access can also be granted to a specific user from any host:

```
+ vnmr1
```

We could even open up this facility to *every* user from a remote host (again, except for `root`), but this is certainly most undesirable from a security point of view:

```
inova500 +
```

Individual Access Control, ~/.rhosts

Any user can specify additional hosts and users that are allowed (or denied) logging into the user's account without being asked for a password, overriding or complementing the data in `/etc/hosts.equiv`. This is achieved with a file `.rhosts` in the user's home directory (`~/.rhosts`). The functionality and the structure of this file are identical to the file `/etc/hosts.equiv`, except that if users are specified behind a host name, access is only granted to the local account, not into any account, as with `/etc/hosts.equiv`. It is advisable to only specify users in `~/.rhosts` (of course, it is acceptable to deny access for specific users through `/etc/hosts.equiv`).

For security reasons, `root` is a special case. For `root`, only the file `/.rhosts` is checked, not `/etc/hosts.equiv`. While opening up the access to normal users, `root`

can stay restrictive in the access into its own account. It is crucial for the network security that `/.rhosts` is set up in a very restrictive way.

It is obvious that the files `/etc/hosts.equiv` and `~/.rhosts` are sensitive points with respect to security, especially on larger networks. It is therefore recommended to carefully check entries in `/etc/hosts.equiv`. Even more important than `hosts.equiv` and `.rhosts` is the password protection, of course. Even the most restrictive use of the files mentioned above is worth nothing if `root` (or any other account) does not have a password on a single machine in the network. It is strongly recommended that the network administrator periodically check all accounts for the presence of a password.

A good password is not a word, an abbreviation, or a combination of characters that can be found in any book, newspaper, or dictionary. Users should be strongly discouraged from using a birthday, a name of a friend, or a pet as password, or character sequences that can be easily decoded when they are typed in. We recommend using a combination of alphabetic and numeric characters, some uppercase and some lowercase. Passwords should be at least six characters long (better eight, which is the maximum number of significant characters in a password). Furthermore, urge users not to write down their password and to change their password periodically, like once a month.

12.4 Checking the Network

Once the Ethernet is turned on (see above), the kernel starts looking for the Ethernet hardware, even if there is no communication going on. This provides some basic and continuous system diagnostics.

- Should the branch cable not be connected (transceiver cannot be reached), the following error message appears in the console window:

  ```
  ie0: no carrier
  ```

  or

  ```
  le0: No carrier - transceiver cable problem?
  ```

  Such messages show up every minute or so, or more often if any attempt to communicate via Ethernet is made. This can also indicate that the branch cable is just loose. Unfortunately, the mechanism that fixes a branch cable to the connector does not quite meet the standards of the other Sun or Ethernet hardware. Be careful not to move the cable once it is installed.

- The following message means that the Ethernet hardware has not been activated (no file `/etc/hostname.le0`)

  ```
  network is unreachable
  ```

- The following messages indicate that the Ethernet transceiver is not hooked up to the backbone cable or that the backbone is not terminated correctly.

  ```
  Ethernet jammed
  ```

  or

  ```
  Ethernet cable problem
  ```

The next step is to check Ethernet with real communication programs. The first question is whether the other host up and running, and is its software (also the local files) for Ethernet set up correctly?

The first test is use the `ping` command to see if the two CPUs recognize each other via Ethernet. The usage of `ping` and the reply, if successful, for a remote host named *`otherhost`*:

```
ping otherhost
otherhost is alive
```

In case of problems, the response would be as follows (after several seconds):

```
ping otherhost
no answer from otherhost
```

The next higher level of testing is use the `spray` command to transmit data and to check how successfully and efficiently the data transfer works. Ethernet sends data in packets. If two devices transmit data at the same time, this results in a collision, and the packets will be dropped and transmitted again, which will affect the transfer rate, of course. Entering the `spray` command with a remote host name might give the following response:

```
spray otherhost
sending 1162 packets of lnth 86 to otherhost ...
        no packets dropped by otherhost
        219 packets/sec, 18855 bytes/sec
```

The number of packets and their length can be varied. The maximum length of a packet is 1514 bytes:

```
spray otherhost -c 1000 -l 1514
sending 1000 packets of lnth 1514 to otherhost ...
        72 packets (7.200%) dropped by otherhost
        133 packets/sec, 202448 bytes/sec
```

Larger packets obviously make the communication much more efficient. In practice, however, the standard test (`spray` with only the host name as argument) is much more critical as check on collisions and other communication problems. `spray` can serve as a test for the maximum practically achievable transfer rate at a given packet size.

If `ping` and `spray` are working, this proves that the Ethernet connection is functioning properly, that both hosts are up, their Ethernet enabled (`/etc/hostname.le0`), and that the file `/etc/hosts` is set up correctly.

A final level in the test procedure is to check programs with remote operations. This requires `/etc/hosts.equiv` to be set up correctly. These commands are `rlogin` (remote login), `rsh` (remote shell) and, and `rcp` (remote copy). Each is described in "Remote Login, Remote Shells, Remote Copy," page 140.

The commands `rlogin` and `rsh` do not ask for a password, as long as a host is listed in `/etc/hosts.equiv` on the remote system and as long as the user name is known on both systems. Users from hosts that are not listed in `/etc/hosts.equiv` need to supply a password.

The command `rcp` only works if no password is requested and if write permission exists in the target directory. If `rsh` works without problems and without password, `rcp` should also work without problems. If `vnmr1` has a home directory on both systems, it should be possible for `vnmr1` to copy between the two home directories and to or from any of the `/vnmr` directories. `root` may have a problem copying into a remote directory. Its user-ID on the remote host is reset to –2 (65534), which is equivalent to "nobody," and therefore `root` can only write into remote directories with global write access. In general, it is better to use remote copy to copy files into the local account (if necessary, do a remote login first).

12.5 Disabling the Ethernet

Just unplugging the Ethernet or the branch cable results in messages that show up periodically in the console window. If any attempts to communicate are made, these error

messages accumulate dramatically, which is the least of all possible troubles. In some situations, the computer can be locked up due to continuous attempts to do something over the network. When operating in a nonwindow environment, the error messages actually pop up in the middle of the current command line. When using a full-screen editor like `vi`, it disrupts the screen.

Therefore, it is strongly recommended that Ethernet is properly disabled. This is achieved by renaming the file `/etc/hostname.le0` (or `/etc/hostname.mce0` on some systems) into `/etc/hostname.xx0`. After that, the system must be rebooted completely; otherwise, the Ethernet controller will not be deactivated.

In theory, you could directly deactivate the Ethernet hardware with a command such as `/etc/ifconfig le0 down`, but because rebooting also switches off any daemon that could try to use Ethernet, renaming the file is safer. Consider active disabling only for emergencies such as if the system is flooded with Ethernet hardware error messages.

The next step is to make sure no part of the software tries to use Ethernet. This mostly concerns mounting (see below) and NIS (see "NIS and NIS+," page 153), where additional steps are necessary to shut down these activities. Of course, the same precautions are necessary for other systems trying to access your disk via Ethernet.

12.6 Remote Login, Remote Shells, Remote Copy

This section describes the commands `rlogin`, `rsh`, and `rcp` that are used for remote login, to execute a command on a remote host, and to make a copy remotely.

The `rlogin` command handles remote login on a remote host, for example:

`rlogin rhost`	Remote login on host *`rhost`* under same *`username`*
`rlogin rhost -l username`	Remote login on host *`rhost`* under new *`username`*

The `rsh` command allows executing commands on a remote host, for example:

`rsh rhost command`	Execute command on remote host *`rhost`* under the same user-ID
`rsh rhost -l username command`	Execute *`command`* on remote host *rhost* under a new *`username`*
`rsh rhost`	Same as `rlogin` *`rhost`*

The `rcp` command makes a remote copy. *Do not use it for local copying*. For example:

`rcp source target`	Remote copy, where *`source`* or *`target`* (or both) have form *`rhost:pathname`*
`rcp -r source target`	Recursive remote copy, where *`source`* and *`target`* must be a directories

Each command requires a correct setup of `/etc/hosts.equiv` or `~/.rhosts` (or both) on both systems involved.

12.7 Remote Printing and Plotting

Remote printing and plotting are as easy to set up as local output. Refer to the he manual *VnmrJ Installation and Administration* manual for instructions. Just ensure that the printer and plotter definitions are consistent throughout the network. Even cross-platform printing

and plotting should be fully functional, with few exceptions (we have seen occasional compatibility problems when remotely printing or plotting between SunOS 4.x and Solaris 2.x systems).

12.8 Advanced Networking: Connecting to the Internet

To connect to other networks, a number of additional steps must be undertaken. Besides telling the system in which fully qualified Internet domain it resides, we must establish mechanisms for the system to access IP addresses on other networks through a gateway computer (IP routing), and then we want to enable a mechanism for the system to convert fully qualified Internet Host names into numeric IP addresses (Domain Name Services, DNS).

Defining the Default Gateway (/etc/defaultrouter)

Your network administrator provides information on the Internet router to use. You should store the *numeric IP address* (such as 192.72.123.2) of the router (rather than the host name, even if the router is defined in `/etc/hosts`) in the file `/etc/defaultrouter`. When booting up next time, the system automatically accesses the router whenever you try communicating with an IP address that is not on the local network. You can also directly activate this IP routing using the command

```
route -f add default `cat /etc/defaultrouter` 1
```

Activating DNS (/etc/resolv.conf, /etc/nsswitch.conf)

With the above steps, you can now communicate with any system on Internet, provided you know its numeric IP address. Any host for which you know the IP address can be listed in `/etc/hosts`, and then it can be accessed with the host name rather than the numeric IP address. However, you may not always know the numeric IP address of the system with which you want to communicate. As numbers are difficult to memorize, people prefer addressing remote systems by their Internet host name, such as

```
nmr600.al.abc.corp.com
```

There is no global address book for translating such host names into IP addresses, but there are name servers running specialized software that gathers such information from the Internet. All you need to do for the local host is to tell it which name server to contact for resolving host names not listed in `/etc/hosts`. The primary step for achieving this is, to create a file `/etc/resolv.conf` of the format

```
nameserver   132.190.185.172
nameserver   192.76.144.66
domain ch.corp.com
```

The file contains one or more lines starting with `nameserver`, followed by the IP address of each name server (ask your system administrator about the name servers to use). These name servers are typically *not* on the local network. It is advisable to list the name server that is easiest to access on the first line, the secondary name server on the second line, and so on. Specifying multiple name servers is helpful in case one of these systems is temporarily down, or if one system has problems resolving an address. The last line (starting with `domain`) defines the local Internet domain name (same as in `/etc/defaultdomain`).

This alone doesn't yet activate DNS. The file `/etc/nsswitch.conf` defines where information for the various networking services can be retrieved from. On a stand-alone system (without NIS), this file (without comment lines) looks as follows:

```
passwd:          files
group:           files
hosts:           files
networks:        files
protocols:       files
rpc:             files
ethers:          files
netmasks:        files
bootparams:      files
publickey:       files
netgroup:        files
automount:       files
aliases:         files
services:        files
sendmailvars:    files
```

This means that for all these categories, the information is to be retrieved from the local files in `/etc`. All you need to do now is to change the two lines starting with `hosts:` and `networks:` to

```
hosts:           files dns
networks:        files dns
```

This tells the system to retrieve host and network information using DNS (as configured through `/etc/resolv.conf`). On systems using NIS, `/etc/nisswitch.conf` has a different format.

Of course, apart from systems on your own network, you can still only access systems that are accessible to the public. Some systems, such as most systems at Varian, are hidden behind a security door, typically firewall software, which allows connections from a local system to the outside world, but which blocks off incoming traffic (except perhaps for a few selected protocols, such as simple mail).

12.9 Mounting File Systems (mount, /etc/dfs/dfstab)

The mounting process allows building a file system or a part of a file tree into another file tree. Mounting requires two files—the file system to be mounted (it can be a disk partition or a partial file system such as `/usr/man`) and a mount directory onto which the file system is mounted. This process hides any previous contents of the mount directory, which only can be accessed by unmounting the file first. Therefore, mount directories normally are empty.

Local Mounting

What can mounting be used for? Mounting is already used on every UNIX machine. For example, `/dev/dsk/c0t3d0s5` is mounted onto `/usr`, and `/dev/dsk/c0t3d0s6` is mounted onto `/export/home`. `/usr` and `/export/home` are empty mount directories. These files are mounted as ufs type (UNIX file systems). This type of mounting

is always used for a system's own disks (more exactly, disk slices or file systems). These disk partitions can also be mounted, explicitly, using the `mount` command:

```
mount -F ufs -o rw /dev/dsk/c0t3d0s6 /export/home
```

This would be the full command (it allows for a large number of options). The file type is specified as `ufs`, the partition is mounted with `rw` (read/write) permissions. For local disks, these are the default options, and a simpler command would do as well:

```
mount /dev/dsk/c0t3d0s6 /export/home
```

If this file system is listed in `/etc/vfstab`, which is usually the case (see below), even

```
mount /export/home
```

or

```
mount /dev/dsk/c0t3d0s6
```

is sufficient. Only block devices from `/dev` (not raw devices such as `/dev/rdsk/*`) can be mounted this way.

The `mount` command without arguments displays a list of the mounted file systems, for example:

```
> mount
/ on /dev/dsk/c0t3d0s0 read/write/setuid on Mon Oct 27 13:27:33 1997
/usr on /dev/dsk/c0t3d0s5 read/write/setuid on Mon Oct 27 13:27:33
1997
/proc on /proc read/write/setuid on Mon Oct 27 13:27:33 1997
/var on /dev/dsk/c0t3d0s3 read/write/setuid on Mon Oct 27 13:27:33
1997
/export/home on /dev/dsk/c0t3d0s6 setuid/read/write on Mon Oct 27
13:28:27 1997
/opt on /dev/dsk/c0t3d0s4 setuid/read/write on Mon Oct 27 13:28:27
1997
/disk on /dev/dsk/c0t2d0s2 setuid/read/write on Mon Oct 27 13:28:27
1997
/tmp on swap read/write on Mon Oct 27 13:28:27 1997
```

Unmounting a Local File System

The command `umount` enables the unmounting of mounted file systems, for example:

```
umount /dev/dsk/c0t3d0s6
```

The following simpler version works as well:

```
umount /export/home
```

Because all ufs-type mounted file systems are usually essential for stand-alone operation (and mounting is done automatically at bootup time), the `umount` command is rarely used for local file systems. File systems that are in use, such as the `/` and `/usr` file systems, can not be unmounted. A file system is in use as soon as a user or a shell in general has the working directory within it or as long as a program keeps a file open in that slice.

On desktop SPARCstations, floppies with UNIX file systems can be mounted the same way as hard disk partitions (note that the floppy must be mounted as a partition). Floppy disks with DOS file system require special options to be used with the `mount` command (see also "Using Floppy Disks," page 178).

Remote (NFS) Mounting

With Sun computers (and most other UNIX workstations), the `mount` command has a very important extension: NFS-type mounting of remote file systems. NFS is Sun's Network

File System, a popular mechanism to extend a file system across a network. NFS mounting works very similar to ufs mounting:

```
mount -F nfs -o ro unity500:/export/home /home500
```

This mounts the `/export/home` file system of the host `unity500` in `ro` (read-only) mode onto the local directory `/home500`, which must exist before the `mount` command can be called. In this case, `unity500` is a file server for the local host, which is then called an NFS client machine. Note that NFS mounting only works if a file system is shared by the server system, see "NFS Mounting Options, Sharing a File System," page 144.

There are certain default values. For remote files, NFS-type mounting is default; thus,

```
mount unity500:/export/home /home500
```

mounts the same file with read-write permission (read-write is the default).

Unmounting a Remote File System

Unmounting works the same way as for ufs-mounted file systems:

```
umount unity500:/export/home
```

or simply

```
umount /home500
```

Accessing NFS files is more complex than accessing than local disks because NFS must provide for facilities that can cope with a broken Ethernet, an inactive file server, or a temporarily broken or overloaded Ethernet.

NFS Mounting Options, Sharing a File System

This leads to a number of new options for NFS type of mounting, including:

- Number of mount retries (`retry`)
- Whether or not to retry in foreground (`fg` and `bg`)
- NFS time-out (`timeo`, in tenths of a second)
- Number of retransmissions
- Whether to return an error if the server does not respond (`soft`) or to retry until the server does respond (`hard`)
- Buffer sizes

The defaults are `fg`, `retry=10000`, `timeo=7`, `retrans=3`, and `hard`.

Similar to the commands `rlogin`, `rsh`, and `rcp`, NFS can affect the security of a UNIX system in a network. Therefore, the operating system has a facility allowing specification of the file systems on the server that can be mounted by remote clients. This is achieved with the `/etc/dfs/dfstab` file. This is a simple text file that lists file systems that are shared with (and can therefore be mounted by) other systems. `/etc/dfs/dfstab` contains one line with a share command per shared file system, e.g.:

```
share -F nfs -o ro=unity:mercury /export/home
share -F nfs /data
```

This allows the `/export/home` and `/data` file systems to be mounted remotely. By default, file systems (such as `/data` in the above example) are shared with read-write permission (but the standard permission mechanism can still be used to secure individual files and directories). Also, by default, an shared file system can be mounted by every host on your network, not just in your local subdomain. If you want to share a file system with selected hosts only, you must use the following syntax:

```
share -F nfs -o ro /usr/share
share -F nfs -o ro=mercury /export/home
share -F nfs -o rw=i400:i500:mercury,ro=server /data
```

In this case, `/usr/share` is shared with any system, but is read-only; `/export/home` can be mounted read-only and by the named host only; and `/data` can be mounted read/write, but only by the listed hosts plus read-only by a system named `server`.

Note that options to the `share` command (`rw`, `ro`, etc.) must be separated by a comma, and in options with host name lists, the names must be separated by a colon.

Creating the file `/etc/dfs/dfstab` does not automatically share the file systems. If the file is newly created, the NFS and mount daemons (`nfsd` and `mountd`) are not even running yet. Rebooting the system is the safest and recommended way to achieve this. If you have already shared file systems before and were just adding another entry in `/etc/dfs/dfstab`, you can share the newly added file systems by entering

```
shareall
```

To see which file systems are currently shared on your system, enter `share` without any argument (this information is also found in `/etc/dfs/sharetab`). This does not indicate whether the file systems are actually mounted by any remote system. It is also possible to unshare selected file systems (if they are not currently mounted by any remote system) using the `unshare` command. For details, see `man unshare`.

If mounting doesn't work after entering `shareall` (and `share` indicates that the file system is really shared), most likely the NFS daemon or the `mountd` daemon (or both) are not running. Check the process table using `ps -ef` to see if `/usr/lib/nfs/nfsd` and `/usr/lib/nfs/mountd` are running. When you don't want to reboot the server system but still want to start sharing file systems, you can start the NFS and `mountd` daemons (as `root`, of course) by entering:

```
/usr/lib/nfs/nfsd -a 16
/usr/lib/nfs/mountd
```

Limitations

Note that `mount` makes files from many partitions, disks, and even computers look like a single tree structure. For most commands, this is really the case. For other commands, such as `mv` on directories, do not work between different computers or partitions and always recognize mount points.

With NFS, the user gets easy access to files on other machines, too easy in some ways. A user can move around in foreign directories, execute commands from remote disks, and can copy those onto the local disks. There is no check whatsoever whether compiled software is compatible with the user's hardware. Incompatibilities can lead to error messages and even serious crashes. Unless the server and the client both have the same architecture and operating system, such as two Sun workstations running Solaris 2.x, you need to be quite careful when dealing with compiled programs on remote disks.

NFS Mounting and `root` Access

The user `root` faces another problem. When working on mounted remote file systems, its user-ID is automatically reset to 60001, and its permissions correspond to those of nobody, which only gives access to files with permission for others. This prevents inadvertent file destruction by `root` on remote data systems. On the other hand, it also does not allow `root` to create files on remote disks. `Acqproc` is owned by `root` and, therefore, cannot acquire (write) into experiments located in NFS-mounted file systems. There is a remedy

for this problem, however (giving up on some security features). In /etc/dfs/dfstab you can specify hosts from where root keeps root permission on exported and NFS-mounted file systems:

```
share -F nfs -o root=unity:mercury /data
```

gives root from unity and mercury root access on the remotely mounted /data file system. Therefore, users are allowed to have their experiments or home directories on that remote file system and do acquisitions into these experiments, see "Remote Home Directories," page 154. Note that multiple hosts are separated by a colon; the comma is the separator between multiple arguments.

12.10 Automatic Mounting (/etc/vfstab)

When UNIX is booted up into multiuser mode, the file systems /dev/dsk/c0t3d0s0, /dev/dsk/c0t3d0s5, and others are mounted automatically. On a standard system, this feature is used for all local disk partitions set up by the installation software when loading Solaris. The information as to what partitions need to be mounted is taken from the file /etc/vfstab, which on a stand-alone system with a single disk may consist of the following lines:

```
#device             device               mount          FS      fsck   mount     mount
#to mount           to fsck                             point   pass   at boot   options
#
fd                  -                    /dev/fd        fd      -      no        -
/proc               -                    /proc          proc    -      no        -
/dev/dsk/c0t3d0s1   -                    -              swap    -      no        -
/dev/dsk/c0t3d0s0   /dev/rdsk/c0t3d0s0   /              ufs     1      no        -
/dev/dsk/c0t3d0s5   /dev/rdsk/c0t3d0s5   /usr           ufs     1      no        -
/dev/dsk/c0t3d0s3   /dev/rdsk/c0t3d0s3   /var           ufs     1      no        -
/dev/dsk/c0t3d0s6   /dev/rdsk/c0t3d0s6   /export/home   ufs     2      yes       -
/dev/dsk/c0t3d0s4   /dev/rdsk/c0t3d0s4   /opt           ufs     2      yes       -
/dev/dsk/c0t6d0s0   -                    /cdrom         hsf     -      no        ro
swap                -                    /tmp           tmpfs   -      yes       -
```

The meaning of the individual entries is reasonably obvious:

- Device to be mounted (a block device for disk slices)
- Device address used by the fsck command (a raw device for disk slices),
- Mount point (mount directory)
- Mount type (ufs for local disk slices)
- fsck pass number defining the sequence in which disk slices are checked (a hyphen - means no fsck on that file system)
- Flag (yes or no) that defines whether a slice or file system is to be mounted automatically at boot time
- Options (such as ro for read only, rw for read/write, or simply -) to select the default mount options for the given mount type).

The line starting with fd refers to a floppy drive that requires special mounting (see "Using Floppy Disks," page 178). Also the CD-ROM, /dev/dsk/c0t6d0s0, requires a special mounting mechanism (hsfs for the High-Sierra File System). File systems that are not mounted automatically can be mounted manually, using the mount command. Still it

makes sense to have such entries, because then only the name of the mount directory or of the file system needs to be specified with the `mount` command.

The file `/etc/vfstab` can now be extended to also automatically include NFS mounted files, for example:

```
#device                 device                  mount           FS      fsck    mount     mount
#to mount               to fsck                 point           type    pass    at boot   options
#
fd                      -                       /dev/fd         fd      -       no        -
/proc                   -                       /proc           proc    -       no        -
/dev/dsk/c0t3d0s1       -                       -               swap    -       no        -
/dev/dsk/c0t3d0s0       /dev/rdsk/c0t3d0s0      /               ufs     1       no        -
/dev/dsk/c0t3d0s5       /dev/rdsk/c0t3d0s5      /usr            ufs     1       no        -
/dev/dsk/c0t3d0s3       /dev/rdsk/c0t3d0s3      /var            ufs     1       no        -
/dev/dsk/c0t3d0s6       /dev/rdsk/c0t3d0s6      /export/home    ufs     2       yes       -
/dev/dsk/c0t3d0s4       /dev/rdsk/c0t3d0s4      /opt            ufs     2       yes       -
/dev/dsk/c0t6d0s0       -                       /cdrom          hsfs    -       no        ro
swap                    -                       /tmp            tmpfs   -       yes       -
server:/usr/share/man   -                       /usr/share/man  nfs     -       yes       ro,soft
nmr:/data               -                       /data           fs      -       yes       rw,bg
```

This also mounts the `/usr/man` directory from a remote host server. The UNIX manual is mounted `ro` and `soft`, and `/data` from the same host is mounted as well. The `fsck` pass number is not specified (`-`) because NFS mounted file systems are normally not checked with `fsck`. The `bg` option can be used to avoid that the bootup procedure gets stuck when trying to mount. If a file should temporarily not be mounted, unmount it manually and set the second-to-last item to `no` (no automatic mount at boot time).

To see what file systems are currently mounted (ufs and nfs), use `mount` without arguments. To mount all file systems specified in `/etc/vfstab`, you can use

```
mountall
```

The `mountall` command also tries mounting items such as the floppy disk and CR-ROM (if there are entries in `/etc/vfstab` for these devices). To mount only file systems of a specific type, use

```
mountall -F ufs
```

or

```
mountall -F nfs
```

which only mount the ufs or nfs type file systems from `/etc/vfstab`. This is no problem if files from `/etc/fstab` are mounted already. You don't need to unmount first because `mountall` makes the necessary additions.

The same options are valid for `umount`:

```
umountall
```

tries to unmount all mounted file systems. It does not unmount file systems currently in use, such as if a user currently has his or her working directory in a file system that is mounted. Only certain mount types can be addressed, which usually makes more sense:

```
umountall -F nfs
```

tries to unmount all NFS mounted files.

Finally, here are some hints and useful applications for `mount` and `umount`:

- Before calling `fsck` or `find /` (runs find on all accessible file systems), you don't necessarily want to cover NFS-mounted file systems. You could first unmount all NFS

mounted files. Alternatively, you can use the `prune` option for NFS-type directories, for example:

```
find / -name '*.fid' -print -fstype nfs -prune
```

Note that the `-fstype nfs -prune` options should be specified last.

- If Ethernet or a file server is down while booting up, and you have NFS-type entries in `/etc/vfstab`, the system gets stuck while booting up. Press Ctrl-d to stop the mount process, and the bootup will continue. Log in as `root` afterwards, and mount all NFS entries in `vfstab` with `mountall -F nfs`.

  It is better to use the `soft` option for non-vital file systems like `/usr/share/man`, and the `bg` option for all other file systems that are to be mounted automatically. This allows the bootup process to proceed even in the case of networking problems.

12.11 Automounter

The automounter does more than automatic NFS mounting (standard NFS mounting is already automatic if set up via `/etc/vfstab`). The automounter is mounting on demand. It works with automount directories, such as `/home` or `/net`—special directories with an associated automounter map. As automounting only happens as needed (requested), it is more economical on system and network resources than standard NFS mounting. Also, automounting has been reported as being more robust and user-friendly in the case of temporary network problems.

How it Works: An Example

Your system has an automount directory `/net` (this is standard with Solaris 2.x), and we assume that you have a remote host `i500` on your network that is sharing the file systems `/data` and `/export/home`.

```
> cd /net
> ls
>
```

This indicates that right now there is no automatic mounting to `/net`. Now watch the following steps:

```
> cd /net/i500
> ls
data    export
> ls export
home
>
```

Note that the subdirectory `i500` was not there before we changed directory into it. The point is that mount points inside an automount directory as generated as you need them. The details of the automounting are fixed in an automounter map in `/etc`, see below. In this case, the definition is that shared directories from remote hosts can be accessed as `/net/`*`remotehost`*`/`*`shared_dir`*, which in the above example gives us the new paths `/net/i500/data` and `/net/i500/export/home`.

Again, this is a standard Solaris 2.x feature—all that is needed for this to work is that the remote system shares a file system first. *You can now see why we recommend sharing file systems with specified hosts only.*

If you list the contents of the remote directories, you see these files as if they were local. If your system is not granted access to certain shared file systems (see “NFS Mounting

Options, Sharing a File System," page 144), you still see the subdirectories (`data` and `export/home` in the above example), but these directories would appear to be empty.

The automount subdirectories (mount points) are there for as long as we need them (e.g., as long as we work on files in this temporary file system hierarchy or as long as any user has his or her current working directory somewhere in the temporary directory structure). When these directories and their subfiles are not used or accessed for more than five minutes (default), the automount daemon (`automountd`) automatically unmounts them.

Automount Maps (/etc/auto_master)

The central definition file for the automount daemon is `/etc/auto_master`. The relevant lines from this file are

```
+auto_master
/net     -hosts     -nosuid
/home    auto_home
/xfn     -xfn
```

The first line refers to an external NIS or NIS+ map and is not discussed further here (if an NIS or NIS+ map named `auto_master` exists, its contents are used in place of the above first line). The following lines stand for one automount directory each:

- `/net` automount directory – The `-hosts` option specifies that shared directories from remote NFS servers are to be mounted as described in the previous section. The `-nosuid` option prevents SUID operations (see "Special Permissions," page 56) and is an extra security feature.
- `/home` automount directory – A separate automount map, `/etc/auto_home`, is to be used.
- `/xfn` automount directory – A directory used in connection with the "federated naming system," It will not be discussed here.

The automounter map for `/home`, the file `/etc/auto_home`, in its default form contains only one single line for systems with NIS or NIS+:

```
+auto_home
```

This means that unless we have the corresponding NIS or NIS+ map set up, automounting in `/home` is not activated. If we want to use automounted home directories in `/home`, we need to expand `/etc/auto_home`:

```
+auto_home
user1    server:/export/home/user1
user2    server:/export/home/user2
```

It is advisable to have a local home directory, at least for `vnmr1` (see also "Remote Home Directories," page 154). In the above case (assuming `/export/home` on the server is shared), these directories are already accessible by using a path such as `/net/server/export/home/user1`. The path `/home` just provides for a simpler path than `/net`, at the expense of some extra setup work. You must have consistent user (user- and group-ID) definitions across your network for this to work properly.

12.12 Using FTP

NFS mounting is certainly the most convenient way of accessing remote files or making local files accessible to remote systems, especially if such connections are to be set up permanently. For a certain larger range of hosts, another option is to enable `rcp` for

occasionally transferring files between two workstations. However, both these options require extra administrative overhead and a friendly administrator who sets up the networking parameters and permissions to make `rcp` possible.

In case you can't do `rcp`, you can still use FTP (file transport protocol) to transfer your data to a remote system. For transferring data to and from distant host, and in particular to download software and patches from a public FTP site, you have no other choice than to use FTP.

Using FTP Interactively

FTP can be used several ways: by using the `ftp` command directly or by using a Web browser or other graphical user interfaces, such as `ftptool` (public domain software). Web browsers and `ftptool` offer more comfort and security than using `ftp` directly. In particular, they automatically use the proper transfer modes. But as long as you observe some basic precautions, you can use the simple `ftp` command, even for multiple files.

Here are some tips on using the `ftp` command:

- By default, `ftp` operates in ASCII mode. This mode should only be used for ASCII files (normal text files, README files, pulse sequence source code, etc.). With binary files (compiled programs, compressed and tar files, binary data such as FIDs, etc.) you must use the binary mode; otherwise, the transferred data may be corrupted. Enter `bin` to switch to binary mode. Actually, ASCII files also work with the binary mode, so it is probably a good idea to use binary mode all the time.
- If you don't want to overwrite an existing file, you can specify with the `put` and `get` commands an optional target file name instead of the current working directory:

  `get` *`source_file target_file`*

  `put` *`source_file target_file`*
- For transferring multiple files from the same directory, use the `mget` and `mput` commands with multiple file names or wildcard characters (or both), for example,

  `mget 5.3BacqSOLino103.* 5.3BpsgSOLino101.*`

  With `mget` and `mput`, the target file name is always the same as the source file name.
- The `mget` and `mput` commands ask for a confirmation for every single file that they are about to transfer. This may be useful if a wildcard argument covers files that you don't want to transfer, but mostly it is unwanted and just slows down the transfer, requiring unnecessary user interaction. You can easily switch off (and on) this prompting with the `prompt` command. The interactive prompting can also be suppressed by calling `ftp` with the `-i` option. This option is particularly helpful when using `mget` or `mput` in `ftp` within shell scripts, see "Calling ftp Within Shell Scripts," page 152.
- Even without prompting, `mget` and `mput` still produce lots of output—reporting established data connections, transfer rates, file sizes, and so on. When transferring many small files, this is sometimes undesirable. With the `verbose` command you can toggle FTP between the silent and verbose modes.
- Most people don't know that `ftp` also has `mkdir`, `rmdir`, `delete`, `mdelete` and `rename` commands. All of these operate on the remote system.
- With `lcd` you can change the local working directory (the `cd` command within FTP is remote). For quickly inspecting local directory listings or for performing any other local UNIX command without quitting `ftp`, you can use a shell call via the exclamation mark as escape character:

  `!ls -l`

More information of `ftp` is found in UNIX manuals (use `man ftp`), or within `ftp` by using the `?` command for getting a list of available `ftp` commands.

Making FTP More Automatic

With FTP, you must first log in and specify a password. This can be simplified by creating a file `~/.netrc` that can contain a user name and, optionally, the corresponding password for an arbitrary number of FTP sites, e.g.:

```
machine www.nmr.corp.com login anonymous password
rob.jones@nmr.corp.com
machine news.nmr.corp.com login usergroup password news4web
machine codonics1 login anonymous password 2
```

The format is the keyword `machine` followed by a host name (either a local host, a fully qualified Internet address, or a numeric IP address), the keyword `login` followed by a login name (for anonymous FTP, use `anonymous` as login name), and finally the keyword `password` followed by the appropriate password (for anonymous FTP sites use your e-mail address as the password). The `password` keyword and password is optional.

You can add as many entries as you like. If the file contains passwords, it must have permission 600; otherwise, `ftp` refuses to use it:

```
chmod 600 ~/.netrc
```

When calling `ftp` with a host name and password specified in `~/.netrc`, you can bypass the login procedure and directly start with the FTP transactions, for example:

```
> ftp -i www.nmr.varianinc.com
Connected to gabriel.
220 gabriel FTP server (SunOS 5.6) ready.
331 Guest login ok, send ident as password.
230 Guest login ok, access restrictions apply.
ftp> cd pub/patches
250 CWD command successful.
ftp> bin
200 Type set to I.
ftp> mget 5.3BacqSOLmer103*
200 PORT command successful.
150 Binary data connection for 5.3BacqSOLmer103.Readme
(132.190.187.200,40020)
(1472 bytes).
226 Binary Transfer complete.
local: 5.3BacqSOLmer103.Readme remote:
5.3BacqSOLmer103.Readme
1472 bytes received in 1 seconds (1.4 Kbytes/s)
200 PORT command successful.
150 Binary data connection for 5.3BacqSOLmer103.tar.Z
(132.190.187.200,40021)
(167569 bytes).
226 Binary Transfer complete.
ftp> !ls -l
total 2908
-rw-r--r--   1 vnmr1    nmr 1472 Jan  9 01:38
5.3BacqSOLmer103.Readme
```

```
-rw-r--r--   1 vnmr1    nmr 167569 Jan  9 01:38
5.3BacqSOLmer103.tar.Z
ftp> bye
221 Goodbye.
>
```

or in silent mode (using the `verbose` command), with less output:

```
> ftp -i www.nmr.varianinc.com
Connected to gabriel.
220 gabriel FTP server (SunOS 5.6) ready.
331 Guest login ok, send ident as password.
230 Guest login ok, access restrictions apply.
ftp> cd pub/patches
250 CWD command successful.
ftp> bin
200 Type set to I.
ftp> verbose
Verbose mode off.
ftp> mget 5.3BacqSOLmer103* 5.3BpsgSOLmer101*
5.3BgenSOLall101*
ftp> bye
>
```

Calling ftp Within Shell Scripts

With the presence of a `~/.netrc` file, it even becomes possible to call `ftp` within a shell script and without user interaction. An example where this could be used is with printing on a Codonics dye sublimation printer. These devices are driven by a SPARC engine, and plots are typically submitted through FTP. The user name is arbitrary, but the password is a number that specifies what print options are to be selected (e.g., 2 is a typical value for printouts in landscape format).

The `~/.netrc` file shown above includes an example line for such a printer, named `codonics1`. The following shell script `imagepr` permits submitting a plot to that printer by specifying the name of the file to be printed (typically a Sun raster or a PostScript file):

```
#!/bin/sh
ftp codonics1 << +
bin
put $1
bye
+
```

This script (make sure it is executable) can now be called with a file name, for example:

```
imagepr image.rs
```

We could also suppress the `ftp` output to make this script silent:

```
#!/bin/sh
ftp codonics1 > /dev/null << +
bin
put $1
bye
+
```

12.13 NIS and NIS+

NIS, the Network Information Service (initially called the “yellow pages”) is a database for networks that allows for an easy maintenance of system administration files throughout a network. NIS covers files such as `/etc/passwd`, `/etc/shadow`, `/etc/group`, `/etc/networks`, `/etc/hosts`, and many others. The advantage is that all those files are centralized and need only to be maintained on a single host, whereas the others fetch this information via network.

With Solaris 2.x, NIS is being replaced by the newer **NIS+,** which offers enhanced security by using encryption and other features. NIS+ retains some back-compatibility with NIS.

Within an NIS/NIS+ domain, which or may not may not include all hosts within a network, there are three classes of hosts:

- NIS master server
- NIS slave servers
- NIS clients

The slave servers improve network safety by maintaining a duplicate of the master data base, the case the master is not operative.

Unfortunately, setting up an NIS+ service is a rather complex operation and covering that is far beyond the scope of this manual. Refer to the Solaris documentation *Name Services Administration Guide* and *Name Services Configuration Guide* for more information.

12.14 Heterogeneous Networks

Ethernet can be used to talk to computers other than Suns, of course. In fact, various configurations allow communication via Ethernet:

- The other system is a UNIX system and has NFS. Such an implementation should be straightforward.
- The other system is a UNIX system and does *not* have NFS, but TCP/IP is present. The implementation should still be simple. Commands like `rlogin`, `rsh`, and `rcp` work, but not NFS mounting of file systems.
- The other system is a non-UNIX system, such as a VAX under VMS. Either the other system requires TCP/IP or you’ll need to have special software on the Sun to talk to the non-UNIX system. For example, TCP/IP is available for VAX computers but is expensive. The less expensive DECNET for the Sun achieves similar results but you have to review appropriate manuals to operate it.
- Other computers without standard operating systems—such as Gemini, VXR, and XL spectrometers—require special software (such as the Varian limNET software in the case of these systems) for both the remote computer and for the Sun to allow for some lower-level but still fast file transfers.
- TCP/IP, `ftp`, and `telnet` are also available for Apple Macintosh computers, and PC running Microsoft Windows. Therefore, a high-level protocol can be used, even with these systems.

12.15 Remote Home Directories

In a network, it is not necessary to have a home directory on each system for each user. Every user can basically have a single home directory. How can this be achieved?

Using the Automounter

Setting up remote home directories is relatively easy with the automounter, see "Automounter," page 148. You set up an automount map `/etc/auto_home` according to the following scheme:

```
vnmr1    server_name:/export/home/vnmr1
vnmr2    server_name:/export/home/vnmr2
```

provided that on the server system the file system `/export/home` is shared, see "NFS Mounting Options, Sharing a File System," page 144. You can then define these users with the same user- and group-ID as on the server and with a home directory in `/home` (rather than `/export/home`), using the Solaris AdminTool.

In the case of network problems, however, users with automounted home directories cannot even log in. If everything works as it should, a user's home directory is mounted automatically when the user logs in, and it is automatically unmounted five minutes after a user logs out.

If you prefer a solution with static mounting and with a possible fall-back position in the case of network problems, see the next section.

Using Standard NFS Mounting

For standard NFS mounting, on each system where that user should have access, the password and group files (and eventually other files that regulate the access to certain files or systems) must be updated (this is easy with NIS). Then, an empty home directory is generated on all systems (`/home/`*`username`*). This home directory might contain some default files (dot files, eventually also a small `vnmrsys` for VNMR) that would allow the user to log in and work, even if the network cannot be reached. The path to that home directory may be the same on all systems. This would allow fetching all information in the password file via NIS (see "NIS and NIS+," page 153).

Next, on each remote system, the full home directory must be mounted onto that directory. The mount should be read-write (type NFS, of course) and, therefore, hard mounted. The corresponding entry in `vfstab` would be, for example:

```
inmr5:/export/home/eva - /export/home/eva nfs - yes rw,bg
```

Once the home directory is mounted with the command `mountall -F nfs` or by rebooting, the contents of the default directory on the local machine are hidden so they can only be accessed by unmounting the NFS-mounted home directory exported from the remote host. The home directory on the remote host can be accessed like a local home directory. `/etc/dfs/dfstab` on the file server must allow mounting the home directory remotely.

Alternatively, if the whole remote disk or at least the file containing the home directory is NFS mounted (it would have to be hard mounted, with read-write access again), a simple symbolic link can be created instead of a default home directory:

```
ln -s /sun/home/eva /export/home/eva
```

This would be less safe solution, in that it would not provide for a home directory if the network is down.

Note that if a spectrometer host is involved, `Acqproc` must have write access to the home directory. There can be problems if the disk with the active experiment is not local to the spectrometer. This can be fixed by exporting the file system with `root` access for remote systems, see "Mounting File Systems (mount, /etc/dfs/dfstab)," page 142. Without that, the acquisition process, which is owned by `root` and has only "other" permission on remote systems (its user-ID is reset to 60001 for "nobody" on the remote host) cannot access the experiment files unless permissions are opened up dramatically (which is undesirable).

12.16 Gateway Machines

A *gateway machine* is a CPU that can talk to two different networks, usually on two Ethernet cables. It links two different networks and allows all users of the linked networks to talk to each other. This can be repeated on the added-on network and allows creating large scale networks.

What are the requirements for the gateway machine?

- It requires two Ethernet controllers. On Sun workstations, this means an additional Ethernet controller board. For installation of the extra board, see Sun manuals. Obviously, systems without a free expansion slot cannot be made a gateway system.
- It has two identities—one network number and one host name per network

The file `/etc/hosts` on a gateway contains two lines for the local CPU, defining two unique host names for both networks:

```
192.9.200.1          sun loghost       #(ie0)
192.9.201.33         vnmr-net          #(ie1)
```

The same file also contains the network addresses for all additional systems the gateway should talk to.

To activate the Ethernet hardware (through `rc.boot` at bootup time), you need an equivalent to `/etc/hostname.le0`. The file is named `/etc/hostname.le1` for standard Ethernet interfaces and contains the host name for the other network (`vnmr-net` is used in our example).

After these changes, the gateway must be rebooted using `boot -r`. The `/etc/hosts` file also must be updated on all other systems to include the newly accessible network addresses, unless the gateway is also the NIS master server.

Spectrometer Hosts Should Not Act as Gateways!

From a hardware configuration point-of-view, host computers of networked UNITY*INOVA*, *MERCURY*-Series, and *GEMINI 2000* spectrometers are also gateway computers if they have two Ethernet interfaces. Hosts computers setup using a single network card and router are not gateway computers. We do not activate any gateway functionality in either the dual network card hosts or the single network card hosts, however, because the acquisition network is not intended for communication with other networks. Creating a file `/etc/defaultrouter` ensures that no Internet routing or name daemon (`in.routed` or `in.named`) is started on a spectrometer host. If any of these daemons is running by mistake, the spectrometer host can be severely slowed down. An even safer way to suppress unwanted routing activities is to create a file `/etc/notrouter`, by entering:

```
su
touch /etc/notrouter
```

12.17 Diskless Systems

Solaris permits setting up servers that store all the disk-based information for other Sun computers on local disks. These other systems then become diskless clients, because they don't have any local disks. The server knows the Ethernet hardware address for these clients. Upon booting, a client broadcasts its Ethernet address, and the server responds with its own and the client's IP addresses. The client then downloads the booting software, followed by the kernel (from its dedicated root partition on the server), and finally it starts UNIX, using shared partitions on the server.

Such a setup has some advantages:

- All disk-based information is stored, administered and backed up at a central location. Once it is set up, the system administration is centralized and easier.
- All user software needs to be installed only once.
- Because most of UNIX is stored in shared partitions and all system share the home directory and data partitions, disk space and disk hardware is saved. But because disk space is no longer excessively expensive, this argument loses much of its validity.

But there are also disadvantages:

- The initial setup is considerably more complex.
- There are performance losses due to network speed limitations.
- With fast networks, the disk might become a performance bottleneck.
- Such a setup is more vulnerable. If either the server or the network is down, the clients are down as well.

Because such configurations are now used even more rarely than a few years ago (when disks were much more expensive relative to the costs of a workstation), this topic is not discussed further in this manual.

Chapter 13. Other Networking Options

Sections in this chapter:

- 13.1 “Hardware Options,” this page
- 13.2 “Direct Connections,” page 158
- 13.3 “Using a Modem,” page 159
- 13.4 “Using Mail on a Client Machine,” page 160
- 13.5 “Avoiding Disk Full Problems with Mail,” page 160
- 13.6 “Other Software,” page 162

This chapter is an attempt to cover the broad topic of how to connect two Sun systems through a direct RS-232 serial link, a modem, a phone connection, or a packet-switched data network (PSDN). Also included are some instructions on using electronic mail (e-mail) on systems connected to Internet:

This topic is quite extensive, the software and its configuration files are rather convoluted and, to some extent, specific to the hardware involved. It is not possible to cover this topic completely in this manual. In many cases, it will be necessary for you to seek the help of an expert to solve site-specific problems.

13.1 Hardware Options

The simplest way of connecting two systems via serial line is a direct RS-232 connection. Compared to RS-232, Ethernet is about 100 times faster, simpler to use, more reliable, built into every Sun, and can span over longer distances (depending on the hardware). For these many reasons, the RS-232 connection is almost never used except for a special application, such as to provide some loose but safe connection between a mail server on a public network and another system on a local network, to prevent outside people from breaking into a local network. With such a connection, overall transfer rates of up to approximately 1750 bytes per second (at 19,200 baud) can be obtained (compared to several hundred kilobytes per second with Ethernet).

The main application for connections via serial lines is to link two systems that are not located in the same site. In fact, any two systems in the world can, in principle, be connected using the serial ports. There are two basic options for doing this: using regular phone lines in a dial-up connection or using a dedicated line into a public data network. Phone lines are typically accessed via modem. Common modem transfer rates are 9600 to 28800 baud (about 1000 to 3000 characters per second) or sometimes over 50 kilobaud using data compression.

To access public networks (assuming you cannot access the Internet via local network and gateway), a simple dial-up serial line using a modem is not good enough. You need a full IP (TCP/IP) connection such that e-mail transfers, FTP (data transfers with error checking),

telnet (remote logins in general), X terminal operation over Internet, and Web browsing are possible. There are a number of alternatives:

- Direct networking connection to a local gateway computer
- PPP (or SLIP) over a modem
- PPP over ISDN
- X.25 over a dedicated line
- Leased line access

These alternatives are often accessed through a dedicated line. Special protocols using synchronous data transfer allow for transfer rates between 2400 baud and 64 kilobaud (about 6000 bytes per second). The line rental fee depends mostly on the data transfer rate. Alternatively, dial-up connections to the data networks are permitted (using normal modems, but mostly only at 1200 to 2400 baud, or about 100 to 200 characters per second). Usually, this precludes automatic mail delivery to the dial-in site.

Most universities are already hooked up to public networks, and users that are linked to the local university data network can also access public networks by communicating to a gateway computer via Ethernet. In these special situations, there are local experts who know how to establish a connection, and this case will not be covered here (except for the discussion of `mail`).

13.2 Direct Connections

Direct data transfer is possible without a software protocol, with the `tip` program, and using `uucp` on a direct line. Each is covered in the following sections.

Transfers without Software Protocol

Direct data transfer can be achieved without any special command or utility. Simply connect two Sun systems via one of their RS-232 ports such as `/dev/ttya` (port a), using a null modem (see "Connecting Serial Terminals," page 190). On the receiving system you type

```
cp /dev/ttya targetfile
```

This puts anything that comes from port a into a new file *`targetfile`*. On the transmitting system, you can now send the contents of that new file to the RS-232 connection:

```
cat sourcefile > /dev/ttya
```

This copies the file via RS-232. Because nothing has been specified, the transfer happens under standard conditions (9600 baud, 7 bits, no parity bit, 1 stop bit). The `stty` command can be used to change the speed if a different speed is required:

```
stty 1200
```

This command, which must be entered beforehand on both systems, changes the baud rate to 1200. The `echo` command automatically terminates the transmission by sending a Ctrl-d as terminating end-of-text (EOT) character, but on the receiving system, which is still waiting for further data, the file transfer must still be terminated by pressing Ctrl-c.

Using tip on a Direct Line

The above procedure provides only very rudimentary error checking, if any, and the integrity of the transferred data cannot be guaranteed. Also, it is rather awkward as a method, as it requires coordinated actions on both systems involved *for every file transfer*.

It is much better to use software such as `tip` that provides proper error checking, such as checksum comparison, and permits handling data transfers from one side only. `tip` is much more because it allows for a full-duplex terminal session on a remote host. Again, we connect two serial ports via a null modem. On the remote system, set up the corresponding port (`/dev/ttya` or `/dev/ttyb`) for terminal operation and send a hangup signal to the `init` process (see "Connecting Serial Terminals," page 190).

Note that the default speed on Sun serial ports is 9600 baud. You can use this value on both systems. The serial line serves as a one-way terminal connection, and you must not activate a login on the local port.

To define a port for `tip` on the local machine, edit the file `/etc/remote` by changing the entry `hardwire` for the correct port and baud rate (if necessary):

```
hardwire:\
        :dv=/dev/ttya:br#9600:el=^C^S^Q^U^D:ie=%$:oe=^D:
```

For an explanation of the entries, see the file `/etc/remote`. You can now use `tip`:

```
tip hardwire
```

At this point, you are in a terminal session, working on the remote system. You should get a login prompt and can work on the remote system. The login session is terminated by a entering `logout` or by pressing Ctrl-d.

To terminate the `tip` session (you don't log you out automatically), type a special escape sequence using the tilde (~) sign (other escape sequences serve to transfer files):

`~.`	Terminate `tip` session (no automatic `logout`)
`~p source <target>`	Send a file to the remote computer, and if *`target`* is not specified, it has the same name as the local file
`~t source <target>`	Receive a file from a remote computer, and if *`target`* is not specified, it has same name as the remote file
`~c <directory>`	Change directory on the local machine

Sessions using `tip` can also be preconfigured and customized with the file `~/.tiprc`. See Sun documentation for details. Note that `tip` can only transfer ASCII files, because control characters are used to terminate the file transfer.

Using uucp on a Direct Line

The `tip` program only allows for online file transfers during a login session, and only ASCII files can be transferred. The `uucp` (UNIX-to-UNIX copy) program allows off-line transfers and some binary file transfers. Using `uucp` on a direct line provides for a loose, but safer link, between a system with public network access and a local network.

13.3 Using a Modem

Not only is there a multitude of modems on the market, but each modem type has different properties and requires a different setup (a different software interface, in essence). Trying to cope with the different modem types on the market would be beyond the scope of this manual. You are referred to the documentation that accompanies the modem.

13.4 Using Mail on a Client Machine

A mail client machine is a system that only knows two classes of mail—local messages and all other messages. Mail between two local users is distributed directly, and all other mail is sent to a mail server for further distribution.

Mail administration on such a machine is fairly simple. In `/etc/hosts`, add the alias `mailhost` to the machine that is the mail server on your network, for example:

```
192.9.200.1      unity500 loghost
192.9.200.2      vxr400
192.9.200.3      vxr300
192.9.200.4      sun mailhost
127.0.0.1        localhost
```

You don't even have to reboot. Your system is now ready to send e-mail to any place and user the mail server can reach.

13.5 Avoiding Disk Full Problems with Mail

Receiving mail is a slow process because a network (TCP/IP over Ethernet or X.25, or `uucp` over X.25 or modem) is involved.

Receiving Large Mail Files

Incoming mail is first built up in the directory `/var/spool/mqueue` or in the directory `/var/spool/uucp/<`*`remotehostname`*`>` on systems using `uucp` for mail transfers. While mail is arriving, a typical listing of the `mqueue` directory may look as follows:

```
-rw-------  2 root            0 Aug  5 14:12 lfAA01781
-rw-------  2 root            0 Aug  5 14:12 qfAA01781
-rw-r--r--  1 root            0 Aug  5 14:12 xfAA01781
-rw-------  1 root            0 Aug  5 14:12 dfAA01781
```

The file `df<`*`message-ID`*`>` receives the incoming message and grows as the message arrives (in many cases, the directory listing does not indicate the intermediate or final size, because the directory is never adjusted before the file is deleted again). The other files in the directory (`lf<`*`message-ID`*`>`, `qf<`*`message-ID`*`>`, etc.) are described below.

After the entire message is received, it is then copied into the file `/var/spool/mail/<`*`username`*`>`. Temporarily, disk space of twice the size of the message is required (the source file is only deleted when the target file is completely written out). Later on, when looking at the message through `mailtool`, the `mailtool` program opens a temporary file `/tmp/MTda<`*`PID`*`>`, and again, disk space of twice the size of the message is required.

Sending Large Mail files

Building up an outgoing message is a fast process (several megabytes of disk space can be eaten up in a few seconds) because it is purely local. As a consequence, the disk may get filled very quickly, but you may not be able to see which file is using up the disk space, because the directory entry for that file is probably never really made (it would only show up after completing the file).

The only command that shows disk space usage is `df`. It turns out that for sending mail, the data is built up in `/tmp`. The message is built up in one file and then copied into a file `/var/spool/mqueue/df<message-ID>` such that, temporarily, the space requirement is twice the size of the message (concluding from the disk space consumption, it seems that before copying the data to `/var/spool/mqueue`, a second copy of the data is made temporarily within `/tmp`).

The other files in `/var/spool/mqueue` are used with the spooling software: `lf<message-ID>` and `xf<message-ID>` are lock files, and `qf<message-ID>` contains routing and header information for `sendmail` and `smtp`.

How to Avoid Problems with Mail

Solutions to disk full problems with mail include the following::

- *Increase the size of the root partition* – This is not really recommended because it is not trivial (it requires reformatting the entire disk and reloading UNIX) and it permanently takes away disk space from other partitions. Furthermore, with a smaller internal hard disk (e.g.: less than 500 MB), it is often impossible to increase the size of the root partition, because this limits the size of the swap space or the size of `/usr` (and hence the number of loadable software options) or both. See Chapter 1, “Solaris 9 Installation,” or Chapter 2, “Solaris 8 Installation,” for a method for changing the size of the disk partitions during the Solaris installation.
- *Move the spooling directories to another disk partition* – This has the advantage that it doesn't require changing disk partitioning (you can work with standard disk partitions) and no additional free space is permanently locked away in the root partition. Moving spooling directories to another partition also resolves problems with editing large files, and it can easily be expanded to resolve problems with plotting large files from a remote system (so the data resides in `/var/spool/<hostname>_<portname>`, not in `/vnmr/tmp`). The following procedure assumes a large disk partition `/data` (with enough free space guaranteed) exists:

  ```
  su
  cd /
  tar cf - tmp var/spool/mail var/spool/mqueue |(cd /data; tar xvfBp -
  mv tmp tmp.std; ln -s /data/tmp tmp
  cd var/spool
  mv mail mail.std; ln -s /data/var/spool/mail mail
  mv mqueue mqueue.std; ln -s /data/var/spool/mqueue mqueue
  ```

  It is obvious that filling `/data` can now disable mail spooling. It should be made sure that this partition is never filled to 100%.

  Systems using `uucp` should also include `/var/spool/uucp`:

  ```
  su
  cd /
  tar cf - tmp var/spool/mail var/spool/mqueue var/spool/uucp | \
      (cd /data; tar xvfBp -)
  mv tmp tmp.std; ln -s /data/tmp tmp
  cd var/spool
  mv mail mail.std; ln -s /data/var/spool/mail mail
  mv mqueue mqueue.std; ln -s /data/var/spool/mqueue mqueue
  mv uucp uucp.std; ln -s /data/var/spool/uucp uucp
  ```

A more general solution to spooling disk space problems would be the following:

```
su
cd /
tar cf - tmp var/spool | (cd /data; tar xvfBp -)
mv tmp tmp.std; ln -s /data/tmp tmp
cd var
mv spool spool.std; ln -s /data/var/spool spool
```

Remember, this is not a 100% safe solution. The directory `/data` can also fill up, which then can lead to problems receiving and sending mail, printing and plotting, and even editing files. On the other hand, this may be the only solution that permits sending or receiving large data files.

13.6 Other Software

If data should be transferred to a PC that is not equipped with Ethernet, this can also be achieved through serial line (RS-232) using the Kermit protocol. For Sun computers running VNMR software, the Kermit software has been submitted to the user library (for Sun-4 computers only). No further discussion of this subject is provided here.

For Apple Macintosh computers, a software package, PlotView (from Stevens Creek Software, 21346 Rumford Drive, Cupertino, CA 95014), allows capturing plotter output directly through RS-232 (the Macintosh is hooked up as a plotter). This software is made specifically for HP-GL plotter output and is designed for direct incorporation of spectral results into reports, publications, etc.

Another application of RS-232 communication is the use of terminals and terminal emulation software on IBM PCs and Apple Macintoshes. This (as well as the use of X terminals and terminal emulators) is not discussed any further here, mainly because in this case the processing happens on the workstation, and only display information is transferred to the terminal.

Chapter 14. Text Editing

Sections in this chapter:

- 14.1 “Text Editors dtpad and textedit,” this page
- 14.2 “Text Editor vi,” page 164
- 14.3 “Other Editors,” page 168

UNIX includes several programs for editing text, including `textedit`, `vi`, `ex`, and `ed`. Each is described in this chapter (`textedit` and the line editors `ex` an `ed` only briefly):

For further information, refer to Sun manuals. Refer also to the *VNMR Command and Parameter Reference*.

From VNMR, UNIX editors can be called directly, for example, by `macroedit` or `macrovi`. The VNMR command `macroedit` calls an text editor that is defined in the variable `vnmreditor`. It can be changed by

```
setenv vnmreditor new_editor
```

where *`new_editor`* is one of the editors listed above, and can be made active by recalling VNMR from this shell. For a permanent change, add this command to the user’s `.login` file. The VNMR command `macrovi` always uses the `vi` editor.

14.1 Text Editors dtpad and textedit

dtpad

Both Solaris 8 and Solaris 9 include the CDE program text editing program, `dtpad`. This is a mouse-oriented, window-based text editing tool that always runs in its own window and is called by mouse clicking on the editor icon or right mouse clicking on the background and selecting it from the pop up menu.

The `dtpad` program is easy to learn. It is always in the insert mode, the mouse can be used to set the cursor point to insert or delete characters, and a pop-up menu allows access to more complex functions.

Drag-and-drop feature (see “Shortcuts within Open Look and CDE,” page 81) is supported with `dtpad` , making the editor even more attractive. `dtpad` can also be made to automatically wrap the lines at word breaks, not just at any character. This makes it look like it is automatically formatting lines. However, note that `dtpad` in such a case does *not* insert a newline character.

vi

When using `vi` for editing a file that has been generated or modified using `textedit`, you may encounter very long lines of text. With `dtpad`, the CDE text editor, this can be avoided: it is possible to have linefeeds inserted in wrapped lines when a text is saved.

You can access the Edit menu easily, which gives you fast and direct access to the Again, Undo, Copy, Paste and Cut functions. Within the Open Look environment, all menus (except for the Extras) are available from the top of the window as well as within the window (menu mouse button).

Experienced UNIX users will probably prefer the `vi` text editor, which is more complex (three modes) and has a less sophisticated user interface (the mouse is not used) but is faster and more powerful, especially for repetitive tasks such as making global changes. The real advantage of `textedit` and `dtpad` is that they can be used reasonably well after a few minutes of training, whereas `vi` requires much more time to learn completely.

14.2 Text Editor vi

The `vi` program (pronounced “vee-eye”) is the standard full-screen editor under UNIX, compatible with all full-screen terminals and RS-232 terminals. Although the `vi` editor is powerful, it is much more complex than `textedit and dtpad`, and the user interface for `vi` is not as sophisticated. `vi` does not use a mouse or menus and provides very little status information to the user. On the other hand, `vi` is particularly well suited for editing line-oriented documents such as source code, tables, and UNIX text definition files.

`vi` is called with the command `vi` *`filename`*. If `vi` is entered with no argument, the file to be edited can be read in from within the editor. If a single file is given as an argument, `vi` opens with that file on display. If multiple file names are given (wildcard characters can be used), `vi` reads each file in sequentially for editing one at a time.

`vi` operates in three modes, called command, insert, and last line.

Command Mode

`vi` starts in the *command mode,* where you can do the following:

- Move the cursor around the file
- Enter commands to work with the existing text
- Change to the insert mode to add text
- Change to the last line mode quit `vi`, read another file, etc.

A sample of commands in the command mode is given below. Because there is no command line, the commands do not show up on the screen but are executed immediately without pressing the Return key.

Moving Around the File

To move around in a file, press the up-down-left-right arrow keys on the keyboard or enter the following commands:

`k`	Move cursor up one line
`j`	Move cursor down one line
`l`	Move cursor one character right
`h`	Move cursor one character left

`G`	Go to the start of the last line in the file
`3G`	Go to the start of line 3
`0` (zero)	Go to the start of the current line
`$`	Go to the end of the current line
`w`	Go forward one word (to the start of the next word)
`3w`	Go forward three words
`b`	Go backward one word (to the start of the current word)
`5b`	Go backward five words
Return	Go to start of next line
`3` Return	Go to start of line 3 lines below cursor
`-`	Go to start of previous line
`3-`	Go to start of line 3 lines above cursor
`(`	Go to start of current sentence
`)`	Go to start of next sentence
Ctrl-d	Scroll down (forward) half a screen
Ctrl-f	Scroll forward by a full screen
Ctrl-u	Scroll up (back) half a screen
Ctrl-b	Scroll back by a full screen
`/`*`expression`*	Find next occurrence of *`expression`*, jump to its first character
`?`*`expression`*	Find last occurrence of *`expression`*, jump to its first character
`n`	Find the next occurrence of *`expression`* (the last search)
`N`	Find the last previous occurrence of *`expression`* (the last search)

Working with Text

The following commands are representative of the many commands available for working with text from the command mode.

`x`	Delete 1 character
`dw`	Delete word
`dd`	Delete 1 line and put it into the buffer
`3dd`	Delete 3 lines and put them into the buffer
`yy`	Yank a line (put current line into buffer)
`3yy`	Yank 3 lines
`J`	Join the next line to the current line
`p`	Insert line(s) from buffer below current line
`P`	Insert line(s) from buffer above current line
`r`	Replace one character by one other character
`u`	Undo the last command
`.`	Repeat the last command
`~`	Change character from upper case to lower case and vice versa
Ctrl-l	Redraw the screen (e.g., remove error messages covering the text)
`ZZ`	Write if necessary and quit `vi`

Changing to the Insert Mode

Any of the following commands enable you to add or replace text. When you finish adding text, press the Esc key to return to the command mode.

`a`	Append text after the current cursor position
`A`	Append text to the end of current line
`i`	Insert text before current cursor position
`I`	Insert text at beginning of current line
`cw`	Change word from current cursor position to end
`c3w`	Change 3 words from current cursor position to end
`c)`	Change sentence from current cursor position to end
`C`	Change line from current cursor position to end
`o`	Open line below current line
`O`	Open line above current line
`R`	Replace (overwrite text character by character)
`s`	Substitute one character

Changing to the Last Line Mode

The last line mode is used to quit `vi`, read and write files, call another file for editing, open a UNIX shell, and similar actions. To change from the command mode to the last line mode, type a colon ":" as the first character of a line.

Insert Mode

A series of commands (listed above) switches you to the *insert mode*, where all the text typed on the keyboard (except for the Esc key) shows up in the text. The only way to exit the insert mode is to press the Esc key, which leads back to the command mode.

Unfortunately, there is no indication on the screen whether `vi` is in the command mode or in the insert mode. Inexperienced operators often press the Esc key to make sure they are in the command mode, but as you become familiar with `vi`, this will be seldom necessary. The Esc key is also the only possibility to avoid the execution of commands that have been partially typed, such as when the number has been typed, but not the last character.

Special (normally not displayable) characters can be inserted into the text if they are preceded by Ctrl-v. For example, pressing Ctrl-v Ctrl-q is displayed in the text as "^Q".

Last Line Mode

The *last line mode* is also called the `ex` command mode because in this mode you enter commands from the `ex` program:

`:r` *`file`*	Read a file named *`file`*
`:w`	Write back file
`:w` *`file`*	Write under a new file with name given by *`file`*
`:q`	Quit `vi` (quits only if file is unchanged since opened or saved)
`:wq`	Write back and quit `vi`
`:q!`	Quit editor without saving changes
`:n`	Edit next file
`:sh`	Open a shell

`:%s/old/new/g`	Replace all occurrences of string *old* by string *new*
`:se nu`	Switch on line numbering
`:se nonu`	Switch off line numbering
`:set all`	Show all options
`:e!`	Reedit file, discard changes
`:!commands`	Execute UNIX commands (*commands*) in a temporary shell, return to `vi` afterwards

The `ex` program was an early UNIX line-oriented text editor. For compatibility, `ex` commands were integrated into `vi` as the last line mode, making `ex` a subset of `vi`. With two exceptions, the last line mode is initiated by typing a colon as the first character of a line; thereafter any `ex` command can be entered. The execution of these commands, which requires pressing Return, automatically leads back into the command mode.

Other vi commands

The sections above list only a selection of the most important `vi` commands. For a complete description, refer to the Sun manuals and UNIX textbooks. Many powerful features are available, but they normally are only required for experienced and frequent users. Such features include:

- Options, such as special setups for writing programs.
- Customization file (`.exrc`).
- Macros.
- Abbreviations to avoid retyping long and frequent expressions.
- Additional buffers to increase flexibility in moving text portions around.

For example, the following inline shell calls provide advanced formatting and sorting of text:

`!}fmt -63`	Reformats current text between the cursor and the end of the paragraph to 63 characters per line
`!]]fmt -70`	Reformats current text between the cursor and the end of the file to 70 characters per line
`!}fmt -c -75`	Reformats current text starting at the cursor position (first line) and indents all following lines to the first character of the second line
`!}sort -u`	Alphabetically sorts all lines up to the end of the current paragraph in the text starting at the cursor position, removing duplicate lines.

Problems with Running vi

The `vi` editor formats the screen by moving the cursor around. This can only work if

- The terminal characteristics support such cursor movements
- `vi` knows the exact terminal definition

If there are problems with these points, `vi` either incorrectly formats the screen, making proper editing almost impossible (type ZZ to quit the editor) or switches to “open mode” (it tells you about this on the last line), which essentially is the single line `ex` editor (in this case, type `:q` Return to exit the editor). Other problems are more subtle and relate to shell calls within `vi`, see below.

Using vi in Single-User Mode

Trying to use `vi` in single-user mode can be unsuccessful for several reasons:

- The terminal type is undefined, which makes it impossible for `vi` to format the text screen (or window) properly and control the cursor movements.
- The `PATH` variable might not include `/bin` or `/usr/bin`, such that the `vi` command cannot be found.

There is an easy way out, which gives access to `vi` even in single-user mode (note that this is in a Bourne shell environment—you cannot use `setenv`):

`TERM=sun`	Defines the terminal type.
`export TERM`	Makes `TERM` information available to child processes (e.g. when calling a shell from within `vi`)
`vi`	Calls `vi`

If the `PATH` does not include `/bin` or `/usr/bin`, call `vi` with the full path by entering `/usr/bin/vi`.

Problems with Running vi Remotely

After a remote login through `rlogin` or `telnet`, `vi` often does not work properly because the `term` variable (`TERM` in the Bourne shell) is not defined properly. In particular, the X environment, the `term` variable after a remote login is usually set to `xterm`, which is okay if you are working in an `xterm` window but not in a `shelltool` or `dtterm` (the standard CDE shell window). To correct this, use the following commands:

`set term=dtterm`	For `dtterm` (CDE shell windows)
`set term=sun`	For `shelltool` (nonscrolling OpenWindows shell windows)

After this, `vi` should function as expected. If you are (remotely) logged in as `root` and/or working in a Bourne shell, use `TERM=dtterm` or `TERM=sun` instead.

Problems with Running vi Remotely in CDE

A peculiar problem arises when calling `vi` from a standard CDE window (`dtterm`) while logged in to a SunOS (and probably any other non-Solaris or non-CDE) system—the proper setting for the `term` environment variable is `dtterm`. In Solaris 8 and newer (`/etc/termcap`) contains a definition for the terminal type `dtterm`. This was not true in earlier Solaris released. If the path is not set up at login, `dtterm` is located in `/usr/dt/bin`. A simple solution is to temporarily switch to the DEC VT-100 terminal definition by entering `set term=vt100`. For basic `vi` usage, this seems to work reasonably well.

14.3 Other Editors

The other editors in the standard Sun software are line-oriented in which you work on only one line at a time (apart from printing portions of the text). These editors were basically written for early computer terminals such as the teletype. Under these conditions, editing consisted of entering, modifying. and displaying (printing) single lines.

Obviously, such editors are much more difficult to use than full-screen editors. UNIX provides two of them: `ed` and `ex`.

ed and ex Editors

The `ed` program was the first UNIX text editor. It has a very terse user interface (e.g. a single error message: the question mark) and is not very comfortable.

The `ex` editor is actually a subset of `vi` and, therefore, provides a better user interface (at least one with decent error messages) than `ed`. At the same time, `ex` is an extended and enhanced version of `ed` and was written at the University of California at Berkeley by William Joy.

Line-oriented editors are only used in special circumstances: `ex` can be useful if for some reason only a line terminal is available (which is seldom the case, because most standard RS-232 terminals are supported by `vi`) or if for some reason the terminal setup in UNIX has been corrupted. `ed` is used if for some other reason only a limited UNIX command set is available (e.g., `vi` command file corrupted or erased). Because such cases are rare, these editors are not explained in detail here. A good introduction to both of them is found in the Sun manuals.

emacs Editor

Other editors are available on the market for the Sun. Best known among them is `emacs`, a sophisticated and powerful, screen-oriented editor that many people prefer over `vi`. `emacs` seems to combine the good parts of both `vi` and `textedit`. It is window-oriented, mouse-friendly, modeless, and yet still fast and easy to use. On the other hand, it makes heavy use of the Fn and Rn function keys, and many people don’t like that. Unfortunately, it is not part of the standard Sun software, but Sun user groups can make it available for little or no money because it is public domain software.

Chapter 15. Using Tapes and Floppy Disks

Sections in this chapter:

- 15.1 “tar Command,” this page
- 15.2 “ufsdump, ufsrestore, and dd Commands,” page 177
- 15.3 “Using Floppy Disks,” page 178

OpenWindows includes the Tape Tool and File Manager programs for handling tapes and floppy disks. The OpenWindows desktop utilities can also be called from CDE via the Application Manager. This chapter provides some information about the UNIX commands used by these tools, which is necessary for a full exploitation of those media or for a direct use in a shell window:

15.1 tar Command

The Tape Tool is an interface for selected options of the UNIX `tar` command (derived from “tape archive”). This tool can be used to transfer files by `tar` to or from devices such as a tape drive, floppy disk drive or file. The button labeled props... opens a popup menu for configuring the `tar` command. To fill in the necessary options or to run the `tar` command from a shell, you need to know the structure and features of the `tar` command. We start with tape operations, the most frequent application of the `tar` command.

The `tar` command can be used to read and write files to any kind of tape. It is the standard command for storing directories with all subdirectories on tape. Files that have been written with `tar` must be read back with the same command. `tar` is quite straightforward to use, but it has some drawbacks:

- It does not keep track of the amount of tape used. An attempt to write very large directories to tape starts correctly, but when the end of the tape is reached, `tar` issues an error message and does not ask for further tapes.
- It makes no mapping, therefore reading single subfiles can be time-consuming.

Entire file systems cannot be backed up with `tar`. The `ufsdump`, `ufsrestore` and `dd` commands do file system backup (`ufsdump` uses mapping and can write on multiple tapes where necessary). Chapter 6, “File Security, Repair, and Archiving,” covers this further.

Command Syntax

A typical call to `tar` is

```
tar xvbf 40 /dev/rmt/0mb
```

The `tar` command has a number of single character options, such as `xvbf` used here: `x` for extracting, `v` for verbose showing, `b` for blocking factor, and `f` for source/target file. Unlike most other UNIX commands, the options to `tar` are not preceded by a minus sign.

Some options require an additional argument (e.g., `40` for the option `b`, `/dev/rmt/0mb` for the option `f`). The sequence of arguments *must* correspond to the sequence of these options. `tar xvfb /dev/rmt/0mb 40` is also correct and has the same effect.

`/dev/rmt/0mb` is the location (source or target) of the `tar` file. In this example, the location is a magnetic tape (`rmt` identifies a removable magnetic tape) with logical number `0` and a medium write density `m`.

The final `b` means the device follows the BSD (Berkley Software Distribution); otherwise, the SVR4 (System V, Release 4 of AT&T) convention is used.

Note: If you want to write tapes that are compatible with SunOS 4.x and BDS UNIX, you *must* use the "b" device option (`/dev/rmt/0b`, `/dev/rmt/0bn`, `/dev/rmt/0lb`, `/dev/rmt/0lbn`, etc.).

tar Files

The `tar` program is a converter between the UNIX file system (a hierarchical structure) and a `tar` file, and vice versa. A `tar` file is a linear structure and has no directory. You must scan the file itself to find out what its contents are.

In most cases, the `tar` file is located on a magnetic tape. The `f` option indicates the location of the `tar` file. The default value (leaving out the `f` option) is `/dev/rmt/0`, which is taken from `/etc/defaults/tar`. The default can be altered there or with an environment variable `TAPE`. If you want `/dev/rmt/0mb` to be the default device, add

```
setenv TAPE /dev/rmt/0mb
```

to your `.login` file. If the default matches the device you need, you don't have to specify the `f` option (and the device name itself).

The standard target file for the `tar` command is `/dev/rmt/0`. Having more than one tape drive installed, the only requirement under Solaris 2.x is a different SCSI number. During the boot process (`boot -r`), the tape devices are configured automatically. You will find `/dev/rmt/0`..., `/dev/rmt/1`..., and so on. With `ls -l /dev/rmt`, you can see which of the targets are connected to which device number (...`st@4`...: tape at target 4).

`tar` automatically rewinds the tape at the end of the operation, unless its name is preceded with an `n` (e.g., `/dev/rmt/0n`).

The `tar` file can also be a disk file. Just specify a (plain) disk file, and `tar` creates (or reads from) a `tar` file on the disk:

```
tar cf userlib.tar userlib
```

This is useful for compressing data, because the `compress` command can only compress single files, not complete hierarchical file structures.

A common convention is to use the file name extension `.tar` for `tar` files. As a consequence, do not try to use `tar` with a nonexisting tape device (such as `/dev/rmt/3`). When writing to that "device," this would eventually create a plain `tar` file `3` in `/dev/rmt` that could easily fill the root partition to 111%, (see "Managing Free Space on Disks," page 201).

`tar` can also produce output to the standard output or read data from standard input if a minus sign is specified as `tar` file. This is useful for transferring directories (see "Using tar to Transfer Directories," page 175) or to create `tar` files on the disk:

```
tar cf - userlib > userlib.tar
```

QIC Tape Formats

All Sun tape drives can read quarter-inch cartridges (QIC) in 4- and 9-track mode. Tapes are automatically written in 9-track mode, providing maximum capacity. SPARCstations use different tape drives. They can only write in the 24-track, 150-MB QIC-150 format, but they can also read in the other two formats.

If a tape contains information in 4-track or 9-track format already, the SPARCstation detects that and refuses to overwrite because the tape has the wrong format (even if it is the right tape quality). In such a case, the tape has to be completely degaussed, either with a degaussing coil or by holding the tape to the area with the strongest stray field at the bottom of the NMR magnet, and then removing it slowly while wagging it (flipping it over) rapidly. The fringe field of a 200-MHz magnet may not be strong enough to erase tapes completely. In general, old 25 MB and 60 MB tapes can not be overwritten on a SPARCstation (even after erasing), because the tape characteristics don't match the tape drive.

SPARCstations with 150-MB tape drive require 3M DC6150, DC600-XTD (600 feet, *150 MB maximum*) or equivalent quality tapes for writing on tape. Other tape qualities can be only be read. For newer systems with 2.5-GB QIC tape drive, DC 9250 or equivalent quality tapes are required in order to exploit the full capacity, but DC 6150 tapes can also be used to store up to 150 MB. It is possible that tape drives on different Sun machines have slightly different head adjustments, so that tapes sometimes cannot be exchanged between specific machines.

Blocking Factor

As the name *streaming tape* indicates, cartridge drives operate in *streaming mode*, which means that the drive has to speed up first over a number *s* of sectors before it starts reading or writing a data batch of *b* blocks at full speed. To create a continuous disk file, the tape drive rewinds s sectors from the end-of-file mark and then spools forward *s+b* sectors (writing after the end-of-file mark), rewinds *s* sectors, and so on. Obviously, if *b* is the same or smaller than *s*, streaming tape operations are very inefficient. The size of the data batch is determined by the *blocking factor*, which indicates the number of 512-byte data blocks in a data batch. The value is entered as an argument using the `b` option.

The default blocking factor for tape commands is 20. The optimum blocking factor depends on the tape drive and the speed of the disk. For SPARCstations, the optimum is 30 to 40 (QIC-150 tape drives speed up much faster). An optimum blocking factor can speed up tape operations by up to 35% (SPARCstations).

Blocking factors of up to 2000 and more can be selected, but there is be no further gain in speed. Very large blocking factors can eventually lead to an error message

```
could not allocate memory
```

In such a case, try `tar` again with a smaller blocking factor or temporarily exit VNMR first. Because the total number of blocks written to the tape is a multiple of the blocking factor, very large blocking factors are inefficient for writing small amounts of data.

Writing Files

Writing a `tar` file is done using the `c` option, for example,

```
tar cvfb /dev/rmt/0 40 vnmrsys
```

(the `c` option requires the source file to be specified). `tar` can be used to write multiple files in one go. Just enter multiple file names or groups of file names, for example,

```
tar cvbf 40 /dev/rmt/0 file1 file2 file3
```

```
tar cvbf 40 /dev/rmt/0 *.fid
tar cvbf 40 /dev/rmt/0 .
```

As mentioned above, `tar` does not check on or keep track of the tape usage. It is up to the user to make sure there is enough space on the tape. Use `du` to check the data size before storing large amounts of data on tape.

`tar` does not follow symbolic links, but it rather writes the symbolic link as such into the `tar` file. This is different from the way the `cp` command works. If absolute paths are specified, `tar` writes absolute names into the `tar` file (see below).

If the verbose option `v` is specified when writing a tape, `tar` prints out (standard output) the character "`a`" ("archiving"), the file name, and the file size in blocks (or information on the symbolic link).

Tape Catalogs

Tape catalogs are obtained with the `t` option, for example,

```
tar tvf /dev/rmt/0
```

The blocking factor does not matter for reading tape catalogs. If the `v` option is also specified, the catalog contains not only the file names, but also information about the permissions, user-ID and group-ID of the file (both IDs in numeric form), the file size (in bytes), and the date of last modification (the date is given in an uniform format, unlike with the `ls` command). Directories are shown with file size 0. A typical `tar` listing (using the `v` option) is displayed as follows:

```
rwxr-xr-x 10/30    0 Jan 26 10:33 1993 bin/
rwxr-xr-x 10/30    0 Jan 25 09:32 1993 vnmrsys/
rwxrwxr-x 10/30    0 Jan 25 09:35 1993 vnmrsys/exp1
rw-rw-r-- 10/30   17 Jan 25 09:35 1993 vnmrsys/exp1/text
...
```

Reading a tape catalog takes considerable time. Because there is no directory, the entire file must be scanned. Therefore, it is advisable to make a catalog once and then to either print it out or (better yet) store that catalog on disk, see also "Verbose Option," page 175.

Reading Files

Reading from a `tar` file is done with the `x` option, for example,

```
tar xvf /dev/rmt/0
```

If the blocking factor specified for reading a tape is different from the factor that was used for writing, this can lead to error messages. To avoid errors, it is recommended to use the same blocking factor for reading as for writing or (better) not to specify the blocking factor when reading tapes.

If no file name is specified when reading from a `tar` file, all files are read back. If selective files should be extracted from the `tar` file, the complete file name must be specified, exactly the way it shows up in the output with the `t` or the `v` options. `tar` does not know wildcard characters for reading, and file names cannot be abbreviated. If a directory is specified, it is extracted with all its subfiles. If only a subdirectory or a subfile is extracted, `tar` automatically generates the necessary directory levels above it. If the file to be extracted already exists on the disk, the disk file is overwritten (provided the user has write permission on it).

If the files were stored with absolute paths, they are also read back the same way. Therefore, storing files with relative paths is highly recommended. Upon extracting, there is no way to redirect files that were stored with absolute path.

Verbose Option

The `tar` program is often used in the verbose mode (`v` option), which can generate considerable output. Running `tar` in a separate window is recommended. If the output is to be inspected during or after the execution of the command, you should either pipe the output to the more command, for example,

```
tar tvbf 40 /dev/rmt/0 | more
```

or use a Command Tool (scrollable window) for scrolling back later on.

Alternatively, the output can be dumped into a text file, for example,

```
tar tvf /dev/rmt/0 > tapecatalog
```

for easy and fast scanning of tape contents. Using the `grep` command, catalogs from dozens of tapes can be scanned for a specific file name in a fraction of a second.

Making Things Simpler

It is recommended that you use the environment variable `TAPE`. This allows you to leave out the `f` option in most cases. To simplify operation using `tar`, you can also define the following aliases in `.cshrc`:

```
alias twrite tar cvbf 40 /dev/rmt/0
alias tcat tar tvf /dev/rmt/0
alias tread tar xvf /dev/rmt/0
```

This makes the short commands `twrite`, `tcat`, and `tread` available for tape operations.

The Tape Tool in Open Windows allows to do `tar` read/write operations in an interactive way using a simple graphical interface that opens options such as `c`, `x`, `t`, `b`, and (mandatory) `f` for the source/target. The `f` option extents Tape Tool to a tool for transferring files also to floppy disks or other files (file systems).

Using tar to Transfer Directories

You can also use `tar` to copy directories (with subfiles) between file systems without using a tape (see the online manual or the Sun command reference manual). For example:

```
su
cd /export/home
mkdir vnmr.bk
cd /vnmr
tar cf - * | (cd /export/home/vnmr.bk; tar xvfBp -)
```

All files and directories (including subdirectories) in `/vnmr` are transferred to the standard output (`-`), then piped (`|`) to the standard input where a second `tar` process, which is started in `/export/home/vnmr.bk` beforehand, reads all incoming files and saves them there in the original form. The `B` option adjusts the blocking factor to the standard input, and the `p` option tries to keep ownerships, permissions, and modification dates the same as in the source files (the ownership part of this option only works if this is done as root). This sounds much the same as `cp`, but there is a subtle difference between the two commands—`cp` does not copy symbolic links but rather the file itself which the link is pointing at.

Using Multifile Tapes

`tar` automatically rewinds the tape at the end. Therefore, the `tar` command cannot be applied twice on the same tape without erasing the first data (only one write operation per tape is possible) unless special measures are taken.

A data segment generated with a single tape command can be skipped with a magnetic tape control command

```
mt -f /dev/rmt/0n fsf 1
```

where the `n` stands for no rewind and the `1` is the number of data segments to be skipped. `tar` can also be called with a “no rewind” option by using the device name `/dev/rmt/0n`, for example,

```
tar cvbf 40 /dev/rmt/0n *
```

But, in general, this practice is not recommended because the access to the second (or a further) data segment is rather awkward. A simple tape catalog (`tar tvf /dev/rmt/0`) only looks at the first data segment and does not indicate the existence of further data on the tape. For most read or catalog operations, the `mt` command has to be called for skipping the required number of data blocks.

Further instructions on how to handle multisegment tapes, and a shell script for obtaining complete multisegment tape catalogs were published by S. Patt and E. Williams of Varian, Palo Alto, in *Magnetic Moments*, Vol. IV, No. 1, pp. 14 -15. An updated version of that script is available with `bin/archive` from the VNMR user library.

Using tar Remotely

Frequently, a tape drive is only available on a remote system in the local network (see Figure 11).

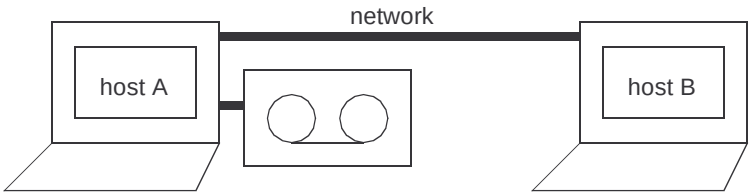

Figure 11. Local Network with Tape Drive

So how can you transfer data to or from this tape drive via the network without temporarily storing data on the disk of the tape server? The solution is to transfer the unextracted `tar` file via the network. On the system with the tape drive, you call `dd` to read or write the `tar` file to or from the tape drive and then you use a UNIX pipe to link the `dd` input (or output) with a `tar` command that is called on the system with the disk.

Reading Data from a Remote Tape Drive

Reading data from a remote tape drive can be initiated from both hosts involved. If you are working on host A, you can use the following command line:

```
dd if=/dev/rmt/0 ibs=20b | rsh B "(cd dir; tar xvf - files)"
```

to read the specified files, or

```
dd if=/dev/rmt/0 ibs=20b | rsh B "(cd dir; tar xvf -)"
```

if you want to read all files. The dd option ibs=20b (or whatever block size is selected) only needs to be specified if the tape was written with a nonstandard blocking factor, or when the blocking factor was specified when writing the tape). If you are logged into host B, you can use the commands

```
cd directory_name
rsh A dd if=/dev/rmt/0 ibs=20b | tar xvBfb - 20 files
```

(you are now executing the dd command remotely, rather than tar).

Writing Data to a Remote Tape Drive

Writing data to a remote tape drive can be performed from both hosts involved. If you are working on host A, you can use the following command line:

```
rsh B "(cd dir; tar cvf - files)" | dd of=/dev/rmt/0 obs=20b
```

It is not really necessary to specify the blocking factor (option obs=20b with the dd command), but if you do so, you need to specify the same blocking factor when reading that tape. If you are logged into host B, you can use the commands

```
cd directory_name
tar cvf - files | rsh A dd of=/dev/rmt/0 obs=20b
```

More information on the UNIX commands rsh and dd can be found in the UNIX manuals (use the man command).

15.2 ufsdump, ufsrestore, and dd Commands

The ufsdump, ufsrestore, and dd commands are used almost exclusively for backing up and restoring disks, which is part of the system administration. Chapter 6, “File Security, Repair, and Archiving,” discusses this in detail.

dd can also handle byte-swapping where necessary (on certain computer systems the byte-order is reversed as compared to the Sun).

Reading Gemini and VXR-4000 Tapes

dd is also a general-purpose command for reading tapes of any format, including the format used with other computer systems such as Gemini and VXR-4000 systems.

VXR-4000 tapes created with the command STAPE(WRITE,DSKn,...) consist of two files: a 1-sector (512 bytes) header file and the data file. Reading such tapes is as simple as skip one file and then read the next file. The VNMR software contains the tape utility that reads such tapes. Unfortunately, tape is very slow in reading 1/4-inch tape cassettes, because it reads the data sector by sector (like using tar with a blocking factor of 1). Therefore, the following procedure might be useful as an alternative:

`mt -f /dev/rmt/0lbn rewind`	Rewind tape
`mt -f /dev/rmt/0lbn fsf 1`	Skip tape header
`dd if=/dev/rmt/0lbn of=tape.288 ibs=512000`	Read the second tape file
`mt -f /dev/rmt/0lbn rewind`	Rewind tape
`decomp tape.288`	Decompose tape file

In this example, a blocking factor of 512000 bytes (1000 blocks) was selected. This size speeds up the reading operation dramatically.

The name of the output file can be selected freely except for the suffix, which must be equal to the maximum number of directory entries (I2 for the file FCB on the Gemini or VXR-4000). Otherwise, the `decomp` command, which splits up a VXR-4000 file into its subfiles, cannot work properly.

Disk units on the VXR-4000 always have 288 entries. Other directories (e.g. an automation file or user-created files) likely have a different number of entries. Of course, `tape` extracts that information from the tape header. If you are not sure about the number of directory entries, you might want to try out different values of the suffix:

`rm -r tape`	Remove previously decomposed file
`mv tape.288 tape.144`	Select a different suffix
`decomp tape.144`	Decompose again

You only need to try multiples of 36 as suffix. Because `decomp` does not alter the source file, you can try many times. The VNMR command `tape` has one advantage—it allows *selective* file extraction from a VXR-4000 tape, whereas with the above procedure, the entire tape (which can be as large as 16 MB) must be read onto the disk.

15.3 Using Floppy Disks

All desktop SPARCstation and Ultra models can be equipped with a 3.5-inch floppy disk drive that can read and write floppy disks in the DOS format (PCFS), UNIX file system format (UFS), or `tar` format. Again, for most operations, the Open Windows File Manager can handle the PCFS and UFS formats, and the Tape Tool the `tar` format.

This section describes these operations in detail, starting with UNIX commands and referring to the tools mentioned above.

Formatting Floppy Disks

Before they can be used under DOS or UNIX, floppies must be formatted into the UFS or DOS format. Note that the SPARCstation recognizes high-density floppies by the additional square hole at the rear edge of the floppy disk and refuses to format such floppies in low-density mode (of course, the opposite is also true). The raw size of formatted floppies is 1.44 MB for high density (with the extra hole in the corner) or 720 KB for double density.

To format a floppy, insert it into the drive (remove the write protection first by making sure the corner hole is covered). Then enter `fdformat` for a high-density floppy (1.44 MB) or `fdformat -l` for a low-density floppy (720 MB). The DOS format is provided if a `-d` option is used. To eject the floppy after formatting, add the `-e` option to the `fdformat` command or enter the command `eject` after the formatting is finished. The `-v` option causes `fdformat` to verify the floppy after formatting.

In case the floppy disk has a raw format, the above procedure is straightforward. If it has a DOS or UNIX file system format, the volume manager takes over the control and blocks any operation on a floppy from a shell. A superuser can kill the volume manager, format the disk, and can start the manager again (see “Floppies with the tar File System,” next). A more direct way is to add the `-U` option for `fdformat`, which unmounts any file system. This can be done by any user. To mount it again, eject the disk by the `eject` command and insert it again.

The File Manager also provides a format option. It turns out, however, that this tool cannot alter the format of a floppy disk. It denies reformatting a DOS floppy into a UNIX

formatted floppy disk, and vice versa. Formatting is only possible using an unformatted or `tar` formatted disk.

A trick overcomes this handicap. In most cases, a new floppy disk is preformatted with the DOS file system. If you want to use a UNIX file system instead, you can “unformat” the disk by destroying the format in the stray field of a supercon magnet. In such a case, hold the disks to the area with the strongest stray field at the bottom of the NMR magnet, and then remove it slowly while wagging it (flipping it over) rapidly. After this, the File Manager formats the disk.

Floppies with the tar File System

The most direct way to use a floppy disk is with the `tar` command. The disk must be preformatted (UNIX or DOS). In principle, there are the same options and arguments as for tape operations. Typically a command used has the form:

```
tar cvf /dev/diskette file1,file2,...
```

Instead of *`diskette`*, other names such as `floppy` or `rfd0a` ... `rfd0e` could be defined in `/dev` pointing to the same or similar floppy disk drives. The `r` stands for raw (no file system is expected), `fd` for floppy disk, and `a` through `e` determine which cylinders are used (see the UNIX manual). Using `diskette` is highly recommended, because it guarantees the best exploitation of the floppy disk.

The Tape Tool can also be used to store files on a floppy disk with `tar`. The operation is similar to that with a tape. However, the device name has to be changed to, for example, `/dev/diskette`.

To use Tape Tool or the `tar` command from a shell, you to get access to the floppy disk drive. It is usually occupied by the volume manager. Trying to execute `tar` results in an error message:

```
tar: /dev/diskette: Device busy.
```

To get access, you need to kill the `vold` process:

```
su root -c '/etc/init.d/volgmt stop'
```

With `start` instead of `stop`, the management is started back up again.

Alternatively, you can also just kill the process named “`vold`”. First, get the process-id:

`ps -ef | grep vold`	Gives 3-digit id-number of the process

and kill it as a superuser by *one* of the following:

`su; kill` *`id-number`*`;` [Ctrl-d]	Kill process in one step
`su root -c 'kill` *`id-number`*`'`	As `root`, kill the process

To have the volume management back (e.g., to run a floppy disk drive with File Manager) you must start it again as a superuser:

```
su root -c '/usr/sbin/vold &'
```

Floppies with DOS Format

For transferring data to a PC or a Macintosh, the floppy drive in desktop SPARCstations can also handle floppies with a DOS file system. Such floppies can be formatted on a PC or on the Sun. Use `fdformat -U -d` to format the floppy and also create a DOS file system on it.

The easiest way to handle DOS formatted floppy disks is via the File Manager of Open Windows. It automatically recognizes the format and transfers files between UNIX and DOS by mouse clicks.

For a manual mounting/unmounting you have to do the following:

To mount it, we create a directory `/pcfs` and add a line to the file `/etc/vfstab`:

```
/dev/diskette - /pcfs pcfs - no -
```

The floppy can now be mounted with the command

```
mount /pcfs
```

Any user can change directory into the new disk and create, copy, modify and delete files, directories and subfiles (a hierarchical file system).

To eject the floppy, you must first change the working directory (the floppy cannot be unmounted while somebody has their working directory in it), then unmount and eject the floppy with the commands:

```
umount /pcfs
eject
```

Unfortunately `mount` and `umount` can only be done by `root`, which is a security drawback with this method of using the floppy drive.

The DOS file system will have 1423 KB of free space (the boot block uses up some space). The floppy is optimized for space, *not speed*. To write a single 1-MB file onto the floppy can easily take over 10 minutes.

The DOS file system is also not optimized towards storing many tiny files. Trying to store all VNMR macros on a floppy stops after 225 files, because the directory space is exhausted (although only 250 KB are stored on the floppy). The disk directory is not allowed to contain more than 225 subfiles. All macros *can* be stored in a directory *within* the floppy, but it takes about 90 minutes to write 500 macros (compared to about a minute with `tar`), and even then it doesn't work properly, due to the file name restrictions, see below.

In some aspects, the DOS file system almost behaves like a UNIX file system, but nevertheless it has to follow DOS conventions: file names are truncated to 8 characters and any extension to 3 characters. Lowercase characters are mapped into uppercase (`ls` shows all names in lowercase). This is another severe limitation for any attempt to use a DOS floppy like a generic UNIX file system.

In addition to that, there are no permissions and no owner-ID or group-ID in a DOS directory, of course. `ls` shows all files with `rwxrwxrwx` permission, and all files are owned by the user `root` and the group `daemon`.

For transferring text files to a DOS format, you should make sure that the text itself fulfills DOS conventions, such as regarding `cr` and `lf` characters. The Sun software contains utilities to convert files in both directions:

```
unix2dos -ascii unix_file dos_file
dos2unix -ascii dos_file unix_file
```

For Sun/UNIX text file, it is advisable to use the `-ascii` option, and eventually also the -7 option, but this may need to be tried out, depending on the target application program on the PC or Macintosh.

Floppies with UNIX File System

Another possibility with floppy disks is to create a UNIX file system on them after formatting. This makes the floppy identical to the other disks in every aspect (except for

size and speed, of course). You gain a fully hierarchical file system, within which we can freely create, copy, delete, and modify files. There are also no DOS limitations. A large number of files can be stored without restrictions in the file names and with the full UNIX file protection scheme (permissions, ownership), the same as on a hard disk.

Not only do you gain additional flexibility, but also a great deal of performance over the DOS file system. When storing a plain 1-MB file *onto* the floppy, UNIX returns the control to you after just two to three seconds. The data transfer finishes in background after 50 seconds (UNIX data transfers *to* the disk are buffered; `tar` does it in 36 seconds, the DOS file system takes 11 minutes). Storing the VNMR macros or the entire `maclib` directory is no problem, although 500 macros take about 9 minutes to store (45 seconds with `tar`, about 90 minutes with the DOS file system).

The easiest way is to use the File Manager. The UNIX formatted disk is mounted automatically. The floppy disk can be used as any other disk. Any user is able to do this.

What is necessary to know about and what happens in the mount and unmount process?

This can be understood best going the manual way in a shell window while the Volume Manager `vold` is not running. (See above for how to stop `vold`).

The UNIX file system is generated with the command

```
 newfs /dev/diskette
```

(note that we have to specify a partition, not a disk). After this the disk can be mounted on an empty directory using the commands

```
mkdir /floppy
mount /dev/diskette /floppy
```

and now the floppy can be used. We can change a directory into it, create directories, store and edit files on it, like with any other file system. The only problem is that mounting can only be done by `root`, and before it can be ejected, it has to be unmounted again by `root`. Note that here the floppy also uses the standard UNIX permissions and ownership to protect the data, different from a DOS file system.

The following is a typical sequence of commands without the File Manager for using a new floppy containing a UNIX file system:

```
fdformat
newfs /dev/diskette
mount /dev/diskette /floppy
...
(use floppy)
...
umount /floppy
eject
```

As mentioned above, note that you have to be `root` to mount and unmount the disk, and also that unmounting is only possible if nobody is using the floppy as working directory. The `newfs` command gives the following output:

```
/dev/rdiskette: 2880 sectors in 80 cylinders of 2 tracks, 18
sectors
        1.4MB in 5 cyl groups (16 c/g, 0.28MB/g, 128 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 640, 1184, 1792, 2336,
```

If we use `df` to see how much space there is on the floppy, we get the following output:

```
/dev/diskette     1263      9     1134     1%      /floppy
```

Obviously, a part of the capacity of the disk has been lost in the creation of features necessary for the UNIX file system. The data are organized in 8-KB blocks, the disk is organized in 5 cylinder groups to make the file system fast, every cylinder group has a spare superblock for recovering from disk problems, and the disk reports 100% full when the filling degree is only 90%. We don't need all these features, because the floppy is slow anyway, and we don't need all the backup provided for hard disks.

To store more data on a floppy, use 4 KB instead of 8 KB block size, reduce the fragment size from 1 KB to 512 bytes, use 1 cylinder group of 80 sectors (covering the entire disk) only, and allow for 100% disk usage instead of 90%. The command:

```
varian# newfs -b 4096 -f 512 -m 0 -c 80 /dev/diskette
setting optimization for space with minfree less than 10%
newfs: /dev/rdiskette last mounted as /floppy
newfs: construct a new file system /dev/rdiskette: (y/n)? y
/dev/rdiskette: 2880 sectors in 80 cylinders of 2 tracks, 18
sectors
   1.4MB in 1 cyl groups (80 c/g, 1.41MB/g, 704 i/g)
super-block backups (for fsck -F ufs -o b=#) at
  32,
```

You could further increase the space by decreasing the number of I-nodes, but this may limit the number of files that can be put onto the floppy.

`newfs` also creates a directory `lost+found` in the new file system, for the file system check command `fsck` to store recovered files. If you don't use `fsck` on the floppy, you can also delete that directory. If `df` is now used to see how much space is on the floppy, you get the following output:

```
/dev/diskette       1323         4      1319     1%    /floppy
```

This has recovered more than 10% of disk space over the standard setup (we have only lost 10% of the `tar` floppy disk capacity. Note that although we have deliberately not optimized the file system for speed—it is still much faster than with the DOS file system. The above performance tests were performed with the space-optimized file system.

CAUTION: **Only experts should apply this or a similar procedures to improve the capacity or the performance of hard disk partitions. In general, it is not advisable to change the defaults for hard disks (in special cases, you can use `tunefs` for file system optimization either for speed or for capacity, see the Sun manuals or enter `man tunefs`).**

Of course, mounting can be simplified by creating an appropriate entry in `/etc/vfstab` (see "Automatic Mounting (/etc/vfstab)," page 146). For example:

```
/dev/diskette - /floppy  ufs - no -
```

Mounting can now be done by entering

```
mount /floppy
```

The above line in `/etc/vfstab` can coexist with the line for mounting a floppy with DOS file system, because nothing is mounted automatically.

Transferring Large Files Via Floppy Disks

Occasionally, there is a need to store and transport large files via floppy.

The `cpio` command opens up a possibility to store such large files on multiple floppies. In order to use `cpio`, you must first (as `root`) stop volume management using the command

```
/etc/init.d/volmgt stop
```

Now, you can insert a formatted floppy and enter:

```
echo filename.tar.Z | cpio -oc > /dev/rdiskette
```

After a while, the system ejects the floppy and you get the output

```
End of medium on "output".
To continue, type device/file name when ready.
```

At this point (and whenever this output turns up again), insert a new (formatted) floppy and type (but do not press Return because it will cause the command to abort):

```
/dev/rdiskette
```

At the end of the data transfer, the system reports the total number of blocks, and you can eject the last floppy with the command

```
eject diskette
```

For important files, it is advisable to make a second copy. At the very least, you should immediately verify whether the file can be read back completely, using the procedure given below. Finally, you can restart the volume management as `root`:

```
/etc/init.d/volmgt start
```

The procedure to read the file back from the floppies is similar (you again must disable the volume management for the duration of this task):

```
varian> cpio -ic < /dev/rdiskette
End of medium on "input".
To continue, type device/file name when ready.
/dev/rdiskette
End of medium on "input".
To continue, type device/file name when ready.
/dev/rdiskette
7671 blocks
   varian> eject diskette
   varian> ls -l
total 7696
-rw-r--r--  1 vnmr1  nmr 927375 Dec  5 12:00
filename.tar.Z
```

Reading bar Floppies

In SunOS 4.x, files could be written to floppy disks using the `bar` command. In contrast to `tar`, `bar` allowed adding files to an existing `bar` file on the floppy, and it also permitted storing big files on multiple floppies. `bar` is no longer supported in Solaris 2.x, and you should use `tar` or `cpio` (for large files, see above) instead. `cpio` also permits reading `bar` floppies that were generated under SunOS 4.x. To do this, you must first stop the volume management as `root`:

```
/etc/init.d/volmgt stop
```

Then you can insert and read the `bar` floppy, and then eject it using `eject`:

```
cpio -iH bar < /dev/rdiskette
eject diskette
```

After the `cpio` step, you can restart it again (as `root`) using

```
/etc/init.d/volmgt start
```

Chapter 16. Output Devices, Terminals, Serial Ports

Sections in this chapter:

- 16.1 “/vnmr/devicenames File,” this page
- 16.2 “/vnmr/devicetable File,” page 186
- 16.3 “Printing and Plotting under VnmrJ and VNMR,” page 187
- 16.4 “UNIX Aspects of Printing and Plotting,” page 188
- 16.5 “Null Modems for Printers and Plotters,” page 189
- 16.6 “Fixing Problems with Printing and Plotting,” page 190
- 16.7 “Connecting Serial Terminals,” page 190
- 16.8 “Using Extra Ports,” page 191
- 16.9 “Using an I/O Port for Acquisition Diagnostics,” page 191

This chapter describes the functionality of the VnmrJ, VNMR and UNIX printing facilities. In most cases, printer installation is achieved using easy-to-use graphical utilities, and *the detailed information provided in this chapter is usually not required for installing and using printers and plotters under UNIX, VnmrJ, and VNMR*. However, for debugging in the case of printing problems or for the installation in non-standard cases (printers on extra ports, non-standard printers, certain remote printers, even printing on non-standard paper formats), it is advantageous to know how to read and modify the administration files directly.

Most commands mentioned in this chapter must be performed by the UNIX system administrator `root`. Should you not have `root` permission on your system, you must involve the system administrator in these operations.

16.1 /vnmr/devicenames File

The text file `/vnmr/devicenames` defines printers and plotters for use with VnmrJ and VNMR. After installing both the Solaris OS and VnmrJ, use the VnmrJ adm interface to edit this file, refer to *VnmrJ Installation and Administration* manual section on adding printers and plotters. You can reuse the `devicenames` file from the previous version of VnmrJ or VNMR.

Even though you may not edit `devicenames` directly, it is sometimes useful to be able to be able to interpret the contents of this file in case of problems with printing or plotting. Each printer or plotter must be defined in this file with the following entries:

- `Name` – Can be anything. Value can be different from `Type`.
- `Usage` – Can be `Printer` or `Plotter`. Under Solaris 2.x `Both` should not be used.
- `Type` – A selection from a list given in the comment.

- Host – The name of the host to which the device is hooked up.
- Port – /dev/ttya or /dev/ttyb for devices connected to a serial port, /dev/bpp0 for the parallel port. Some network printers (printers connected to Ethernet) also use a pseudo port entry.
- Baud – 9600 or 19200 for most devices, unused for devices on the parallel port.
- Shared – Usually set to No, see comment section.

Any device may be defined several times under different names (e.g., HP7550 and hp), although usually it is preferable to use a single entry with a short name indicating what the device is (in networks, the name may also indicate where the device is located). Some devices might be defined with different characteristics (different resolutions for matrix printers, devices in "reversed" mode). Changing the printer or plotter by menu later is easier if a minimum number of options are activated.

For example, for a LaserJet printer (which can also be used as a plotter), define one entry (typically Laserjet_300R) for standard quality plotting (Usage: Plotter) and perhaps one other entry (typically LaserJet_150R) for rapid output (Usage: Plotter). Leave out the portrait format options for plotting because they are rarely used.

For printing, it is recommended to have the standard printer defined with the name lp, because that will make life easier later on (with a LaserJet, take any unused entry, name it lp, and set the usage to Printer). lp is the default device with the vnmrprint shell script. If lp is a defined printer, the printer name does not have to be specified when printing out files in UNIX using vnmrprint.

The end of the comment can be found easily in vi by searching for the expression COMMENTEND.

Even if you don't have a PostScript or HP/GL plotter, you may still want to have such devices defined in /vnmr/devicenames. This allows you to select such a device in VnmrJ and VNMR for plotting into a file. Use page(filename) in VnmrJ and VNMR, instead of just page, and make sure you don't incidentally send data to a non-existent printer!

16.2 /vnmr/devicetable File

The file /vnmr/devicetable defines the characteristics of all possible printers and plotters. It defines attributes such as steps per mm, character size, offsets, and drawing limits. If "artificial shoulders" are produced (perhaps by using extra-thin pens) with plotters, the number of steps per mm might be increased (e.g., change the parameter ppmm from 5 to 10) for smoother plotting output. In such a case, note that the character size (xcharp1 and ycharp1, in steps) must be increased accordingly; otherwise, text output shrinks in size.

The default character size can also be changed easily in the devicetable file. Make a backup copy of the original file before you start changing it. After changes are made to the devicetable file, the plotter must be selected again to enable the changes. This can be achieved by setting plotter=plotter in VNMR, which reenters the current plotter and causes VnmrJ and VNMR to reread the entry in /vnmr/devicetable.

Most entries in this file are quite obvious. The Printcap entry is not used with SVR4 (lp) printing. For VnmrJ and VNMR, the entry raster defines the type and orientation of the plot output: 0 is HP-GL output, 1 is PCL (raster, used on LaserJet and other dot matrix printers) in "portrait" orientation, 2 is PCL in "landscape" orientation, 3 is for PostScript in "portrait orientation, and 4 is PostScript in "landscape" orientation.

16.3 Printing and Plotting under VnmrJ and VNMR

VnmrJ and VNMR store information for printing and plotting in files in `/vnmr/tmp`.

Printing and Plotting Files

For plotting, the file name for such files is composed of four parts without a separator:

- Local host name
- Letters `PLT`.
- Process-ID of the Vnmr foreground process.
- Number that is incremented (starting at 0).

For printing, `PLT` is replaced by `PRINT`.

Some examples of file names:

```
uster1PLT88940
uster1PLT88941
...
uster1PLT889413
uster1PRINT88940
uster1PRINT88941
```

In all these examples, the host name is `uster1` and the VNMR process-ID was `8894`.

vnmrprint Shell Script

For printing, VnmrJ and VNMR do not send the files straight to the printer (by calling `lp <pr_file>`). Rather, it calls a shell script `/vnmr/bin/vnmrprint` as follows:

```
vnmrprint /vnmr/tmp/pr_file <printer_name> <file>
```

The `printer_name` argument could serve to do something specific to a certain output device type (as defined in `/vnmr/devicetable`) in the shell script. At the same time, this argument causes `vnmrprint` to delete the file after printing it. The `file` argument is used to indicate a disk file into which the output should be captured (rather than sending it to the printer). If the last argument is `clear`, this indicates that the accumulated printing information is to be discarded. VnmrJ and VNMR use this with the command `page('clear')`.

Let's have a look at the shell script `/vnmr/bin/vnmrprint` (actually a Bourne shell script, see "Bourne Shell Scripts," page 103). The central line in that file could be written as follows:

```
expand $1 | sed -e 's/^/    /' | fold | lp -s -o nobanner-d $2
```

In this part of the script, `$1` is the name of the file to be printed and `$2` is the name of the printer. The print data are sent through a number of separate programs before they are sent to the printer: `expand` replaces `tab` characters by spaces (8 spaces each—the printer does not "know" the standard UNIX tab settings), `sed` inserts 6 spaces at the beginning of each line (the "hat" sign "^" between the slashes indicates the beginning of each line).

The `fold` command in this script folds over lines longer than 80 characters (otherwise, characters behind column 80 would be lost with certain printers). The line lengths can be specified (e.g., using `fold -88`). With this setup, normal lines start at column 7 and the remainders of folded lines start at column 1. If all lines should start at column 7, the commands in the script would have to be rearranged as follows:

```
... | fold -74 | sed -e 's/^/     /' | ...
```

Note that `fold` now folds after column 74, because `sed` is going to insert blanks after folding.

Finally, `lp` sends the data to the output device (or, more correctly, to the spooling software). The `-s` option causes `lp` to suppress printing messages; the `-o nobanner` option suppresses the header ("burst") page.

The lines shown above from `/vnmr/bin/vnmrprint` appear four times:

- For the case where only the file name is given in the argument, a standard printer named `lp` is assumed, and the file is not deleted after printing.
- If two arguments are given, the file is sent to the specified printer and is not deleted after printing.
- If three arguments are given for other printers, the file is sent to the specified printer, and is deleted after printing.
- If four arguments are given, the print data are sent to a file (indicated in the fourth argument) instead of being plotted. If the fourth argument is `clear`, the data are discarded (no plot, no file).

The modes with two, three, or four arguments are the only ones used by VnmrJ and VNMR. The first mode, only a file name, can be used for calls from within a shell (e.g., a UNIX window). Make sure that from within VnmrJ and VNMR, `printon` `dg` `printoff` prints correctly. If you want to change `vnmrprint` for interactive use, you only need to change the first (if `lp` is a defined printer) or second occurrence of the line shown above.

Why is printing under VnmrJ and VNMR so complicated? If you send a print file directly to `lp`, you have no formatting control whatsoever (unless you explicitly pipe the data through a filter program). The idea behind `vnmrprint` is to have some minimal treatment that would ensure proper printing of unformatted documents but not severely affect the formatting of preformatted documents.

vnmrplot Shell Script

For plotting, a shell script `vnmrplot` is used instead of `vnmrprint`. `vnmrplot` contains a simple call to `lp` of the type

```
lp -s -c -o nobanner -d $2 $1
```

where `$2` is the name of the plotter, and `$1` is the file to plot. The `-o nobanner` option omits the header (banner) page, the `-s` option causes lp to suppress printing messages, and the `-c` option makes `lp` place a copy of the data in the spool directory, rather than just creating a symbolic link and plotting from the original file itself (in which case, it would be quite likely that `vnmrplot` would delete the file before plotting is finished).

Plot files can be quite large. A 5-MB file is not unusual for HP-GL or Postscript 2D plots with many signals and contour levels. Therefore, be sure there is plenty of free space in `/var` and `/tmp`. Similar to `vnmrprint`, an additional argument to `vnmrplot` permits "plotting into a disk file."

16.4 UNIX Aspects of Printing and Plotting

There are two aspects to using printers under UNIX: the user aspect (the actual print command, and commands to cancel print requests and check scheduled requests) and the aspect of printer setup and administration. Setting up new printers is covered in the *VnmrJ Installation and Administration* manual. In the following sections, we want to give you

some more insight in the Solaris printing software, to enable you to find and fix eventual printing problems.

How to Use Printers Under Solaris

The command to print under Solaris is `lp`. The data to be printed can either be specified as a file name:

```
lp -d printer_name filename
```

Alternatively you can use a UNIX pipe to print the output of a UNIX command:

```
ls -l | lp -d printer_name
```

This can also be used for formatting a text file prior to sending it to the printer:

```
cat filename | fmt -c -80 | lp -d printer_name
```

The `fmt` command is a simple text formatter.

If you have defined your printer as `lp`, then you don't need to specify the printer:

```
lp filename
cat filename | lp
```

Even if your printer is not named `lp`, you can still define it as default device to the `lp` command by defining an environment variable LPDEST:

```
setenv LPDEST printer_name
```

It is best to define this in your local `~/.login` file. Solaris print requests normally generate a leading header page that indicates who generated the following output, what the file name was (if the file name was specified as argument), date and time when the print request was submitted, and the print job number (see below). If you want to suppress this header page, you can use the `-o nobanner` option:

```
lp -d printer_name -o nobanner filename
```

or, when using a UNIX pipe:

```
cat filename | lp -d printer_name -o nobanner
```

The `lp` command reports the ID for your print request. For example,

```
> lp -d laser filename
request id is laser-23583 (1 file(s))
```

where `laser` is the name of the printer and `23583` is the ID of the print request.

The ID number can be used to cancel specific requests using the `cancel` command, if required (see `man cancel`). The same can be done more easily using the printer tool from the CDE toolbar.

How Does Solaris Printing Work?

In this section, which will be added in the next version of this manual, we plan to give you information on the setup files and the directories involved in the definition of the printers and in the spooling process. This might enable you to locate errors in the case of printing problems.

16.5 Null Modems for Printers and Plotters

Printers and plotters connected to port a or port b on the CPU board require a null modem; however, most RS-232 expansion boards can be configured so that no null modem is

required. Often, a null modem cable is used instead of null modem. We use “null modem” because what is commonly used for a null modem is a little box inserted between the cable and one of the connectors the cable connects to.

Although many output devices only use software handshaking (which only connects lines 2, 3, and 7 of the RS-232 cable), it is recommended that you use a proper, full RS-232 null modem (pins 1–>1, 2–>3, 3–>2, 4–>5, 5–>4, 6+8–>20, 7–>7, and 20–>6+8). A null modem can also be devised be rewiring one connector of a cable (clearly mark the cable in this case), but this is not a commonly recommended practice.

Note that the null modem described here is a standard version recommended by Sun for asynchronous serial communication with software and hardware handshaking. There are other standards on the market, such as for PC computers, that do not ensure proper hardware handshaking. It is essential to use a properly configured null modem.

16.6 Fixing Problems with Printing and Plotting

Hints for fixing printing problems will be added to this section in future version of this manual as we receive the information. Most operations must be performed as the UNIX system administrator `root`.

The first and most important hints are the following:

- If printing or plotting on a certain device does not work, it is usually best to delete the entire printer definition in both VnmrJ and Solaris, and to restart defining that printer from scratch.
- Follow exactly the installation instructions given in the manual *VnmrJ Installation and Administration*.
- If you are setting things up “by hand,” with PostScript printers it is absolutely essential to use *separate* definitions for printing and plotting.

16.7 Connecting Serial Terminals

With the advent of X terminals that communicate with the Sun workstation through Ethernet, simple serial terminals (such as GraphOn and Tektronics) has become far less popular. X terminals are not much more expensive than the serial models, and they offer far more comfort to the user as well as full-featured X Window system operation. X terminal emulators are also available for PCs and Macintoshes. More information will be added in a future version of this manual.

Currently, the main use of serial terminals is in the area of remote access through modems. See Chapter 13, “Other Networking Options,” for more information about this topic.

As with printers and plotters, terminals must be connected through a null modem (with some RS-232 expansion boards no null modem is required). The same type of cable or null modem can be used as for printers or plotters, see “Null Modems for Printers and Plotters,” page 189. Again (as for printers and plotters) you should preferably use a full null modem with the following connections: pin 1–>1, 2–>3, 3–>2, 4–>5, 5–>4, 6+8–>20, 7–>7, and 20–>6+8.

16.8 Using Extra Ports

Sun systems can be equipped with additional serial or parallel printer ports, typically by adding an SBus expansion board. For their installation and use, see the associated documentation. To use such a port under VnmrJ and VNMR, you may want to set up a printer using a standard port and then modify the port name in the newly generated Solaris printer control files.

16.9 Using an I/O Port for Acquisition Diagnostics

Sun and UNIX provide basic facilities to use a window as acquisition diagnostics terminal using any one of the RS-232 ports, involving the `tip` command and the `/etc/remote` file. At present, however, the result is relatively unsatisfactory because it causes CPU slow-down and other problems. This procedure is not recommended, certainly not as a long-term solution.

Chapter 17. X Window System Operation

Sections in this chapter:

- 17.1 “X Window System Environment,” this page
- 17.2 “Using a Workstation as an X Server,” page 193
- 17.3 “Using a Personal Computer as an X Server,” page 195
- 17.4 “Customizing the X Environment,” page 199

17.1 X Window System Environment

In the X framework, one computer, referred to in this chapter as the *X host*, is the computer that runs the software (this computer is also called the *X client*). Either the same computer or another computer acts as the *X server*, which is the computer on whose screen the graphics output of the client appears and whose mouse and keyboard are used for input. An X host can support multiple X servers, which means that a large number of users can be operating the software from multiple remote sites.

X servers can be a variety of things:

- One possibility is that the same computer that is the X host is also the X server. In this case, graphics are displayed on the screen of the same computer in which computations occur.
- A second possibility is that a second computer is used for this task. The second computer has to be equipped with X server software that allows it to function in this manner. If the second computer is a Macintosh or PC computer, X server software must be purchased to allow that computer to receive X output.
- The final possibility for an X server is a dedicated *X terminal*, a special-purpose terminal programmed to receive and interpret commands in the X language.

Note: Varian neither sells nor endorses X Window system products for the PC or Macintosh. Please contact the software companies selling these products or their dealers for information, purchase, and support.

17.2 Using a Workstation as an X Server

Setting Up a System

Of the several ways to configure your system, the following method is the simplest:

1. Enter `xhost +` on the workstation you will be using as an X server, which allows remote hosts to use your computer as an X server, or just enter `xhost` *`hostname`* to allow a single remote host to be used.

2. From an X environment (Motif, Open Windows, or whatever is available on your workstation), use the `rlogin` command to remotely log on to that computer.
3. You are asked about the type of terminal you are logging in from:
   `Choose x)window t)ek d)umb g)raphon [x/t/d/g(default]:`
   Enter `x`.
4. You are also asked to enter the name of your display server:
   `input display server name:`
   Enter the name of your computer.
5. Enter `vnmrj` to start VnmrJ (the X client) on the remote host and display information on your computer (the X server).

The `xhost` command did not execute correctly if you get messages such as the following on the local window:

```
Xlib: connection to "varian:0.0" refused by server.
Xlib: client is not authorized to connect to server.
```

Some general difficulties you may encounter have to do with the interpretation of different keys. When you are working on a remote computer in a general window (shell), the Delete, Backspace, and Ctrl-H keys may all function differently than you expect. If you make a mistake when typing, the one key combination that seems to work universally is Ctrl-U, which erases the entire line you have entered. Within VnmrJ itself, all keys should function normally.

An alternative method of running VnmrJ remotely is to execute a remote command string on the workstation in order to start VnmrJ, without ever logging in directly to that workstation. To do this, the file `.xlogin` is provided with your VnmrJ software to function as an alternative to the standard `.login` file. The `.xlogin` file sets all of the environmental variables required for proper functioning of VnmrJ. Therefore, a remote command string can be set up to execute both this file (`source .xlogin`) and the `vnmrj` command to start VnmrJ. This command string supplies the display variable as an argument:

```
rsh remotehost "source .xlogin; vnnmj -display xserver:0.0"
```

substituting computer names as appropriate, for example,

```
rsh u500 "source .xlogin; vnmrj -display ipx:0.0".
```

Using the `rsh` method of running VNMR remotely requires proper setup of UNIX files, including `/etc/hosts.equiv`. Enter `man rsh` from UNIX for more details.

Some specific difficulties, generally having to do with font availability, are described in subsequent sections of this chapter.

Using the RS/6000 as an X Server

IBM RS/6000 computers running AIX have a virtual terminal mode in which more than one screen (up to 16) can be made available and accessed successively by simultaneously pressing the Alt and Ctrl/Act keys. In this mode, it is essential to know the display number of the particular virtual terminal from which you are going to remotely activate VnmrJ running on another computer.

Before using `rlogin` to log in to the remote computer, but after starting the Motif Window Manager on the RS/6000, open a window and enter `env` to display a list of environmental variables. Look for the entry that reads `DISPLAY unix:n`, where `n` is a number from 1

to 16. Now enter `rlogin` as described above, and respond to the questions, but before starting VnmrJ, enter

`setenv DISPLAY` *`hostname`*`:`*`n`*`.0`

substituting the host name of your RS/6000 for *`hostname`* and setting the number *`n`* as appropriate (e.g., `setenv DISPLAY myibm:2.0`). This step is not necessary if `n=1`. Now start VnmrJ by entering `vnmrj`.

17.3 Using a Personal Computer as an X Server

Personal computers running a version of UNIX generally function similarly to that described above for workstations. The vast majority of personal computers are not running UNIX, however. Personal computers (Microsoft Windows based and Macintosh OS based computers) need X server software to function as an X server for VnmrJ running on a remote workstation. X server programs include MacX and eXodus for the Macintosh OS based PCs, and eXceed, PC-Xview, or ViewNow (ViewNow is incorporated in Windows 2000) for Microsoft Windows based PCs.

Within most X server software packages, two environments exist—rooted and rootless. The *rooted environment* means that all of the windows opened by VnmrJ (or other programs that you run) are contained within a single, larger window. By contrast, in a *rootless environment*, all of the X windows are separate windows. Both environments are acceptable, although you will probably find the rooted environment more convenient.

Starting Up VNMR Using a Personal Computer

Once you are running PC X server software on your personal computer, up to four VnmrJ start-up methods are available, depending upon the X server software:

- Remotely log in to the host computer using the telnet protocol, start the X session on the display server, start vnmrj in the X server display window.
- Execute a blind command string on the workstation to start VnmrJ. To do this, the dot file `.xlogin` is provided with your VnmrJ software as an alternative to the standard `.login` file. The `.xlogin` file sets all of the environmental variables required for proper functioning of VnmrJ. Thus, a remote command string can be set up to execute this file (`source .xlogin`) and then to execute the `vnmrj` command to start VnmrJ, supplying the display variable as an argument (e.g., `source .xlogin; vnmrj -display pcname:0.0`).
- Start a terminal window (`xterm`) using the blind command mechanism, and then, from that window, start VnmrJ.
- Start an `xterm` window; then start a remote window manager on the Sun. Next, start VnmrJ either with typed commands or with choices from a suitably configured menu.

Most X server software offers a choice between two command execution protocols—RSH and REXEC. These protocols differ only in password requirements—RSH does not require a password while REXEC does.

`xterm` operation requires the least overhead and setup. If VnmrJ is started from an `xterm` window, VnmrJ is displayed on the server with fixed windows. With the client windows manager running, the server screen looks like the screen on the Sun when `CDE` is running on the Sun, or it looks like the screen on the IBM RS/6000 when `mwm` is running, or like the screen on an SGI when a 4Dwm Window Manager is running.

Starting VnmrJ in either the client windows manager environment provides sizable and moveable windows. In the CDE environment, the VnmrJ windows can be collapsed to individual icons by pressing F7 on the server keyboard.

Starting X Client from a Telnet Session

The telnet protocol takes several steps, but it is useful during the initial set up of the server and client, and it provides confirmation of network communications between the systems.

Logging in and Starting a Telnet Session

If the X server software in not currently running on the display server system, start the X server software in the passive mode. Refer to the documentation provided with the X server software for setting up and optimizing the X server display options.

Starting the client using a telnet session, the X server is started in the passive mode with the X server software running on the server, awaiting instructions from the client. To communicate with the client and instruct it to display back onto the server, you open the communications to the client through a telnet session.

1. After you log in, the following is displayed on the system that will become the server:

   ```
   Choose x)window t)ek d)umb g)raphon [x/t/d/g(default)]:
   ```

2. Choose `x` to start the X environment. You are then prompted to enter the name of your X server:

   ```
   input display server name:
   ```

3. To start a client on a Sun and display a simple `dtterm` window on the server, the PC in this case, and using the Sun client as the window manager, enter the following command line on the client:

   `/usr/dt/bin/dtterm -display` *`servername`*`:0&`

   where *`servername`* is the name of your X server, 0 is the index of the X server session, and the `&` releases the telnet window without waiting for the application, `dtterm`, to terminate. The `dtterm` will display in the X server display window.

   A second `dtterm` can be started in a new X display on the same X server. Start the new X display session on the server. The X session window will display something similar to: `X Server display:1`. Note the number following **display:** and start the new `dtterm` as follows:

   `/usr/dt/bin/dtterm -display` *`servername`*`:<display number>&`

4. Start VnmrJ by entering **`vnmrj`** at the prompt in the `dtterm` window. If two `dtterm` windows were started, either window can be used. The other `dtterm` window will be free to start other applications.

At this point a `dtterm` window displays on the server. You can also edit the `.login` file to include the above instructions and create a new choice, `p`, for a PC X terminal login. The following is an example of such a modification:

1. Copy the `.login` file to a backup file (e.g., `bkup.login`).
2. Edit the `.login` file by first finding the following line:

   ```
   echo -n 'Choose x)window t)ek d)umb
   g)raphon[x/t/d/g(default)]:
   ```

3. Edit the line to include the following (keeping the entry on a single line):

   ```
   echo -n 'Choose x)window p)cXterm t)ek d)umb g)raphon  \
   ```

```
[x/p/t/d/g(default)]:
```

4. Locate the line:

   ```
   else if ($a = t) then
   ```

   Then insert the following lines above it (replace *servername* as appropriate):

   ```
   else if ($a = p) then
      set term=sun; setenv graphics sun
   ```

1. setenv DISPLAY *servername*:0

   ```
   /usr/dt/bin/dtterm&
   ```

The choice to log in as a PC and display the `xterm` window back on the PC is now available and is identified in the line `setenv DISPLAY servername:0.`

Starting the Client

If you have set up passwords and permissions, you can start the client and display the `xterm` window on the server using either the REXEC (you are prompted for a password) or the RSH (no password prompt—a trusted user login) protocol. In either case, the following instructions are required. These instructions are transmitted to the client by the server when the RSH or REXEC protocol is initiated. If you are using a Telnet session these instructions are entered in the telnet window.

To start a client on a Sun and display a `dtterm` or `xterm` window on the server, the PC in this case, the server instructs the user to enter the following command line on the client:

`/usr/dt/bin/dtterm -display servername:<number>`

or

`/usr/openwin/bin/xterm -display servername:<number>`

The `xterm` window can be further defined to include scroll bars, change the foreground and background colors, and so on.

VnmrJ is started from the command line in either the `dtterm` or `xterm` window.

The `dtterm` or `xterm` window, from which VnmrJ is started, is available to start other processes. For simple `dtterm` or `xterm` operations, this is not as useful as it appears. Since the windows cannot be moved or sized, the `xterm` window is covered by the VnmrJ window. If you choose to start a remote window manage.

VnmrJ can be started without first starting either a `dtterm` or `xterm` window from the command line in the telnet session (provided `/usr/dt/bin/` is in the path and the X server display number is 0) by entering `vnmrj` on the command line:

The effect is to log onto the client and be placed in the VnmrJ environment without any further effort and to start VnmrJ in the background.

A more useful approach, but one that has much greater overhead (provided`/usr/dt/bin/` is in the path), is the following:

`dtterm -display servername:<number> -e Xsession&`

This will start a full CDE desktop in the X Server display window. The dtterm window from which the CDE desktop can be reduced to an icon as it is no longer accessible - the Xsession is using it. Start applications and VnmrJ just as if you were working directly at the Sun. The extra overhead is not a problem if the systems are communicating via ethernet. If dial up communications are used, this is not a good choice. The important differences are:

- VnmrJ must be exited before logging off or closing the X session — and this is also true when you are working directly on the Sun.

- Exit from the CDE bar does not work — just close the X session.
- Solaris Snapshot does not work — use the X screen copy utilities supplied with the X server software.

Other operations, such as printing, are controlled from the Sun Solaris environment that was set up when VnmrJ was installed.

Controlling xterm Window Appearance

The `xterm` window can take on a number attributes depending upon your needs. The following examples of scripts can be made into executable files. As executable files, they can be run during the RSH or REXEC start up of the X server. If you use a telnet login, you do not need to change the scripts to executable files because `source .filename` always works (unlike RSH and REXEC).

In these examples, note that when a line needs to be broken in a script, a backslash (\) is placed at the end of each part of the line.

An xterm Window Started with a login Shell

```
goxterm -ls
#   Xterminal session on the PC
      #
      /usr/openwin/bin/xterm -sb -bg black -fg cyan \
      -bd blue -cr yellow -ms red -b 3 -bw 3 -ls \
      -display servername &
```

In this script, the `xterm` window has scroll bars (shown by `-sb`), black background (`-bg black`), cyan foreground (`-fg cyan`), blue border (`-bd blue`), yellow cursor (`-cr yellow`), red pointer (`-ms red`), internal border width of 3 pixels (`-b 3`) and window border width of 3 pixels (`-bw 3`) and begin a login shell (`-ls`). The `xterm` window is displayed on the X server in the background (`&`), `<servername> (-display <servername>:0&)`

Solving Networking Problems

Some networks use a host address file to list all the machines and their addresses on the network. Other networks use a server to store all the machine names and addresses (e.g., a domain name server). Problems, such as “unknown host,” which causes the server to terminate and not make the connection, can occur with the X server software.

If nominal login and network functions are working, such as dumb terminals `login`, `rcp`, and so on, and you are using a domain name server, the problem might be the inability of the X server to access the host names in the domain name server (assuming no errors in the configuration files for the X server host and client).

A solution to the unknown host problem is to create a host address file in the network directory that has the name and network address of the machine you are logging onto and as well and the name and address of the X server. For each machine you intend to start as an X client, you need that machine’s name and network address added to your host file.

```
!@(#)Qtune 1.2 10/25/9X
*QtuneAppDefaultsVersion: 1
*background: grey90
!*background: lightsteelblue1
! Note: inputFocusColor is different than background,
! but visually identical
*inputFocusColor: #e6e6e6
!*inputFocusColor: #cbe1ff
!
*TextField*inputFocusColor: black
*TextField*blinkRate: 0
*MessageWindow.background: white
*MessageWindow.linesVisible: 5
*MessageWindow.width: 250
*HelpWindow.background: white
*HelpWindow.linesVisible: 30
*HelpWindow.width: 600
*HelpWindow.font: 7x13
*AxisColor: green
*GridColor: forestgreen
*DataColor: cyan
*CursorColor: yellow
*Marker1Color: red
*Marker2Color: hotpink
*Marker3Color: magenta
*TuneDisplay.background: black
!*TuneDisplay.background: blue4
*DisplayGeometry: 800x500+4+25
*DisplayFont: 9x15bold
!*DisplayFont: 7x13bold
*TuneDisplay.geometry: 836x509+0+127
*Qtune.geometry: 297x509+845+127
*TuneCalibration.geometry: 318x156+824+665
*TuneHelp.geometry: 631x508+203+127
```

Figure 12. Example of `app-defaults` X Resource Setup File

17.4 Customizing the X Environment

VnmrJ and VNMR programs have X resource setup files in the `/vnmr/app-defaults` directory that define the parameters of the program windows. These parameters include colors of the objects in the windows, font size and style of the lettering, and the size and location of the window. The files use ASCII text to set the values of the parameters. Editing the files can be done by any UNIX text editor such as `vi` or `textedit`.

For example, Figure 12 shows the default X resource file for the `qtune` command. The full path of the file is `/vnmr/app-defaults/Qtune`. An exclamation mark (`!`) at the start of a line in the file indicates that the line is a comment. Because a comment line can be placed anywhere, a common use of the exclamation mark is to identify an alternate choice to the previous line, as in lines 3 and 4 of `Qtune`:

```
*background:  grey90
```

```
!*background  lightsteelblue1.
```

By simply changing the exclamation mark from line 4 to line 3, the background color is changed from `grey90` to `lightsteelblue1`.

System defaults file for the parameters are provided in `/vnmr/app-defaults` during installation. Users can override the system defaults for group or individual use by creating a similar directory in `~vnmr1` or `~user`.

Chapter 18. Periodic Maintenance and Troubleshooting

Sections in this chapter:

- 18.1 “Daily and Periodical Care of Computer Hardware,” this page
- 18.2 “Managing Free Space on Disks,” page 201
- 18.4 “Automatic File System Maintenance,” page 205
- 18.5 “Troubleshooting,” page 206

Although Sun systems require almost no periodic maintenance, there are several things you can do to keep your system running smoothly and reduce the chance of a breakdown. This chapter should help.

18.1 Daily and Periodical Care of Computer Hardware

Little hardware maintenance is required with Sun systems:

- CDE, automatically activates a screen blanker by default. Using `lockscreen` is only needed when you want password protection while the screen is locked. On SPARCstations and Ultra workstations, switching off the screen overnight (to save energy) is not a problem and is recommended.
- External tape drives and CD-ROM drives are normally used only from time to time. They should be switched off in-between to save energy and to stop dust collection through the cooling fan.
- Degauss color screens, as necessary, by switching it off and on again (monitors that are permanently exposed to strong magnetic fringe fields may require degaussing with an external degaussing coil).
- Clean the screen using a soft cloth and the appropriate cleaning fluid (see the manual for your monitor).
- Periodically check the fans (where applicable) and clean the air filters.
- Clean the head of the streaming magnetic tape drive periodically using a cleaning tape.

18.2 Managing Free Space on Disks

Software maintenance on a multiuser system primarily means file system discipline. Check the file systems periodically with the `df -k` command. For safety reasons, `df -k` reports “100% full” at the 90% level. “Completely full” is actually 111%. Whenever a file system is getting full, put unnecessary data files onto tape and/or remove them.

Use `du -sk *` to check for large files and directories (output is in kilobytes). When file systems are filled up, you most likely detect a drastic slowdown of most UNIX software. In severe cases, individual commands or even UNIX might crash or hang up.

Here are some other suggestions on maintaining free disk space:

- When programs crash (or similar serious things happen), UNIX usually dumps a `core` file (the current program-related contents of the RAM) into the present working directory for later debugging. Because no one is likely to be able to do such debugging, you can safely remove the `core` file, which might be more than 1 MB. Try to educate all users to work in their home directory, whenever possible. If they must temporarily work in some other directory, they should use `cd` to go back into their home directory. Automatic removal of core files can be done using `cron` (see below). A single line

  ```
  limit coredumpsize 0
  ```

  in `~/.cshrc` (also in `/.cshrc` for `root`) stops the system from dumping core files—a much better solution.
- As `root`, you can easily get the `root` partition "111% full" by submitting the wrong arguments to tape commands such as `tar` or `ufsdump`. When the system tries to write onto a nonexisting tape device (e.g., using `tar` as `root` and incorrectly typing `/dev/rmt/9mb` instead of `/dev/rmt/0mb`, or typing a correct device but having it not configured), the data actually ends up in the corresponding file in `/dev/rmt`. There is normally only limited free space on the root partition; therefore, it is easy to fill it up. Only limited operations are possible with a full root partition, but file systems can still be inspected and files can be deleted (if not in multiuser mode, then in the single-user mode). To fix the problem, remove unwanted files. To find those or other erroneous files, enter `ls -l /dev/rmt|more`. All correct files should be symbolic links to `/devices`, so erroneous additional files can be easily recognized.

Of course, software maintenance also means maintaining software security by having files backed up on tape at the appropriate intervals. Use passwords to protect your software and data from hackers and others that could destroy or alter it.

18.3 Shutting Down and Restarting UNIX

The following sections illustrate the safest methods for shutting down UNIX in an orderly fashion so as to minimize the risk of corrupting hard disk data. It is not advisable to shut down UNIX unless absolutely necessary, for example, system maintenance or repair, planned power removal, or moving the system.

CAUTION: **If proper shutdown procedures are not followed, system and user files stored on hard disks can be corrupted.**

Normal UNIX Shut Down

To shut down the system, you must be logged in as `root` from the `login`: prompt or you must have entered `su` command (not from a shell).

If you are currently in VnmrJ or VNMR, exit the program first before starting here.

In the procedure below, entering the `killacqproc` command is not needed if you are only rebooting the system; the reboot will kill `Acqproc`. Also, step 2 to rename the `acqpresent` file is only needed if `Acqproc` is not wanted when rebooting; for instance, when the NMR console will not be used. These instructions assume the user acqproc has not been created (refer to the VnmrJ System Administration manual for creating the user acqproc). If the user acqproc is created, all users can shutdown the acquisition demons by typing **`su acqproc`** in a terminal window. The acqproc user will execute step 1 through step 2 automatically.

1. Log in as root, enter the root password (if implemented), then enter the following commands to stop the Acqproc process:

   ```
   # cd /vnmr/acqbin
   # ./killacqproc
   ```

   The system communicates success by displaying:

   ```
   Acqproc killed
   Acquisition daemon terminated
   ```

   If both of these messages appear, it is safe to go to step 2.

   If *both* of the messages do not appear, repeat step 1. If both of the messages do not appear after a second attempt to stop the Acqproc process, take the following steps, as root:

   a. Enter the ps command to show all active processes and (optionally) the grep command to show only the lines containing "Acq":

      ```
      # ps -ef |grep Acq
      ```

      The system displays a table with five columns. Look at the entries in the COMMAND column (the last column) until you find an "Acqproc" entry. Then look at the number in the PID column (the first column) for the row containing "Acqproc."

   b. On systems other than UNITY*INOVA*, enter the following command, except replace PID_number with the number you found:

      ```
      # kill -9 PID_number
      ```

      For UNITY*INOVA*, it is Expproc, not Acqproc, that must be killed. Instead of the last command enter:

      ```
      # kill -2 PID_number
      ```

      When Expproc is killed this way, by using -2, the complete Proc family (Expproc, Recvproc, etc.) is stopped.

   c. Repeat the ps command you entered in step 1a and look again in the COMMAND column. "Acqproc" should be absent. If it still appears, repeat the kill command in step 1b. When "Acqproc" no longer appears, the Acqproc process has been stopped, and it is then safe to proceed.

2. Rename the file acqpresent as follows:

   ```
   # cd /etc
   # mv acqpresent acqpresent.old
   ```

3. Shut down:

   - immediately, enter:

   ```
   # init 0
   ```

   - Shut down at the given time:

   ```
   #/etc/shutdown -y -gsec -i0 "optional text message"
   ```

   sec is the number of seconds (default is 60 seconds), for example, for 5 minutes to shutdown, enter g300

   For more information on shutdown and init , see Solaris documentation or enter either man shutdown or man init.

   At shutdown, the system forces any information on its way to the hard disk to be written out immediately, cleans up any processes that are running, and executes an orderly shutdown of the system. The process takes about 20 seconds. When the system has safely shut down, the prompt ">" or "OK" appears.

4. It is now safe to turn off the system power. Turn off the power switches in this order:

a. Hard disk(s) and tape drive(s) (if drives have separate power switches).
b. CPU unit.
c. Monitor.

Shutting Down UNIX in an Emergency

If no one can log on as `root` (e.g., the password is lost), and you absolutely must reboot or remove power from the UNIX system, it is necessary to run the risk of corrupting files and possibly losing data. The following procedure is not safe, but in an emergency it is marginally better than simply turning the power off.

Preferable method:

1. Exit VnmrJ or VNMR.
2. Open a terminal window and become the root user.
3. Enter:
   ```
   > init 0
   ```

Another method, if needed is:

1. Exit VnmrJ or VNMR.
2. Open a terminal window and become the root user.
3. Enter:
   ```
   > sync;sync;sync
   ```
4. Press Stop-a (hold down the Stop key and press the a key). On older keyboards, the Stop key is labeled L1, so press L1-a instead.

The `sync;sync;sync` command attempts to ensure that all disk writes are completed but does not do it as thoroughly as the `shutdown` command. The L1-a key combination is a system reset that should be used only in an emergency when nothing else works. Files can be lost using the L1-a reset.

Restarting UNIX

Start with step 1 below if restarting (rebooting) UNIX after power has been removed from one or more system components. If no units are powered down, you can skip steps 1 and 2, and restart UNIX from the boot prompt "`>`" by typing the command `b` and following the instructions in steps 3 and 4 below. From the "`ok`" prompt, type `boot`.

You do not need to perform step 4 in this procedure if you did *not* rename the `acqpresent` file when shutting down.

1. After the ac power cables have been reconnected (all Sun Ultra and newer systems), turn on the power switches to each component of the system in this order:
   - CPU unit
   - Hard disk(s) and tape drive(s) (if the drives have separate power switches)
2. Upon turning on the CPU unit, the system performs a standard self-test, then executes an automatic boot procedure.
3. When the system has successfully booted, the `login` prompt or CDE login screen appears.
4. Do either of the following:

- If the user acqproc exists:

a. Log in as any user (normally vnmr1 user will be restarting the system)

b. Open a terminal window.

c. Enter **su acqproc**

- If the user acqproc does not exist:

a. Log in as root and enter the root password (if implemented).

If you renamed the acqpresent file when the system was shut down, enter the following commands to rename acqpresent and execute startacqproc:

```
# cd /etc
# mv acqpresent.old acqpresent
# cd /vnmr/acqbin
```

b. Execute startacqproc:

```
# ./startacqproc
```

c. If you want to log in as another user (e.g., user vnmr1, the NMR system administrator), enter the logout command, then log in using another valid user name and enter the user password (if implemented):

```
# logout
login: username
Password: user_password
```

18.4 Automatic File System Maintenance

The useful UNIX feature cron can be used to periodically search the file system for obsolete files and reduce chronically growing files to a reasonable size. cron looks up a file /var/spool/cron/crontabs/root every minute. Periodic actions can be specified to be performed at a certain minute of the hour, a certain time of the day, a certain day of the week, and so on.

To find and remove core files from the /home directory on a weekly basis, such as every Saturday at midnight, the following crontab file entry may be used:

```
0 0 * * 6 find /export/home -name core -exec rm {} \;
```

The first 0 stands for the minute, the second 0 for the hour, the third entry is the day of the month (* indicates any), the fourth entry is the month, the next entry stands for the day of the week (0 is Sunday, 1 is Monday, etc.).

Every user can create a separate crontab file. Such a file is created and edited with the command crontab -e. A printout (without editing) can be obtained by crontab -l. Never try to edit the crontab file directly, always use the crontab command because otherwise the new entry is not activated.

A shell uses the uncommon line editor ed as default for crontab -e. To get the vi editor (as superuser), enter the command EDITOR=vi; export EDITOR before executing crontab -e. For a superuser, this can be made permanent by putting those commands into the file .profile under the directory /. For any other user, put the command setenv EDITOR vi into the .login file of the user's home directory.

18.5 Troubleshooting

This section presents some general troubleshooting help including acquisition status codes, Ethernet network problems, system hanging, and other topics. Be sure to contact Varian service for expert maintenance and repair of your system.

Acquisition Status Codes

Whenever `wbs`, `wnt`, `wexp`, or `werr` processing occurs, the acquisition condition that initiated that processing is available from the parameter `acqstatus`. This acquisition condition is represented by two numbers, a "done" code and an "error" code. The done code is set in `acqstatus[1]` and the error code is set in `acqstatus[2]`. Macros can take different actions depending on the acquisition condition.

The done codes and error codes are listed below and in the file `acq_errors` in `/vnmr/manual`. For example, a `werr` command could specify special processing if the maximum number of transients is accumulated. The appropriate test would be:

```
if (acqstatus[2] = 200) then
"do special processing, e.g. dp='y' au"
endif
```

These codes apply to all systems, except codes marked with an asterisk (*) are not used on *MERCURY* and *GEMINI 2000* systems.

Done codes:

11. FID complete
12. Block size complete (error code indicates `bs` number completed)
13. Soft error
14. Warning
15. Hard error
16. Experiment aborted
17. Setup completed (error code indicates type of setup completed)
101. Experiment complete
102. Experiment started

Error codes:

Warnings

101. Low-noise signal
102. High-noise signal
103. ADC overflow occurred
104. Receiver overflow occurred*

Soft errors

200. Maximum transient completed for single precision data
201. Lost lock during experiment (LOCKLOST)

300. *Spinner errors*:
301. Sample fails to spin after 3 attempts to reposition (BUMPFAIL)
302. Spinner did not regulate in the allowed time period (RSPINFAIL)*
303. Spinner went out of regulation during experiment (SPINOUT)*
395. Unknown spinner device specified (SPINUNKNOWN)*
396. Spinner device is not powered up (SPINNOPOWER)*
397. RS-232 cable not connected from console to spinner (SPINRS232)*
398. Spinner does not acknowledge commands (SPINTIMEOUT)*

400. *VT (variable temperature) errors:*
400. VT did not regulate in the given time `vttime` after being set
401. VT went out of regulation during the experiment (VTOUT)
402. VT in manual mode after auto command (see Oxford manual)*
403. VT safety sensor has reached limit (see Oxford manual)*
404. VT cannot turn on cooling gas (see Oxford manual)*
405. VT main sensor on bottom limit (see Oxford manual)*
406. VT main sensor on top limit (see Oxford manual)*
407. VT sc/ss error (see Oxford manual)*
408. VT oc/ss error (see Oxford manual)*
495. Unknown VT device specified (VTUNKNOWN)*
496. VT device not powered up (VTNOPOWER)*
497. RS-232 cable not connected between console and VT (VTRS232)*
498. VT does not acknowledge commands (VTTIMEOUT)

500. *Sample changer errors:*
501. Sample changer has no sample to retrieve
502. Sample changer arm unable to move up during retrieve
503. Sample changer arm unable to move down during retrieve
504. Sample changer arm unable to move sideways during retrieve
505. Invalid sample number during retrieve
506. Invalid temperature during retrieve
507. Gripper abort during retrieve
508. Sample out of range during automatic retrieve
509. Illegal command character during retrieve*
510. Robot arm failed to find home position during retrieve*
511. Sample tray size is not consistent*
512. Sample changer power failure during retrieve*
513. Illegal sample changer command during retrieve*
514. Gripper failed to open during retrieve*
515. Air supply to sample changer failed during retrieve*
525. Tried to insert invalid sample number*
526. Invalid temperature during sample changer insert*
527. Gripper abort during insert*
528. Sample out of range during automatic insert
529. Illegal command character during insert*
530. Robot arm failed to find home position during insert*
531. Sample tray size is not consistent*
532. Sample changer power failure during insert*
533. Illegal sample changer command during insert*
534. Gripper failed to open during insert*
535. Air supply to sample changer failed during insert*
593. Failed to remove sample from magnet*
594. Sample failed to spin after automatic insert
595. Sample failed to insert properly
596. Sample changer not turned on

597. Sample changer not connected to RS-232 interface
598. Sample changer not responding*

600. *Shimming errors:*
601. Shimming user aborted*
602. Lost lock while shimming*
604. Lock saturation while shimming*
608. A shim coil DAC limit hit while shimming*

700. *Autolock errors:*
701. User aborted (ALKABORT)*
702. Autolock failure in finding resonance of sample (ALKRESFAIL)
703. Autolock failure in lock power adjustment (ALKPOWERFAIL)*
704. Autolock failure in lock phase adjustment (ALKPHASFAIL)*
705. Autolock failure, lost in final gain adjustment (ALKGAINFAIL)*

800. *Autogain errors*.
801. Autogain failure, gain driven to 0, reduce `pw` (AGAINFAIL)

Hard errors
901. Incorrect PSG version for acquisition
902. Sum-to-memory error, number of points acquired not equal to `np`
903. FIFO underflow error (a delay too small?)*
904. Requested number of data points (`np`) too large for acquisition*
905. Acquisition bus trap (experiment may be lost)*

1000. *SCSI errors:*
1001. Recoverable SCSI read transfer from console*
1002. Recoverable SCSI write transfer from console**
1003. Unrecoverable SCSI read transfer error*
1004. Unrecoverable SCSI write transfer error*

1100. *Host disk errors:*
1101. Error opening disk file (probably a UNIX permission problem)*
1102. Error on closing disk file*
1103. Error on reading from disk file*
1104. Error on writing to disk file*

Display Hardware Console Status (UNITY*INOVA*)

Use the UNIX command `ihwinfo` to display the status of digital hardware in the UNITY*INOVA* console. The command

```
ihwinfo startup
```

displays the status at the conclusion of the last console startup (powerup, reboot, etc.). The command

```
ihwinfo abort
```

displays the status the last time an acquisition was aborted or the console rebooted from the host computer (`abortallacqs`). In this context, exiting from either the FID display or lock display of `acqi` counts as an abort. Only the status from the last abort can be displayed.

Global Parameter File Corrupted

Symptoms that the global parameter file (`~/vnmrsys/global`) is corrupted include messages such as the following:

- `variable printer not found`
- `variable wcmax not found`
- `variable lastmenu not found`

The most frequent cause for this problem is probably a full disk when exiting VnmrJ or VNMR, but it is also caused by killing the application from exiting the window environment directly.

To correct the problem:

1. Exit VnmrJ or VNMR.
2. Enter:
   `cd /vnmr/user_templates/global ~/vnmrsys`
3. Restart VnmrJ or VNMR .
4. Reselect your printer and plotter.

To avoid global parameter file corruption in the future, follow these precautions:

- Always exit from VnmrJ and VNMR before exiting from the window environment.
- When the disk is full, do not exit VNMR but free up some disk space first.
- Always exit VnmrJ or VNMR by using the Exit VnmrJ menu selection or with VNMR the Exit VNMR button or by using the exit command in VNMR.
- Do not start VnmrJ or VNMR with a full `/home` (or `/export/home`) partition or with a full user disk quota.

Insufficient Swap Space

If the swap space on the host computer is set too low, processes such as PSG may not find sufficient space in which to operate. For proper VnmrJ or VNMR operation, the disk should have been partitioned with swap space equal to twice the computer memory. Enter the command `swap -l` to check the amount of free swap space. If the proper amount was not partitioned, you can increase the available swap space, as follows:

1. Become `root`.
2. Use the `mkfile` command to make a new file for the additional swap space. The size, name, and location of this file are up to you (e.g., for a 16-MB file named `swap` in `/export/home`, enter `mkfile 16m /export/home/swap`).
3. Enter `swap -a /export/home/swap` to add the file to the available swap space.
4. To make the new swap space permanent so that it reappears when the computer is rebooted, edit the file `/etc/vfstab` and add the following line:
   `/export/home/swap - - swap - no -`
   where the first entry is the name of the file created in step 2 above.

To remove a swap space, use the `swap -d` command to release the swap space, followed by the `rm` command to remove the swap file.

tmp Partition Fills Up

If the /tmp partition fills up, it can cause problems for VnmrJ and VNMR users. One solution is to move the tmp partition. Here is one method for doing so, which assumes that you have lots of free space in a different partition (/data in this example).

1. Make sure there is no user activity on the system.
2. Enter:
```
su
cd /
tar cf - tmp | (cd /data; tar xfBp -)
rm -r tmp; ln -s /data/tmp tmp
```
3. Reboot UNIX.

Password Invalid

A password may be invalid because it has expired. Solaris 2.x and newer has a password aging mechanism that disables passwords after a predefined time. This security feature can be activated to force users to periodically change their passwords. root can turn on password aging for an existing user (like vnmr1) by entering:

```
passwd -n 7 -x 1000 -w 30 vnmr1
```

This sets the time between password changes by vnmr1 to 7 days, the time the password is valid to 1000 days, and the time when warning messages start on login to 30 days prior to the expiration date. After expiration, the vnmr1 password becomes invalid and can only be reset by root using the command

```
passwd vnmr1
```

Password aging can also be entered by root through the admintool program.

While password aging may be desirable for standard users, especially in larger networks, it is not appropriate for users such as acqproc that usually have no password (even users with no password can become invalid if password aging is turned on). Should password aging be inadvertently turned on, commands such as su acqproc may become invalid.

To tell the system that a user (such as acqproc) will never again need a password, root should enter:

```
passwd -d acqproc
```

To disable password aging for any user after VnmrJ is installed, root should enter:

```
passwd -x -1 acqproc.
```

Password aging for root should be used with care—if the root password becomes invalid, you may need to reload UNIX.

System Hangs

System “hangs” are most likely to occur when several shells are opened at the same time and then are not closed or exited properly or if the memory buffer for a window (such as the dg window) is exceeded with a lengthy file (e.g., a procpar file). At that point, nothing on the screen appears to be active—the mouse buttons does not work and commands can not be entered. Some suggestions:

1. Try resetting the console with the reset button inside the console.
2. Try holding down the Stop key on the keyboard (labeled L1 on some keyboards).

3. If another workstation is available and networked with the host, try a remote login (`rlogin`) to the host, identify the hung process (using `ps -ef` in Solaris), and then kill that process.
4. When all else fails, hold down Stop (or L1) and then press the A key.

   Pressing Stop-A should not be used more than necessary because of the chance of corrupting data and files on the disk drive (the CPU is constantly accessing and monitoring the disk drive, and the Stop-A key combination could halt processing while a file was being looked at, thereby damaging the file).

   After pressing Stop-A, wait for the boot prompt to appear, reset the console, and then power up the system.

Changing the Name of a Solaris Host

If you need to change the hostname of a computer using Solaris software, the easiest way to proceed is to enter the command `/usr/sbin/sys-unconfig`. Rebooting after this then asks all the questions that were asked at startup about the hostname, IP address, etc. This process removes the `hosts` file, so you should have a copy or be prepared to enter all the information again.

Ethernet Network Troubleshooting

Troubleshooting problems on the Ethernet network involves determining whether the problem lies within the hardware or the software. Try to check for the most obvious problems first (such as cables properly connected and boards properly seated) and then proceed to eliminate factors indicated by suspicious symptoms.

Error Messages

“ie no carrier”
“ie cable problem, Requesting Internet address for 8:20:0:n:n:n”
“Connection timed out”
“? possible cable problem”

Check that the transceiver cable is properly connected to the machine.

Check that the Ethernet Controller board (if applicable) is properly seated.

Check the connections at the multiplexer box (if applicable).

Run the `netstat` command to get packet collision information (see “Commands for troubleshooting the network” below).

“ie ethernet jammed”
“Ethernet transmission error”

Check that the Ethernet cable is terminated at both ends with a 50-ohm terminator.

“machine_name not in hosts database”

Check the `/etc/hosts` file to see if the system name is properly listed along with its complete Internet Protocol (IP) address.

Check that the “Network Information Services” (NIS) database has been updated on the Master NIS file server system and that the associated processes in */etc* are running. To find the name of your NIS server, enter `ypwhich`. To find your nodename in the NIS database, enter `ypcat hosts | grep 'name –n'`.

Find the host system by running the `ping` command (see "Commands for troubleshooting the network" below).

Check the connection with the host system by running the `rlogin` command (see "Troubleshooting commands" below).

"x connection OK - but possible file problem"

Check the `/etc/hosts` file to see if the system name is properly listed along with its complete Internet Protocol (IP) address.

Check `/etc/nd.local` file for proper root and swap entries.

Check `/etc/ethers` file for correct Ethernet address of client machines.

No message or solutions above do not work

Check the `/etc/rc.boot` file for the proper `/etc/ifconfig` statements (without comments). An extra line is needed for a second Ethernet controller on a "gateway" machine.

On Solaris, check `/etc/defaultdomain` and `/var/yp/binding`.

Check that the proper Ethernet controller exists in the system kernel configuration file.

Swap transceiver boxes to determine if the existing boxes might be defective.

Commands for Troubleshooting the Network

The following commands for troubleshooting over the network can be executed when logged in as `root` ("#" prompt), as any VNMR user (">" prompt), or as any other user ("%" prompt).

- To find the host system, enter the `ping` command with the name of the host system you want to check replacing the `host` argument to `ping` (from Solaris only):

  ```
  % /usr/sbin/ping <host>
  ```

  If the program does not find the host system, it displays the message:

  ```
  unknown host <host>
  ```

  For more information, refer to the Sun manual *Administration* (it may refer to the host system as the "machine_name").

- To get packet collision information, use the `netstat` command:

  ```
  % netstat -i
  ```

  A large number of collisions might indicate some kind of cable problem. Refer to Sun manual *Administration*.

- Check the connection by entering the `rlogin` command with the name of the host system you want to join replacing the `host` argument:

  ```
  % rlogin <host>
  ```

  If the connection is not completed, the system might return one of the following error messages:

  ```
  connection refused
  <host>: unknown host
  ```

Console Problems

On UNITY*INOVA* systems, enter the UNIX command `showconsole` to display the console hardware configuration parameters and system versions. Parameter and system version information, which is recorded when the console is booted up, represents the system

hardware options recognized by the acquisition computer. `showconsole` is used mainly during troubleshooting and diagnostics.

System Failure

Perform the following corrective actions in case of a system failure. *In the event of an emergency, contact your system administrator.*

Basic Corrective Actions

- *Check spectral window* – Is the full spectrum displayed, or off-center? Check `tof`, `sw`, `np`, `fn`, enter `ds` and click on the Full button. Enter `full`.
- *Check pulse parameters* – Enter `dps` to display pulse sequence. Are the values reasonable? Check `pw`, `p1`, `tpwr`, `gain`, `nt`, `ss`, and `d1`. Which pulse sequence are you running? Is your pulse sequence compiled?
- *Standard parameters* – If you are having problems with parameters, retrieve the parameters or pulse sequence from a standard source.
- *Cancel command* – Is a command running that has not finished yet? Wait for the command to finish, or click on the Cancel Command button. If the command is a macro, which version of the macro (`vnmrsys/maclib` or `/vnmr/maclib`) are you using?
- *Abort acquisition* – Is an acquisition running in the experiment? To abort the acquisition, enter `aa`. Are there other experiments queued up? Enter `abortallacqs` to abort all acquisitions.
- *Check the probe* – Is the probe tuned? Is the pulse length calibrated?
- *Check hardware* – Are the cables configured correctly? Is the sample seated correctly in the probe? Is the probe seated correctly? Is the VT unit regulating? Is the sample spinning? Is the sample shimmed?
- *Check hardware configuration* – Enter `config`. Check the amplifier types and gradient types.

Advanced Corrective Actions

- *Restart acqproc* – If the acquisition status window shows the Status as Inactive and not Idle, `acqproc` needs to be restarted. In a shell window, enter `su acqproc`. If the message says “Acquisition Terminated,” enter `su acqproc` again. Or become `root`, enter `cd /vnmr/acqbin`, then enter `./killacqproc` or `./startacqproc`, as appropriate.
- *Exit* VnmrJ or VNMR – If you are having persistent VnmrJ or VNMR command problems, exit VnmrJ or VNMR. Restart VnmrJ or VNMR . How much disk space do you have?
- *Log out* – If you are having persistent Sun or UNIX command problems, log out, and log back in.
- *RESET button* – If the acquisition has problems, open the console housing door and press the small red RESET button. You do not need to exit VnmrJ or VNMR before performing this action.
- *Power cycle* – If the acquisition cannot be restarted, power cycle the spectrometer by switching off the main power supply. Wait one minute for all devices to stop. Then switch on the main power supply. Wait for the system to come back up. If you power

cycle the computer and peripheral devices, be sure to switch them on and off in the correct order.

Color Flashing Problem with Adobe Acrobat Reader

Color flashing can occur if *VNMR Online* (`vnmr_ihelp`) is started before VnmrJ or VNMR. Adobe Acrobat Reader, by default, allocates 135 colors in the color map, which does not leave enough colors for VnmrJ or VNMR.

To prevent colormap problems, edit the file `/vnmr/acrobat/sol/sparcsolaris/app-defaults/Acroread` by inserting the following line:

```
*targetColorCube:3
```

This line instructs Acrobat Reader to reduce the default number of colormap entries to 35.

Appendix A. Hardware Reference Information

Sections in this appendix:

- 18.6 “Activating a Second Hard Disk After Solaris is Installed,” page 215
- 18.7 “Active SCSI Termination,” page 220

18.6 Activating a Second Hard Disk After Solaris is Installed

This section describes how to add a second hard disk (external) after Solaris is already installed. If both hard disks were configured when Solaris was installed, skip this section. If only one hard disk was configured when Solaris was installed and you want to add a second hard disk, use the procedure below. If VnmrJ is to be installed on the second disk, the disk must have a `/export/home` directory or partition. This procedure describes how to add a `/data` mount point on a second hard disk. The `/data` mount point uses the slice 2 (`s2`), which spans the entire disk.

Activating a new hard disk requires the following three procedures:

- Installing the new disk.
- Partitioning and labeling the new disk.
- Creating a new file system on the disk.

To Install a New Disk

This procedure describes how to set up the new disk.

1. Exit VnmrJ or VNMR and log out.
2. Log in as root:
   ```
   login: root
   password: root_password
   ```
3. Shut down the UNIX system:
   ```
   # init 0
   ```
   At shutdown, the system forces any information on its way to the hard disk to be written out immediately, cleans up any processes that are running, and executes an orderly shutdown of the system. The process takes about 20 seconds. When the system is safely shut down, the monitor prompt “`>`” or “`ok`” appears.
4. After the monitor prompt “`>`” or “`ok`” appears, turn off the power to the Sun computer.
5. Do the following depending upon the type of disk:
   - “SCSI Disks:,” page 216
   - “IDE Disks,” page 216

SCSI Disks:

1. Set the SCSI target address of the hard disk you are adding. External hard disks usually have a push-button switch on the back of the drive for setting the SCSI target address. Typically, 0 is the SCSI target address for an external hard drive.

   `Connect the SCSI cable and power cable to the hard disk drive.`

2. Power up all the units. Stop the bootup process by pressing Stop-A (L1-A on some keyboards).

3. Wait for the "`>`" or "`ok`" prompt. If the "`>`" prompt is displayed, enter **`n`** to switch to the "`ok`" prompt.

4. Enter the `probe-scsi` command to list all the SCSI devices connected to the SCSI chain. You will see output similar to the following:

5. ok **`probe-scsi`**

```
Target  0
Unit    0    Disk    SEAGATE    ST41600N
SUN1.3G0100541169 ...
Target  3
Unit    0    Disk    SEAGATE     ST1480
SUN0424751600664494 ...
Target  4
Unit    0    Removable Tape  ARCHIVE VIPER 150
21531-004 ...
ok
```

6. Write down thee target address of your new disk. You will need to know it later.

7. Boot the computer using the `boot -r` command. The `-r` option will reconfigure the UNIX kernel and loads the appropriate drivers for the new disk.

   `ok` **`boot -r`**

8. Continue with "To Partition and Label the New Disk," page 217.

IDE Disks

1. Connect the power and data cable to the drive.

2. Set the disk up as a slave (not a boot disk) - refer to the manufactures documentation.

3. Install the disk .

4. Power up all the units. Stop the bootup process by pressing Stop-A (L1-A on some keyboards).

5. Boot the computer using the `boot -r` command. The `-r` option will reconfigure the UNIX kernel and loads the appropriate drivers for the new disk.

   `ok` **`boot -r`**

6. Continue with "To Partition and Label the New Disk," page 217.

To Partition and Label the New Disk

This procedure describes how to use the `format` program to partition and label the new disk drive.

1. After the login prompt appears, log in as root:

```
login: root
password: root_password
```

2. Run the `format` program to partition and label the new disk. The following is example output from the `format` command.

```
# format
Searching for disks ... done

AVAILABLE DISK SELECTIONS:
   0.   c0t0d0   <SUN1.3G cyl 1965 alt 2 hd 17 sec 80>
        /sbus@1,f8000000/exp@,800000/sd@0,0
   1.   c0t3d0   <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
        /sbus@1,f8000000/exp@,800000/sd@3,0
Specify disk (enter its number):0
   selection c0t0d0
   [disk formatted]
```

   Enter the number next to your new disk. In this example, we selected the first disk in the list. This disk happens to be an external 1.3 GB disk at SCSI target address 0.

   The Format Menu appears:

```
FORMAT MENU:
  disk      - select a disk
  type      - select (define) a disk type
  partition - select (define) a partition table
  current   - describe the current disk
  format    - format the disk
  repair    - repair a defective sector
  label     - write label to the disk
  analyze   - surface analysis
  defect    - defect list management
  backup    - search for backup labels
  verify    - read and display labels
  save      - save new disk/partition definitions
  inquiry   - show vendor, product and revision
  volname   - 8-character volume name
  quit
format>
```

3. Enter **p** at the `format>` prompt. The `PARTITION MENU` appears:

```
format> p
PARTITION MENU:
  0       - change ‘0’ partition
  1       - change ‘1’ partition
  2       - change ‘2’ partition
  3       - change ‘3’ partition
  4       - change ‘4’ partition
  5       - change ‘5’ partition
```

```
  6       - change '6' partition
  7       - change '7' partition
  select  - select a predefined table
  modify  - modify a predefined table
  print   - display the current table
  label   - write partition map and label to the disk
  quit
partition>
```

This menu allows you to change the partition values.

4. Change the size of partition 2 to equal the size the disk. At the `partition>` prompt, enter **`2`**, and complete the dialog similar to the following example:

```
partition> 2

Part  Tag         Flag  Cylinders  Size     Blocks
0     backup      wm    0 - 1964   1.27GB   (1965/0/0)
Enter partition id tag [unassigned]: backup
Enter partition permission flags [wm]: wm
Enter new starting cy [0]: 0
Enter partition size [2672400b, 1965c, 0.00mb]: 1304.88mb]:
1304.88mb
partition>
```

Each question provides the current value in brackets. Do not change any value. The size shown in brackets is the size of the entire disk. Use that value.

5. Change the size of the other partitions—0, 1, 3, 4, 5, 6, 7—to 0. Start with partition 0 by entering **`0`** at the `partition>` prompt. An example of the complete dialog appears as follows:

```
partition> 0

Part  Tag          Flag  Cylinders  Size     Blocks
0     unassigned   wm    0          0        (0/0/0)
Enter partition id tag [unassigned]: unassigned
Enter partition permission flags [wm]: wm
Enter new starting cy [0]: 0
Enter partition size [0b, 0c, 0.00mb]: 0.00mb]: 0.00mb
partition>
```

Complete a similar dialog for the other partitions, changing the size of each partition to 0.00mb.

6. After the last partition is finished, enter **`p`** at the `partition>` prompt to list the partition table. Make sure that all partitions except for 2 show 0 under size and that partition 2 shows the size of the entire disk under size.

```
partition> p
current partition table (unnamed):
Total disk cylinders available: 1965 + 2 (reserved cylinders)

Part  Tag          Flag  Cylinders  Size     Blocks
0     unassigned   wm    0          0        90/0/0
1     unassigned   wm    0          0        90/0/0
2     backup       wm    0 - 1964   1.27GB   91965/0/0
3     unassigned   wm    0          0        90/0/0
4     unassigned   wm    0          0        90/0/0
5     unassigned   wm    0          0        90/0/0
6     unassigned   wm    0          0        90/0/0
7     unassigned   wm    0          0        90/0/0
partition>
```

If a partition needs correcting, repeat step 5 above for that partition.

7. Enter **l** (lower case L) at the `partition>` prompt to label the disk:

   ```
   partition> l
   Ready to label disk, continue? y
   ```

8. Enter **q** to quit the partition to leave the `PARTITION MENU` and return to the `FORMAT MENU`.

9. Enter **q** again to leave the format program.

To Create the New File System on the New Disk

This procedure describes how to use the `newfs` command and edit the `vfstab` file to create a new file system on the new disk.

CAUTION: **Running the `newfs` command on a volume will make data unrecoverable on that volume. Be sure the `newfs` command is run on the correct disk.**

1. Run the *v* command on slice 2 of your new disk. The following example shows the command being run on slice 2 (*s2*) of SCSI target 0 (*t0*). Be sure to select the correct target of your new disk.

   ```
   # newfs /dev/rdsk/c0t0d0s2
   newfs: construct a new file system /dev/rdsk/c0t0d0s2:(y/n)? y
   software shows several numbers representing superblock backups
   ```

2. Edit the `/etc/vsftab` to add the new disk:

   ```
   # cd /etc
   # vi vfstab
   ```

   Add the following line:

   `/dev/dsk/c0t0d0s2 /dev/rdsk/c0t0d0s2 /data ufs 5 yes -`

   Where `/data` is the name of the volume. This is a typical name, but you can choose a name that is logical to your system. Whatever name you choose, you will need to know it when you create the mount point later.

   Write and quit from `vi`.

3. Create the mount point to the new disk drive with the following commands:

```
# cd /
# mkdir /data
# mount /data
```

Where `/data` is the name of the vo.lume you used in the previous step.

The new disk is now mounted. You can use the `df -k` command to check it.

18.7 Active SCSI Termination

You may need to use active SCSI terminators if you experience SCSI problems, such as error messages, slow SCSI bus, communication failures, or system crashes. Active terminators can eliminate errors and speed up SCSI bus communications—even on systems with both SCSI-1 and SCSI-2 devices.

Annotated Bibliography

This bibliography is organized into the following sections:

- "Sun and UNIX Newcomers," next
- "UNIX in General," next
- "Specific UNIX Tools," page 222
- "Networking," page 223
- "Network and UNIX Security," page 224
- "Internet and WWW," page 224
- "C Programming," page 225
- "UNIX Programmers," page 225
- "X Window System," page 226

An asterisk (*) indicates a book title that is particularly recommended.

Sun and UNIX Newcomers

The documentation that comes with the Sun workstation gives a brief introduction into the basics of the Solaris user interface. For most sites, the full UNIX documentation (which has to be ordered separately) is not required. Before buying the full documentation, you should first consider loading the UNIX online manuals (when loading UNIX), which will answer most questions.

UNIX in General

*Becker, G., M.S.E. Morris, and K.Slattery, *Solaris Implementation: A Guide for System Administrators*, Prentice Hall, Englewood Cliffs, NJ, 1995. (300 pp.) ISBN 0-13-353350-6

A practical guide to UNIX administrators that migrate their system from SunOS 4.x to Solaris 2.x (up to Solaris 2.4). Covers Solaris installation, boot and shutdown files, system configuration using the AdminTool *and* using command line operations, network setup and maintenance, security setup and maintenance, managing software packages and Solaris patches, disk utilities and archiving procedures, controlling run states with boot files, using unbundled software products such as WABI.

Bourne, S.R., *The UNIX System*, Addison-Wesley Publishing Company, Wokingham, 1983. (351 pp.) ISBN 0-201-13791-7

A practical guide to UNIX that describes basic UNIX operation, the text editors, the Bourne shell, the C programming language, UNIX system programming, `nroff` and `troff`, and the most important data manipulation tools. It does not include some of the 4.2 BSD enhancements, like the C shell.

*Gilly, D., *UNIX in a Nutshell, A Desktop Quick Reference for System V & Solaris 2.0*, O'Reilly & Associates Inc., 1992. (444 pp.) ISBN 1-56592-001-5

A nice, compact quick reference book on System V (SVR4) UNIX and Solaris 2.0. It covers all important user commands in an alphabetically sorted command reference

section (about one-third of the book). Other chapters deal with the different shells, text editing (`emacs`, `vi`, `ex`, `sed`, `awk`), text formatting (`nroff`, `troff`), software development (`sccs`, `rcs`, `make`, `debugging`). System administration is not covered, and `nroff` and `troff` are becoming obsolete, but several subjects (`emacs`, `sccs`, `rcs`, `make`) are included that are barely covered in many larger books.

Libes, D. and S. Ressler, *Life with UNIX*, Prentice-Hall, 1989. (346 pp.) ISBN 0-13-536657-7

Interesting and often amusing book of UNIX trivia, history, etc. Contains lots of information difficult to find elsewhere. Probably for UNIX fans only.

**Nemeth, E., G. Snyder, S. Seebass, and T. Hein, *UNIX System Administration Handbook, 2nd Edition*, Prentice-Hall, 1995. ISBN 0-13-151051-7

Excellent book for everybody involved with UNIX system administration, Contains all the essentials, very easy to read. Includes networking, mail, `uucp`, and many helpful sample scripts for system administration. Almost as good (it's not a life course!) and even broader than administration training at Sun (and much, much cheaper, too!).

*Pew, J.A., *Guide to Solaris*, Ziff-Davis Press, 1993. (625 pp.), ISBN 1-56276-087-4

This is the official, SunSoft-approved learning tool for users of Solaris 2.1, with extensive chapters on the OpenLook, DeskSet, and DeskSet tools (file manager, shell tools, mailtool, calendar manager, etc.), separate chapters on command line UNIX covering the fundamentals, `vi` editor, C and Korn shells, networking (limited), and an outline of UNIX system administration (based on DeskSet system administration tools). Very readable, but for a better coverage of system administration see J. Winsor's *Solaris System Administrator's Guide* below.

*Winsor, J. *Solaris System Administrator's Guide, Second Edition*, Sun Microsystems Press, 1997. (335 pp.) ISBN 1-57870-040-X

Very useful book for system administrators that use Solaris 2.x and especially those who plan switching from Solaris 1.x (SunOS 4.1.x) to Solaris 2.x. Covers all important aspects of system administration that have undergone changes with Solaris 2.x, such as file system administration, networking, printing, troubleshooting, and new utilities (the Korn shell, system administrator tools, etc.). The second edition includes coverage of Solaris 2.6.

*Winsor, J. *Solaris Advanced System Administrator's Guide*, Ziff-Davis Press, 1994. (477 pp.) ISBN 1-56276-131-5

Complements the *Solaris System Administrator's Guide* by the same author. Covers advanced system administration topics such as installing mail services, adding NIS+ clients, the automounter, the service access facility for setting up printers, modems and terminals, and an introduction to Bourne shell programming, including shell scripts for automating routing administration tasks.

Specific UNIX Tools

Aho, A., B. Kernighan, and P. Weinberger, *The awk Programming Language*, Addison-Wesley, 1988. (210 pp.) ISBN 0-201-07981-X

The definitive book on *awk*, by its designers.

Anderson, G., and P. Anderson, *The UNIX C Shell Field Guide*, Prentice-Hall, Englewood Cliffs (New Jersey), 1986. (374 pp.) ISBN 0-13-937468-X

Very good, in-depth textbook on the C shell, one of the main features of 4.2 BSD UNIX not described in the book by S.R. Bourne. For people who want to go into extensive C shell programming.

Hansen, A., *vi—The UNIX Screen Editor, A User's Guide*, Prentice-Hall, New York, 1986. (230 pp.) ISBN 0-89303-928-4

Probably the most complete documentation on *vi*, only for people that intend to use *vi* almost full time.

Harrison, M., and M. McLennan., *Effective Tcl/Tk Programming, Writing Better Programs with Tcl and Tk*, Addison-Wesley, Reading (Mass.), 1998. (405 pp.) ISBN 0-201-63474-0

Nice introduction into Tcl/Tk programming; easy to read, well structured. Covers also newer features, such as grid, but not as detailed and exhaustive in general as the other two books mentioned below.

Ousterhout, J.K., *Tcl and the Tk Toolkit*, Addison-Wesley, Reading (Mass.), 1986. (458 pp.) ISBN 0-201-63337-X

A must for users who want to know more about Tcl/TK programming and who want to start writing their own visual tools. This is *the* reference book on the Tcl/Tk scripting language, written by the creator of the language. Complete, detailed, and exhaustive, although no longer quite up-to-date with the newest standards of the language.

Welch, B.B., *Practical Programming in Tcl and Tk, Second Edition*, Prentice-Hall PTR, Upper Saddle River (NJ.), 1997. (613 pp.) ISBN 0-13-616830-2

Very detailed and complete book, covers the newest level (8.0) of the Tcl/Tk language; definitely more up-to-date than the book by Ousterhout (see above), but not quite as well written. A complete language reference.

Networking

Costales, B., E. Allman, and N. Rickert, *sendmail*, O'Reilly & Associates Inc., 1993. (792 pp.) ISBN 1-56592-056-2

The `sendmail` "bible." One of the authors is the creator of `sendmail`. Contains all the information you need to fully understand and configure `sendmail` on your systems. Very detailed and complete. For network administrators only.

Hunt. C., *TCP/IP Network Administration*, O'Reilly & Associates Inc., 1994. (472 pp.) ISBN 0-937175-82-X

A detailed handbook about the internals of TCP/IP networking, covering the basics, configuration/setup, routing, DNS setup, sendmail, troubleshooting, security. For network administrators only.

Ravin, E., T. O'Reilly, D. Dougherty, and G. Todino, *Using & Managing UUCP*, O'Reilly & Associates Inc., 1996. (416 pp.) ISBN 0-56592-153-4

Excellent handbook on networking via dial-up connections. Covers most current versions of UUCP, and all the commands involved in communicating via UUCP.

Ramsey. R., *All About Administering NIS+*, Prentice Hall, 1994. (451 pp.) ISBN 0-13-309576-2

A detailed book that gives information on how NIS+ works, how to plan and set up the NIS+ service, transitioning from NIS to NIS+. For administrators of NIS+-based networks only.

Tanenbaum, A.S., *Computer Networks*, Prentice Hall, 1989. (658 pp.) ISBN 0-13-162959-X

The bible on the OSI networking model. Covers all aspects (including programming) of the OSI 7-layer networking model (one detailed chapter per layer), network topologies, and a complete annotated bibliography. For network administrators and designers and programmers of networking software.

Network and UNIX Security

Cheswick, W.R., and S.M. Bellovin, *Firewalls and Internet Security*, Addison-Wesley., 1994. (306 pp.) ISBN 0-201-63357-4

Well-written book. Gives an overview of common security holes with the various network protocols (including newer ones, like WWW) and describes the firewall functionality in detail. This book will enable gateway administrators to properly configure a firewall gateway, using a commercial firewall package or even by building a firewall. For specialists and gateway administrators only.

Garfinkel, S., and G. Spafford, *Practical UNIX & Internet Security*, O'Reilly & Associates Inc., 1996. (1004 pp.) ISBN 0-56592-148-8

A well-written, complete book on all aspects of UNIX security. Covers basic security features, communications (`uucp`, NFS, Kerberos and secure RPC), firewall machines, detecting and handling security incidents, data encryption, etc.

Internet and WWW

*Estrada, S., *Connecting to the Internet*, O'Reilly & Associates Inc., 1993. (170 pp.) ISBN 1-56592-061-9

A nice little guide into the basics of how to connect to the Internet. Discusses Network performance, Internet options, choosing a service provider (ISP), dialup and leased line connections. The book also contains a list of ISPs for most parts of the world. Quite useful if you want to set up a new Internet connection, at home or at work.

*Fox, D., and T. Downing, *HTML Web Publisher's Construction Kit*, Waite Group Press, 1995. (673 pp.) ISBN 1-57169-018-2

Nicely written and readable book. The first part discusses the most important Web browsers (Lynx, NCSA Mosaic, Netscape Navigator and others). The second part is a good guide about constructing a Web page, including an introduction to HTML (up to HTML 3), text formatting, graphics, forms, a brief introduction into CGI scripts, conversion of data into HTML format, and HTML editors. A third part deals with setting up a Web server. You obviously only need this book if you are into creating your own Web page.

*Krol, E., *The Whole Internet, User's Guide and Catalog*, O'Reilly & Associates Inc., 1994. (574 pp.) ISBN 1-56592-063-5

Instructive and complete book that explains all the services available on the Internet, telnet, FTP, News, Archie, Usenet, Gopher, WAIS, WWW, X software, etc. Includes a catalog of Internet resources and entry points for a wide range of topics.

C Programming

Brown, D.L., *From Pascal to C, An Introduction to the C Programming Language*, Wadsworth Publishing Company, Belmont (California), 1985. (151 pp.) ISBN 0-534-04602-9

More introductory than an in-depth textbook and not very complete, but still good for people who started programming in Pascal and now want to switch to C.

Harbison, S., and G. Steele, *C: A Reference Manual*, Prentice-Hall, 1987. (404 pp.) ISBN 0-13-109802-0

Excellent reference book for the C programming language, including the draft ANSI C. Much better than Kernighan and Ritchie as reference text.

Kernighan, B.W., and D.M. Ritchie, *The C Programming Language, Second Edition*, Prentice-Hall, Englewood Cliffs (New Jersey), 1988. (272 pp.) ISBN 0-13-110362-8

The "C bible"—the original textbook on C, written by the inventors of the language. Mostly still up to date and a must for most C programmers, although not particularly good for learning C. The second edition has all the code written in ANSI C. As a reference manual, the book by Harbison & Steele is much better.

Koenig, A., *C Traps and Pitfall*s, Addison-Wesley, 1989. (147 pp.) ISBN 0-201-17928-8

Very useful book. It discusses many of the subtle and not so subtle errors that C programmers make at some time.

van Wyk, Ch, *Data Structures and C Programs*, Addison-Wesley, 1988. (387 pp.) ISBN 0-201-16116-8

Typical computer science book of algorithms and data structures. The code examples are very clear and readable, generally rare for books on data structures written in C (most books on data structures are coded in Pascal, which is generally more readable, even for C programmers).

Wortman, L., and T. Sidebottom, *The C Programming Tutor*, Prentice-Hall, 1984. (274 pp.) ISBN 0-13-110024-6

Intensive introduction to C for competent programmers. Good introduction on pointers and some reasonably long (500 to 1000 lines) and useful sample programs.

UNIX Programmers

Bach, M., *The Design of the UNIX Operating System*, Prentice-Hall, 1986. (471 pp.) ISBN 0-13-201757-1

Excellent book about the internals of UNIX, mostly AT&T System V. Lots of pseudocode and C programs describing the internal data structures and algorithms. For UNIX maniacs only.

Kernighan, B., and R. Pike, *The UNIX Programming Environment*, Prentice-Hall, 1984. (357 pp.) ISBN 0-13-937681-X

A wide-ranging introduction to UNIX, intended more for programmers.

Rochkind, M., *Advanced UNIX Programming*, Prentice-Hall, 1985. (265 pp.) ISBN 0-13-011800-1

One of the better books available on UNIX system programming. Extensive example code for all the system calls available under AT&T System V UNIX. Some coverage of system calls under BSD UNIX and Xenix. Very readable.

*Stevens, W.R., *Advanced Programming in the UNIX Environment*, Addison-Wesley, 1992. (744 pp.) ISBN 0-201-56317-7

Excellent book on programming under UNIX in general. Covers both BSD (4.3) and System V UNIX. Lots of sample source code (the concept is learning by reading code), Covers all aspects of programming under UNIX, like I/O, file and directory handling, process control, signals, inter-process communication, daemons, database programming, printer, modem and pseudo-terminal interfaces. Lots of exercises. The sample code in the text is available via anonymous FTP through UUNET.

X Window System

The books listed below (except for Mansfield) are only recommended for X Window system programmers. Most users of the X (be it OpenLook or Motif) do not need to program the system and will find the user-oriented book by Mansfield to be most useful.

Cutler, E., D. Gilly, and T. O'Reilly, *The X Window System in a Nutshel*l, O'Reilly & Associates Inc., 1992. (424 pp.), ISBN 1-56592-017-1.

This manual forms a quick reference to the essentials of volumes 1, 2, 3, and 4 of *The Definitive Guides to the X Window System* (see below), providing faster access to the important information (Xlib and X toolkit functions, structures, data types, font conventions etc.) than the full programming and reference manuals (which may still be needed for detailed references).

Ferguson, P., and D. Brennan, *Motif Reference Manual* (volume 6B of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1993. (920 pp.), ISBN 0-56592-038-4.

A good, complete reference manual for the Motif toolkit. Complements the *Motif Programming Manual* by D. Heller, P. Ferguson, and D. Brennan.

Flanagan, D., *Motif Tools, Streamlined GUI Design and Programming with the Xmt Library* (volume 6C of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1994. (1024 pp.), ISBN 0-56592-044-9.

A book that is supposed to speed up Motif programming. It is a practical guide to Motif programming, with lots of tips and tidbits from other Motif programmers. See also the *Motif Programming Manual* by D. Heller, P. Ferguson, and D. Brennan.

Flanagan, D., *X Toolkit Intrinsics Reference Manual*, (volume 5 of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1992. (916 pp.), ISBN 1-56592-007-4

The official reference book on X Toolkit intrinsics. Probably a must for people that want to do extensive X programming. This is merely a reference manual, not a textbook, and does not provide extensive examples (it goes together with the *X Toolkit Intrinsics Programming Manual*).

Heller, D., P. Ferguson, and D. Brennan, *Motif Programming Manual* (volume 6A of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1994. (1016 pp.), ISBN 0-56592-016-3.

A good, complete reference book on programming for the Motif GUI. Some workstations (like SGI) may come with Motif manuals of their own, in which case this book is not required, and it then depends on the contents of these manuals, whether any of the other X programming or reference manuals are still required. With this book certainly a programmer would still need the manual on the X toolkit intrinsics (volumes 4 and 5 of that series) for basic window programming, as well as the Xlib manuals (volumes 1 and 2) for graphics and low-level programming.

Mansfield, N., *The X Window System: A User's Guide*, Addison-Wesley 1991. (344 pp.) ISBN 0-201-56344-4

A tutorial introduction to X that assumes no knowledge of X or any other windowing system. Describes the background of the X Window system, explains how to use the system, and provides information on ways to customize it.

Nye, A., *Xlib Programming Manual* (volume 1 of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1992. (824 pp.), ISBN 1-56592-002-3

The official programming manual on lower-level X programming (Xlib); required for people who want to program graphics under the X Window system.

Nye, A., *Xlib Reference Manual* (volume 2 of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1992. (1138 pp.), ISBN 1-56592-006-6

The official reference manual on lower-level X programming (Xlib). Required for people that want to program graphics under the X Window system. It goes together with the *Xlib Programming Manual*.

Nye, A., and Tim O'Reilly, *X Toolkit Intrinsics Programming Manual (Motif Edition)*, (volume 4 of *The Definitive Guides to the X Window System*), O'Reilly & Associates Inc., 1992. (674 pp.), ISBN 1-56592-013-9

The official programming textbook for programmers using the X Toolkit Intrinsics (anybody programming new X programs). It addresses both OpenLook and Motif, and the *X Toolkit Intrinsics Reference Manual* is more or less a mandatory addition and complement to that book. For lower-level graphics programming, volumes 1 and 2 most likely are also required, and higher-level programming for OSF/Motif is addressed in volume 6 of that series. (Volume 7 covers XView programming on Sun workstations, but because all manufacturers seem to be converging towards the Motif GUI, this can no longer be recommended.)

Smith, J. D., *X, A Guide for Users*, Prentice Hall, 1994 (350 pp.), ISBN 0-13-123795-0

A tutorial approach to the window manager as well as more advanced features of X. Provides extensive coverage of the X customization process.

UNIX System Laboratories, *OPEN LOOK Graphical User Interface—Programmer's Guide*, Prentice Hall, 1992. (ca. 700 pp.), ISBN 0-13-726605-7

A relatively readable book on programming for the OpenLook GUI environment (about as readable as a single book on that subject can be). Written for advanced C programmers. Covers OLIT (the OpenLook Intrinsics Toolkit). Written for generic System V UNIX systems, some of the procedures (compilation) described in this book will require adaptation for an implementation on Sun workstations. The examples are not bug-free; you will learn much by debugging as well.

Glossary of UNIX, Sun, and VnmrJ Terms

This glossary was written specifically for users of Varian Inc. spectrometers using Sun workstations and VnmrJ software. It is not meant to be valid for all computer users. Therefore, you will find different definitions for the same keywords in other computing environments. This material is also not meant to be read through or even to be learned from start to end, but rather to be a reference for help in reading Varian Inc. and Sun publications, and to be a means to facilitate communication between different VnmrJ users by creating a common base terminology. The glossary does not cover multiprocessor workstations and terms relating to server technology.

100baseT: Mechanical and electrical standard for 100-Mbps twisted pair fast Ethernet; *see also* Ethernet, twisted pair, RJ45.

10/100baseT: Newer Ethernet interfaces are handling both 10baseT (10 Mbps) and 100baseT (100 Mbps), with automatic switching between the two speeds. *See* 100baseT, 10baseT, RJ45.

10baseT: Mechanical and electrical standard for 10 Mbps twisted pair Ethernet. *See also* Ethernet, twisted pair, RJ45.

16-bit computer: Not well defined; for example, a CPU chip such as the MC 68000 from Motorola has a 24-bit address path (16-MB address space), a 16-bit data and instruction I/O path, and can perform 16- and 32-bit calculations. However, because the I/O bandwidth is usually the main speed bottleneck, such a CPU is often called a “16-bit CPU” (but often also 16/32- or 32-bit). Examples of other 16-bit CPU chips are the Intel 8086 and 80286 microprocessors, and the 6800 CPU from Motorola. *See also* 32-bit computer, 64-bit computer.

32-bit computer: Defines a CPU chip with 32-bit wide data and instruction I/O path. 32-bit CPUs are also laid out mainly for 32-bit calculations, and they typically have a 32-bit address path (4 GB address space). Examples for 32-bit CPU chips are the SPARC, MicroSPARC, HyperSPARC, and SuperSPARC CPU chips from Sun; the MIPS R3000; the PowerPC 601, 603 and 604 microprocessors from Motorola—all in the area of RISC computing. Examples of 32-bit CISC processors are the 68020, 68030, and 68040 CPUs from Motorola, and the Intel 386, 486, and Pentium microprocessors. Most current graphics workstations use 32-bit microprocessors in their CPU. *See also* 16-bit computer, 64-bit computer, RISC, CISC, CPU.

3D-RAM: New RAM technology jointly developed by Sun and Mitsubishi for the Sun Creator and Creator3D graphics controllers. Combines DRAM, pixel ALU (arithmetic logical unit), SRAM, and a VRAM video buffer on a single chip. Offers dramatically improved graphics performance over conventional VRAM. *See also* UltraSPARC, Creator, Creator3D, ALU, DRAM, SRAM, VRAM.

4.2BSD (4.2 version, Berkeley software distribution): The basis for the Sun operating system until SunOS 4.1.4 (Solaris 1.1). 4.2BSD was developed at UC Berkeley and derived indirectly from UNIX Version 7 (written at Bell Labs for the PDP-11 microcomputer).

64-bit computer: Defines a CPU chip with 64-bit data and instruction I/O path. 64-bit CPUs are laid out for 32-bit and 64-bit calculations, and they typically have a address path larger than 32 bits (over 4 GB address space). Typical examples for 64-bit CPU chips are the UltraSPARC CPU from Sun, the MIPS R4000, R4400, R4600, R5000, R8000, R10000,

and the Motorola PowerPC 620 and 750 (also called G3). Graphics workstations with 64-bit CPUs are now becoming available. *See also* 16-bit computer, 32-bit computer, UltraSPARC, Ultra, CPU.

68040: *See* MC68040.

abort: Irreversible interruption of a program by some external signal. Under UNIX, programs may be aborted by pressing keys such as Ctrl–c and Stop–a, by special commands like `kill`, or by aborting the parent process, such as exiting Suntools with active windows.

access: *See* permissions.

access time: Sometimes used for the average seek time to access tracks on disk drives (usually 8 to 15 ms), but may also characterize the speed at which RAM or ROM memory can be accessed (currently around 60 nsec on Sun computers). For disk drives, the access to individual sectors is longer than specified by the average seek time, for example, due to rotational delays. *See* latency.

acoustic coupler: Simple, low-priced modem that operates up to 1200 baud and does not require special authorization or installations on the phone system.

Acrobat Reader: Adobe software for viewing and searching documents in PDF (Portable Document Format) files. For the VNMR online manuals, Varian has switched from FrameViewer (and manuals in native FrameMaker format) to Adobe Acrobat Reader and PDF files. *See* PDF, Adobe, FrameViewer, online manual.

active window: Unlike standard Macintosh and similar PC-based software, all windows on a Sun are active processes, even icons. But only one window, called the active window, can have the focus for keyboard entries (although obviously the terminology is wrong). The window focus is indicated by a solid frame. The Sun windowing software (SunView) allows for two options: activate windows by clicking on them with the mouse or move the mouse pointer onto the window (difficult when using small windows). The focus can also be on iconized windows, which are then unfortunately not visualized, and you can (inadvertently) type text into an icon.

address: A number or a sequence of characters enabling computers to find and identify some hardware or a piece of software.

alias: Alternate name for a command.

ALU (Arithmetic Logical Unit): Execution unit inside a CPU chip that handles integer calculations and logical decisions. Modern CPU chips contain multiple ALUs that can execute integer operations in parallel.

anonymous FTP: Special FTP account on public FTP servers that requires entering the user name “anonymous” at login. The account has no real password definition, but for logging purposes (FTP logins are logged into a file) the convention is to enter the user’s e-mail address as a password. Most Web browsers also have a convenient graphical interface for anonymous FTP sites. Anonymous FTP is usually used to download public domain software. *See also* FTP, Web browser.

ANSI C: Newer standard that was proposed for the C programming language. ANSI C differs in many ways from the “traditional”, “Kernighan & Ritchie” C (i.e., the one originally proposed by the authors of the language); the most prominent change is in the declaration of function arguments, which are defined within the parentheses, rather than after, as in K&R C. This provides for better argument type checking. ANSI C (but not K&R C) is a subset of C++. *See also* C, C++.

API (Application Programming Interface): A set of high-level library functions used in programming in or for certain software environments. Examples are the X Toolkit (used

in X Window System programming), OLIT (OpenLook Intrinsics Toolkit), OpenGL, *See* OpenGL, OLIT, X Toolkit.

applet: Small Java program, compiled to byte codes, downloaded as part of a Web page, and executed on the Web client. Applets are used, for example, to produce an animated Web page or to allow for user interaction with the Web page without involving further communication with the Web server. *See* Java, HTML, WWW.

argument: Information passed to a command or subroutine (as part of the actual call) to further direct its operation.

ARPANET: The prototype, the predecessor, and later an early integral part of the Internet. The TCP/IP protocols were developed for ARPANET. *See* Internet, TCP/IP.

array processor: Specialized boards used on some systems (such as some Sun-3 systems) to speed up floating point operations on data arrays (vectors). Due to their architecture (using pipelines and simultaneous processing), array processors are very efficient on vectorized data, even using floating point operations, but they can not be used for scalar operations. The standard floating point hardware used on Sun workstations is equally well suited for vectorized and scalar operations, such that array processors are no longer needed. Array processors require modifications in the software, involve considerable processing overhead (like transferring the data into dedicated memory), and are available to only one process at a time.)

ASCII (American National Standard Code for Information Interchange): Represents characters with 7- or 8-bit numbers. ASCII is the most important standard for the transmission of text files.

ASIC (application-specific integrated circuit): Custom devices that help to simplify printed circuit boards by using fewer components. An example is the video chip used on newer Sun workstations.

assembly language: A low-level programming language that is very close to entering machine language directly. Allows writing very fast programs (normally faster than any compiler could do), but the complexity of these programs is limited, especially in large programs, which are hard to debug and modify because it is difficult to read and fully understand them. Assembly language is used mostly to speed up uniform, repetitive tasks on large data tables (such as Fourier transformations and pixel operations) or with non-standard data formats. The main disadvantage of assembly language programming is that it is intended for certain computer hardware; therefore, programs are usually not portable between different computers. Assembly language programming is usually limited to small modules called from within a high-level program.

asynchronous: Transmission in which the speed is determined by conventions such as the baud rate (in communication over a RS-232 cable) and not through synchronization signals (e.g., lines 15, 17, and 24 in a RS-232 cable). Asynchronous transmission is less demanding on the hardware side. Asynchronous transmission must always account for possible deviations from the convention (with RS-232, up to 38400 baud can be obtained in asynchronous transmission). Note that even with asynchronous transmission, the devices that communicate have some control over each other through handshaking signals that, for instance, enable the receiving device to stop the signal flow when its buffer is full. *See* handshaking.

AT&T Bell Labs: *See* System V.

ATM (Asynchronous Transfer Mode): New, emerging standard for networking, using either unshielded twisted pair (UTP) or fiber-optical transmission media (depending on the speed). ATM transmits all data in the form of 53-byte packets, which simplifies the ATM adaptor hardware and allows for potentially higher throughput than Ethernet. But unlike

Ethernet, ATM transmission rates are isochronous (the transmission speed for a given link is kept constant for the duration of the connection) and not dependent on the network load (on Ethernet, increased collision rates at high loads cause a slowdown in the network efficiency). This makes ATM ideal for the transmission of voice and video data, such as in multimedia applications and video conferencing. Standard ATM speeds are 25, 155, 622, and 2488 Mbps. Sun currently offers ATM adaptors for 155 Mbps (over unshielded twisted pair) and 622 Mbps (over fiber-optic cable). Ethernet operates at 10 Mbps (standard Ethernet) or 100 Mbps (fast Ethernet, available on Sun as a SBus card or in Ultra workstations with Creator/Creator3D graphics). *See also* Ethernet, Ultra, UTP.

AUI (attachment universal interface) connector: Older Ethernet connector and cabling, with shielded 15-pin cabling and slide-lock connectors. AUI is standard for connecting transceivers for coaxial Ethernet (N-Type/Inline, Vampire/Thick Ethernet, and BNC/Thinnet) networks. Older workstations had an AUI connector that could be used to attach a transceiver directly or via AUI transceiver cable. Newer systems offer an MII connector that also permits connecting to UTB and fiber-optic networks (plus an RJ45 connector for 10/100BaseT), and for AUI transceivers an MII-to-AUI cable must be used. *See* MII, Ethernet.

authoring tool: *See* Web authoring tool.

background process: A process detached from the calling environment. The calling window remains accessible during the execution of the background process. Processes that produce lots of output are not suited for background operation, because the output may interfere with other programs running in the calling shell.

backplane: Hardware in the back of a card cage that links all boards through single- or multiple-bus structures. Every board fits into connectors in the backplane.

backup: To copy files to secondary storage (usually a tape or a optical disk on newer systems) for safety in case the originals are lost or unusable.

backup tape: Magnetic tape that contains a copy (duplicate) of data on a hard disk. Used for safety reasons so that data can be recovered if data becomes corrupt or if the disk dies due to a head crash.

bad track: Track on a disk where data cannot be written or read reliably. Most disks have bad tracks. During the surface analysis, which is done after formatting the disk, such tracks are labeled and subsequently replaced by other tracks. Bad tracks can eventually show up due to aging. In such a case, the disk has to be reformatted with surface analysis. Some disk drives automatically recognize bad blocks at runtime and assign spare blocks as replacement.

bandwidth: Characterizes the speed of a display (MHz) or a bus (MB per second). For screens, the bandwidth does *not* describe the number of pixels drawn per second (lines × columns × refresh rate ÷ interlacing, e.g., 1024 × 1280 × 66/1 = 86,507,520), but rather the speed at which the electron beam can switch between bright and dark (in a good monitor, bandwidth must be more than the number of pixels drawn per second).

batch: Confined (well-defined) amount of data.

batch mode: For mainframe computing, the execution of programs in packets (batches), where a program can be executed in a single batch or in several batches, not necessarily one after each other. Because the Sun is a multiuser system, many users could basically use the same plotter; therefore, the plots have to be plotted in batches (usually a full page) such that plot requests from different users are not mixed up on the same page.

battery: An in-chip battery keeps the real-time clock on Sun CPU boards running often for over 10 years.

baud rate: Speed of transmission in serial interfaces (modulations per second, corresponding roughly to bps, or bits per second). Some standard speeds for the RS–232 are 300, 600, 1200, 2400, 4800, 9600, 19200, and 38400 baud.

Berkeley UNIX: *See* 4.2BSD.

beta test: After extensive testing in the factory (software development group, applications laboratory, etc.), selected users test the beta version of software before it is released to customers.

bin: Name often used for a directory that contains executable software.

binary file: File containing information that can directly be interpreted by the CPU. Binary files include 16- or 32-bit integers, 32-bit floating point numbers or compiled programs (object code). Other file types are directories and text files.

bit: Single yes or no (1 or 0) information that makes up the smallest binary data unit.

block: A group of data on a disk or a tape that is transferred as a unit. With 4.2BSD UNIX, data are normally stored in 8-KB blocks. *See also* I–node.

BNC cable: A type of coaxial cable with 50-ohm impedance, used for screens, Ethernet, and other rf connections.

board: Hardware that can contain a complete computer or parts of it.

bootup: The process of starting an operating system. On UNIX systems, this involves a series of steps: loading the primary and secondary bootstrap, loading the kernel, finding all the available hardware, mounting file systems, checking file systems, starting up all the necessary daemons, and initiating the login shell in multiuser mode.

bootup messages: Messages automatically displayed after or during the bootup.

Bourne shell: Widely used UNIX command interpreter, common to all UNIX systems. On UNIX systems, Bourne shells are seldom used for interactive work but rather for shell scripts, where they are most efficient. *See* shell, C shell.

browser: *See* Web browser.

BSD (Berkeley Software Distribution): S*ee* 4.2BSD.

buffer: Memory for temporary data storage. A buffer is often used to link two devices with different inherent speeds, such as a computer and a printer.

bug: Error in software. Totally bug-free software for a large program is considered impossible. Software programmers can only try to reduce the number of bugs, a time-consuming process. *See* debugging.

bus: Hardware for transporting data between different devices. The receiving device can be addressed by the device that wants to send data, so that the data traffic occurs only between two specific units.

bus arbiter: Handles requests for data transfers and decides which device is allowed to become the bus master. *Compare with* bus master.

bus master: Only one device, the bus master, on a bus is allowed to control data transmission at a time.

bus terminator: Most buses require terminators for proper operation. For example, early SCSI bus devices had to be terminated by a termination plug or by a small chip (termination resistors) on the last device. Therefore, when changing the sequence of devices on a SCSI cable, users needed to check whether the bus was still properly terminated or else the bus

and the entire system could be nonfunctional. Many modern SCSI devices are self-terminating: termination plugs are no longer required.

button: Switch that is activated by pressing, for example, the Sun mouse has three buttons. In a graphics user interface (GUI), menus can contain items called buttons that can be selected with a mouse click or by pressing a function key on a keyboard.

byte: A unit of 8 bits, commonly used to characterize the size of files, memory, or other storage devices, although the data might be organized in 16- or 32-bit words, or even 64-bit units or larger for floating point numbers.

C: A powerful, flexible programming language developed for UNIX by Dennis Ritchie based on Ken Thompson's language B in the in the early 1970s. Although Pascal is usually considered better for learning and teaching, C produces very efficient (small and fast) object code. C comes in two "flavors": the original, "Kernighan & Ritchie" (K&R) C, and ANSI C. K&R C was used to write most of the VNMR software. *See also* ANSI C.

C++: Superset of the C (actually: ANSI C) programming language, with tools and utilities for object-oriented programming (OOPS). Most implementations of C++ consist of a preprocessor that first translates C++ into C, followed by normal C compilation. SunSoft's C++ uses a direct compilation technique for faster compiling. Different from C, C++ uses extensive internal argument and type checking, which makes it considerably safer than C. C++ comes with several powerful programming options, such as operator and function overloading, allowing the programmer to redefine the meaning of standard math operators, such that they work with complex objects. This feature makes it easy to define functions with variable number and type of arguments. Compared to C, C++ allows for new levels of simplification and abstraction in programming. Some VNMR utilities, such as the ImageBrowser, are written in C++. *See also* ANSI C, C, OOPS.

C shell: UNIX command interpreter developed and optimized for interactive work. Originally, it was specific to 4.2BSD UNIX systems, but now is mostly used for interactive work and writing shell scripts.

cache memory: Fast memory buffer that communicates directly with the CPU, thus avoiding unnecessary wait-states due to slower transfers to disks or ordinary memory. The most recent generation of workstations uses two levels of cache memory: 6 to 36 KB of primary cache on the CPU chip, and 0 to 2 MB external secondary cache on a separate chip.

capacity: Size of disks or memory, usually in kilobytes (KB), megabytes (MB), or gigabytes (GB).

caddy: Cartridge that holds and protects the CD-ROM in certain types of CD-ROM drives. Without caddy, a CD-ROM could not be inserted in these drives.

card cage: Structure that contains the backplane and the support for printed circuit boards.

case-sensitive: Pertains to systems or programs when the case of letters, either upper or lower, makes a difference. UNIX is case-sensitive in most matters, and the C programming language is always case-sensitive.

CDE (Common Desktop Environment): In response to Microsoft Windows 95, the major UNIX manufacturers (including Sun, IBM, HP, and Novell) decided to define a global standard for the X Window System and UNIX GUI. Sun's CDE is one of the first GUIs that comply with this new standard. CDE is mainly based on Motif standards. Sun brought in some of its desktop and development tools and some internal protocol definitions from OpenLook, while it had to give up many basic OpenLook features such as the look-and-feel. CDE has become the new standard Window interface with Sun and

Solaris; it is included with Solaris 2.5. As of release 5.2F, VNMR is compatible with CDE (while maintaining compatibility with OpenLook). *See also* GUI, OpenLook.

CD-R (recordable compact disk): Optical disk (same form factor as used for audio disks) holding up to 644 MB of software or data. CD-R disks can be written only once and can serve as a permanent data storage for archiving data. Unfortunately, only a few manufacturers are building CD-R drives for UNIX workstations. Writing CD-R disks might be tricky because they typically need to be written once and in one continuous step; an interruption in the data flow might make the disk unusable. In the future, the DVD-R might replace CD-R drives. *See* CD-ROM, CD-RW, DVD-R.

CD-ROM (compact disk-ROM): Read-only optical disk (same as used for audio disks) containing up to 644 MB of software or data. Different from audio records, the information on CD-ROMs is organized in concentric, circular tracks (not in a single spiral track), and a CD-ROM is written starting with the innermost track. The Solaris 2.X operating systems and many software packages, such as VNMR, are exclusively available on CD-ROM. Sun UNIX documentation is also available on CD-ROM. The Sun CD-ROM drive can also be used to play audio CDs.

CD-RW (rewritable compact disk): Removable optical disk that can be rewritten just like a hard disk (same form factor as used for audio disks) holding up to 644 MB of software or data. Only a few manufacturers are building CD-RW drives for UNIX workstations. With the increased storage requirements, the DVD-RAM might replace the CD-RW drives. *See* CD-ROM, CD-R, DVD-RAM.

Centronics port: 8-bit parallel port for printing. A Centronics port is available on many printers, such as the LaserJet series of printers, and is standard on the SPARCstation and Ultra workstations. SBus cards with a parallel port are available for systems that don't have a built-in parallel port. Printing over a parallel port can be more than five times faster than printing through a serial port using RS-232.

CGI (common gateway interface): Defines how a Web client displaying a Web form (an interactive Web page) and a Web server communicate and interact. A user who submits a Web form causes a special character string (as defined in the form) to be sent to the Web server, to a specified processing tool, which is often a script written in Perl but can be a simple shell script or a program written in C or any other language). This software then causes further actions to be performed, including: HTML code is sent back to the Web client, mail is sent to the specified address, a database is searched, and the search results are sent back to the client. CGI is the definition of the interface between the form (the Web client) and the Web server. See WWW, HTML, Perl.

child: A process that is started from another process is called a child of that calling process, as long as it runs in foreground. When the parent of the child process dies or is killed, the child also dies automatically. Error output is normally sent to the parent process. The only way of detaching a child from its parent process is to run the program in background. Output is then still sent to the calling shell as long as that process is alive. Afterwards, the output is lost. *See* process.

CISC (complex instruction set computer): A type of CPU chip. Typical CISC chips are the Motorola 68000 series and most other 16- and 32 bit CPU chips used in personal computers and in older workstations. *Compare with* RISC.

class: An "object type definition" in object-oriented programming style (OOPS). A class defines the properties (private and public elements, constructor, destructor, etc.) of an object. An object is an instantiation of a class. *See* OOPS, class.

clicking: Pressing a mouse button. Most often, this involves moving the mouse pointer onto a particular object shown on the screen and then pressing a button (and in most cases releasing it again).

client: Computer system that uses facilities (usually disk space) on another computer (server). For the X Window System (X.11), the terms client and server have a different meaning. *See* server, X Window System, X client, X server.

clock: Most computers have a built-in real-time clock powered by a battery that typically lasts for 10 years or more.

clock frequency: Basic frequency at which a processor operates. This is *not* the rate at which CPU commands are executed, because many commands require several clock cycles, especially on CISC computers. **coaxial cable:** A high-frequency rf transmission cable that is intended to avoid interference with other rf devices.

collision: Happens when two devices want to transmit data simultaneously over Ethernet. Both devices try again after random delays. The number of collisions increases with the workload on the Ethernet.

command: Stand-alone program (or a shell script, macro, or menu button) that is started by entering its name or clicking on its menu button See *argument*.

command interpreter: Program that reads commands (such as operator input) and calls the necessary system functions to execute the command. VnmrJ and VNMR have a built-in command interpreter, Sun UNIX comes with command interpreters called shells, and many interactive programs have a command interpreter built in. *See* shell.

command line: A line of characters containing one or more commands. Can also refer to a particular position on the screen display, where a command line can appear. The C shell on 4.2BSD UNIX systems includes an online command line editor that allows modification and execution of the current or previous command lines.

command mode: Special mode of operation with some editors (such as `vi`) where keystrokes do not enter characters into the text, but are interpreted as editor commands.

compiler: Program that converts source code (programming instructions that the programmer can easily read, modify and understand) into machine language for the CPU (instructions that are almost impossible to decipher).

computer: General term for an electronic machine that can perform various arithmetic, logical, and flow-control instructions at a high rate of speed.

configuration: A particular assemblage of hardware or software components.

connector: A piece of hardware that links a board to the backplane or a cable.

console: The device on which the operator works, including a keyboard and text output devices such as a screen or a printer. Under UNIX, the console is also the shell (or the window) on which system messages are presented. Output and error messages from programs running in background are often redirected into the console window.

constructor: Definition of actions, such as variable initialization, that occur automatically upon creation, or instantiation, of an object in object-oriented programming. *See* OOPS, object.

converged UNIX: UNIX version that combines parts of the major branches of the UNIX family—4.2BSD from UC Berkeley and System V from AT&T Bell Laboratories. Solaris is a converged UNIX, based on System V.4, containing most features from 4.2BSD.

coprocessor: Extra chip required for functions (such as floating point operations) that are not implemented in certain CPU chip models.

core: The main random-access memory in a computer. The expression originates from the 1960s, when such memory was manufactured from tiny ferrite cores strung on wires.

CPU (central processing unit): The control, arithmetic, and logical unit of a computer. CPU can refer to a complete motherboard in a computer or to just the central processor chip.

crash: Sudden interruption of a program or the operating system from a software bug.

CRT display (cathode ray tube display): Provides visual output of information.

cursor: An indicator on the screen (a line or an arrow) that can be moved around on the screen for pointing to some displayed object. A cursor is moved by keystrokes (as in some text editors) or by some mechanical or optical device such as knobs or a mouse.

cylinder: Most current hard disks contain several disk platters with two heads each. The heads move synchronously and read and write a series of concentric circles called tracks (not spirals like on audio records). A circle that *all* heads can read or write in a single turn of the disk is called a cylinder.

cylinder groups: BSD UNIX and Solaris 2.x organize cylinders into cylinder groups to minimize the average access time. Whenever possible, files are kept within a single cylinder group, such that the disk heads don't have to do large seeks to retrieve or write the data.

daemon: Programs that normally work silently in background and maintain UNIX services. Daemons are periodically active, either upon request or in some constant time intervals. Examples of UNIX daemons are the programs `update` (periodic sync to update the disk), `cron` (performs actions at specified times), and `Acqproc` (acquisition process).

DAT (digital audio tape): DAT drives are available for Sun computers. Such tapes serve as high-capacity storage media. A single tape cassette can hold 2.4 GB of uncompressed data (typically 5 GB compressed) and the transfer data at about 500 KB per second. Sun's 14-GB Exabyte drive (based on 8-mm video technology) is more expensive, but has twice the storage capacity and transfers twice as fast. *See* Exabyte.

data: General term for any kind of digital information, but can refer specifically to the information (numbers, text) a program works on. *data* originates from Latin and is the *plural* of *datum*.

data block: *See* block, I-node.

debugging: Removal of software bugs or errors. The major step in debugging is testing the software. Testing never ends—all software users are also software testers. Each user can help improve the performance of software by reporting bugs. Good bug reporting is very important. Whenever a bug is found, it should be documented thoroughly, including all error messages and the circumstances under which it happened (what was the exact sequence of commands used? What programs were running simultaneously or before the bug was detected?). Any additional observations about a bug may be important. If possible, try to reproduce the bug. And report all of this as soon as possible!

default: Instead of not working if certain inputs are not supplied, a program can also use some internally defined standard input that makes it a useful command without any input or with only partial input, thus saving the user time in entering the command.

degaussing: Due to the large static magnetic field in an NMR system, the monitor housing can become magnetized after some time of operation, resulting in color distortions. The magnetization can be removed by pressing the degauss button on the back of the screen as long as the display oscillates. Many monitors, such as the 17-inch and 20-inch Trinitron color screens on Sun workstations, are degaussed automatically when switched on.

delete: Removal of a file or a file system. UNIX uses the `rm` command to delete files.

desktop publishing: *See* DTP.

desktop storage pack: External module for SPARCstations, containing a disk (2.1 GB), a QIC, DAT or Exabyte tape unit, or a 644 MB CD-ROM drive.

desktop workstation: Workstation where the CPU and display unit usually placed on a desk. The CPU (sometimes including hard disks) normally is placed under the monitor and often has just the same footprint as the monitor or even less. Most current workstations are desktop computers.

destructor: Definition of actions that occur upon deletion of an object in object-oriented programming. *See* OOPS, object.

device driver: Software, built into the UNIX kernel, that allows communication with external devices such as the acquisition computer.

diagnostics: Special software for checking hardware or software. Most computer hardware on a Sun can be checked in the monitor level. For help on monitor commands, type `h` and press Return. SPARCstations have a new monitor mode that uses the ok prompt. Enter `help` for more information.

dialog box: Window on the screen that displays some information and then waits for user input.

directory file: A file containing other files. Unlike older systems, UNIX directories are just a listing of other files on the disk and do not actually contain files. All UNIX files, including directories, are attached to a higher-level parent directory. The top-level directory in UNIX (/) is then attached to itself because the parent directory of / is / itself. *See* text file, binary file.

disk: Non-volatile magnetic storage media for large amounts of data. Used for data in continuous use or currently worked on. For safety reasons (data might be erased or overwritten, or the disk drive might fail), make backup copies of important files.

disk drive: Individual disk unit. Current disk drives used on the Sun for Varian spectrometers are 3.5-inch hard disk drives with formatted storage capacities of 535 MB, 1.05, 2.1, or 4.2 GB (SCSI disk drives), or 4.3 GB (EIDE disk drives).

disk partition: Under UNIX, a hard disk can be partitioned into up to 8 slices, or partitions. The first disk is normally partitioned into a root partition (`c0t3d0s0`), `/usr` partition (`c0t3d0s6`), `/var` partition (`c0t3d0s4`), `/opt` partition (`c0t3d0s5`), and `/export/home` (`c0t3d0s7`). A second disk is usually partitioned into a single slice (`/data`, `c0t0d0s2`).

disk slice: New term for disk partitions. With the transition to Solaris 2.x, Sun started calling disk partitions “slices”. The term is actually misleading, as a slice is a series of consecutive cylinder groups, rather than being slice-shaped. *See* cylinder groups, disk partition.

disk space: The amount of data that can be put onto a disk. In many computer systems (such as the VXR-4000 system), disk size is specified before formatting, so a “140-MB disk” may hold only 120 MB of actual data. On other systems, such as the Sun, disk space is specified after formatting, so a 2.1-GB disk holds 2.1 GB of data. But not all of this space may be apparent to the user. The swap space, for example, is actually a hidden space on the disk. Thus, the apparent size of the system disk is reduced by the amount of swap space.Additionally, in order to maintain disk performance at high levels, UNIX will report partitions being “100% full” when only 90% of the disk space is filled.

display: Usually the CRT screen, but often refers to the displayed information and not the hardware unit.

DNS (Domain Name Service): Software component that translates full Internet host domain names (e.g., `lal750.al.nmr.varian.com`) into a numeric IP address (e.g., `132.190.42.62`). DNS goes to domain name server hosts, who know how to query Internet address from the network, to find out about IP addresses. It then builds up an internal translation table, so that it doesn't have to check the servers for every request.

dot files: Special files for the customization of the user interface. In UNIX, these files have names that start with a dot (e.g., `.cshrc`). In normal directory listings using the `ls` command, these file are considered hidden or invisible, so their names do not show up unless the `ls -a` or `ls -A` command options are used.

dot matrix printer: Printer that composes characters from a dot matrix, as opposed to typewriters and daisywheel printers that produce entire characters from a single impact. Laser printers and ink jet printers are dot matrix printers. Dot matrix printers are more flexible because they can usually print graphics and several fonts.

double-indirect block: *See* I-node.

dpi (dots per inch): A unit of measure for the resolution of dot matrix printers or the pixel density of a display.

DRAM (dynamic RAM): A type of RAM chip, available currently with up to 64 megabits per chip. Dynamic means that it requires periodic refreshing (read and write back) operations. DRAM chips are slower than SRAM chips but are less expensive. DRAMS are used in Sun computers for the RAM memory. *See also* RAM.

DTP (desktop publishing): The creation of complex documentations, such as the VNMR manuals (typically with text, illustrations, tables, index, etc.) on a desktop computer (a UNIX workstation, a Macintosh, or a PC). VNMR manuals are created mainly by using FrameMaker software. *See* FrameMaker.

DVD (digital versatile disk): New, upcoming standard for digital, optical disks. DVD disks have the same form factor as the CD-ROM, but can hold much more information (4 GB and up to 17 GB, depending on the format, compared to 644 MB for a CD-ROM), due to a much higher information density. DVD drives can also read CD-ROMs. Even though CD-ROMs will continue to exist for many years to come, DVD drives will probably replace CD-ROM drives. DVD disks exist in five different formats: DVD-R, DVD-ROM, DVD-RAM, DVD-Video, and DVD-Audio (for the consumer market, no standard has been fixed yet for DVD-Audio). With current drives, double-sided disks (DVD-R, DVD-RAM, some DVD-ROM disks) must be taken out of the drive and inverted by hand in order to access the other side. Current DVD drives have average access times of around 200 milliseconds, and data transfer rates of around 2.8 MBytes/second. *See* CD-ROM, DVD-R, DVD-ROM, DVD-RAM.

DVD-R (DVD-Recordable): DVD disk that can be recorded (written) once (basically a WORM drive), with a capacity of 4 GB per side. DVD-R drives are currently still too expensive for the typical workstation user. *See* CD-ROM, CD-R, DVD, DVD-ROM, DVD-RAM, WORM.

DVD-RAM: DVD disk that can be read and rewritten just like a hard disk, with a capacity of 2.3 GB per side. DVD-RAM disks require special drives that can also read DVD-ROM disks, but DVD-RAM disks cannot be read on DVD-ROM or DVD-R drives. *See* CD-ROM, CD-RW, DVD, DVD-R, DVD-RAM, MOD disk.

DVD-ROM: Read-only DVD disk, with a capacity of 4.7 GB (single-sided/single layer), 8.5 GB (single-sided/double layer), 9.4 GB (double sided/single layer), or up to 17 GB

(double sided/double layer). DVD-ROM drives can read DVD-R, DVD-Video, and DVD-Audio disks, but not DVD-RAM disks. *See* CD-ROM, DVD, DVD-R, DVD-ROM.

EBus (external bus): Bus for peripherals. The EBus is used in the latest PCI-bus-based Ultra workstations and connects to the keyboard, mouse, serial ports, parallel port controller, floppy drive, audio I/O controller, NVRAM and real-time clock, as well as EIDE disks. *See* EIDE, NVRAM.

ECC memory (error checking and correction memory): A device providing self-correction of 1-bit-per-byte errors and detection of 2-bit-per-byte errors. Only available in high-end workstations, such as the Sun SPARCstation 10 and 20, and the Ultra. *See also* SPARCstation 10, SPARCstation 20, Ultra.

editor: A program to modify text and data in files.

EEPROM (electrically erasable programmable ROM): Similar to EPROMs, except that EEPROMs can be erased by applying a specified voltage. Sun computers use EEPROMs to permanently store some configuration parameters that might be altered by the user. *See also* EPROM, ROM.

EIDE (enhanced IDE): Low-cost standard disk type often used in PCs. Ultra 5 and Ultra 10 workstations are equipped with 4.3-GB EIDE disks. *See also* Ultra 5, Ultra 10.

Elite3D: Graphics accelerator used in certain Sun Ultra workstations (Ultra 30, Ultra 60). The Elite3D graphics board provides top-end 3D graphics speed at screen resolutions up to HDTV format (the Elite3D internally performs floating point operations at 1.2 TFlops, i.e., 1.2×10^{12} floating point operations per second). Available with Ultra 10, Ultra 30, and Ultra 60 workstations. *See also* Creator, Ultra10, Ultra 30, Ultra 60.

e-mail (electronic mail): Exchange of information over a worldwide computer network that does not usually involve printing the message. Electronic mail is becoming extremely popular compared to telephone calls, faxes, or ordinary mail, because e-mail is fast (much faster than letters), it takes less effort than writing and sending a fax, and it does not interrupt the addressee's work, but can be replied at the responder's convenience. In addition to all that, digital information (text files, programs, etc.) can be sent from one computer directly to another computer. Typing and handwriting as a source for new errors can be eliminated. The Varian applications laboratories, technical support, and customer service functions can be reached via electronic mail.

emulation: Software simulation of a different hardware and software configuration. For example, standards such as DEC VT-100 have been created for terminals. Many other terminals now have the ability to emulate VT-100 and other standards. The same is true for printers and plotters. Some current standards are HP/GL for plotters and the HP mode, Epson, HP LaserJet, and Postscript standards for printers.

EPROM (erasable programmable ROM): Memory chip that can be erased by exposure to ultraviolet light. The chip can then be programmed again, using a PROM programming device. *See also* EEPROM, ROM.

ergonomics: Working with comfort at a computer for extended periods. Besides a comfortable chair at the correct height of the desk, the positioning of the screen and keyboard is important. To avoid eyestrain from screen reflections, a window or bright objects should not be located behind the operator.

error message: Message that informs the user about software problems or command entry syntax errors. Under UNIX such messages tend to be somewhat cryptic. You may need to ask a UNIX guru what some messages mean.

Ethernet: Xerox trademark for a type of hardware used to communicate between computers. The modulation of a 10-MHz rf signal transmits data, using special (complex)

protocols (characterized as CSMA/CD, Carrier Sensing Multiple Access / Collision Detection). Ethernet uses special coaxial cable (up to about 2.5 km) or standard 50-ohm BNC cable connections (also known as thin Ethernet, only up to 500 m). For transmission, the cable must be terminated with a 50-ohm impedance load on both ends and branching is not allowed. A faster version of Ethernet, called fast Ethernet, operating at 100 Mbps over unshielded twisted pair (UTP, 100baseT) is now becoming available, allowing for much higher network throughput. *See also* fast Ethernet, 10baseT, 100baseT, UTP, Ultra.

Ethernet address: Every device on an Ethernet must have a unique address that consists of a series of hexadecimal characters of the form xx:xx:xx:xx:xx:xx. This address is hard-coded on a PROM into every Ethernet board. In Varian limNET software, this hardware address of the Ethernet board is used. In most modern Internet communication protocols, such as TCP/IP and NFS, a user-assignable software address is used, independent of the hardware address.

Ethernet terminator: Load with 50-ohm impedance on both ends of the Ethernet cable.

Exabyte: Company that manufactures 14-GB tape drives based in 8-mm video technology. The name Exabyte often is used as synonym for the tape drive itself. Such drives are available for Sun workstations. A single tape can hold 14 GB with hardware data compression, and the tape writes and reads data at about 1 MB per second, compared to the 75 to 90 KB per second of a 150-MB 1/4-inch cartridge (QIC) tape drive. Media costs are about 100 times lower per MB than with QIC drives.

executable: A compiled program or a shell script that can be executed by typing its name and pressing Return. The calling user must have execute permission. If desired, other users or groups can be restricted from executing the program or shell script.

exit: Leaving or quitting a computing environment such as VNMR, a shell, or OpenLook.

expansion board: Board added to the computer that enhances the capabilities of a system by adding more memory, more computing power (e.g., co-processors), additional I/O devices or capabilities (e.g., RS-232 expansion board), or higher graphics performance.

expansion memory: Memory module that expands the standard RAM memory. On current Sun workstations, the maximum total memory is between 256 MB and 1 GB.

expansion slot: Free space in a computer that allows insertion of an expansion board. All desktop Sun workstations have special expansion connectors for the SBus (one slot may be used for the color graphics controller). Larger workstations have extra expansion connectors for extra CPU modules.

fast Ethernet: Fast version of the Ethernet protocol that communicates between computers at 100 Mbps, compared to 10 Mbps for standard Ethernet. Fast Ethernet operates over unshielded twisted pair (UTP, 100baseT). The Sun 100baseT Ethernet interface is autosensing. It does both 10 Mbps and 100 Mbps communication on the same network, adjusting itself to the speed of the system it is communicating with. *See also* Ethernet, 100baseT.

fiber-optic cable: Data transfers over computer networks at speeds above 200 Mbps cannot be achieved using electrical cables because of signal losses and rf interference, requiring extensive or expensive shielding. Instead, fast network protocols, such as 622 Mbps ATM, use fiber-optic cable (multimode optical fiber) for the data transmission. This has the additional advantage of providing for electrical insulation between networked computers, avoiding problems with ground loops. Unlike unshielded twisted pair cables, fiber-optic cabling avoids problems with rf interference. *See also* ATM, UTP.

file: For some programming languages, a file is defined as data that can only be read sequentially, with no random access, but now a file more commonly refers to any well-confined amount of data, with any structure.

file access mode: Sets the access protection of a UNIX file (who can read, write, or execute a file) among the three classes of users: the owner of the file, members of a specified group, or all others.

file system: Complex and well-confined system of a directory, its subdirectories, and all the files contained in the system. Under UNIX, a file system normally cannot be extended across disk partitions, although some features, such as the entire file systems mounted onto others, make the user believe that this is possible. Certain commands, such as recursive copying with `cp -r`, work only within local file systems, while others, such as `mv` on directories, work only within a single file system or disk partition.

filter: Program, or sequence of programs, that take input and modify the data for the output. For example, the filter `tr` replaces specified characters in the input with other character. Filter functions are often realized by using multiple pipes (each pipe indicated by a vertical bar), for example, `cat /etc/termcap | grep tvi | more`.

firewall: Software product, typically installed on a gateway computer, that shields a network from intruders. It makes the gateway transparent for users on the local network, while it blocks off most Internet protocols (such as FTP, telnet, rlogin, rsh, rcp, NFS, and HTTP) and only leaves access to simple protocols such as SMTP for e-mail from the other side. A firewall is often used to shield a company-internal network—local users may have full Internet access while outside users can only access people on the local net by e-mail. *See also* gateway.

firmware: Software permanently stored in ROM. On a Sun workstation, the firmware consists of the primary bootstrap and diagnostics software.

flickering: Visible oscillations on a CRT display (typically 1 to 20 Hz). Sun displays should be flicker-free. If this is not the case, look for transformers near the display, perhaps in a printer or other device.

floating point coprocessor: Dedicated processor chip for floating point operations.

floating point number: Number in binary scientific notation, which has a binary exponent and mantissa. In computers, the exponent indicates powers of 2, not 10. This translates into the number of left- or right-shifts in the binary mantissa. Standard formats are 32-bit and 64-bit floating point numbers (sometimes 80- or 128-bit numbers).The 32-bit format uses an 8-bit exponent (–128 to 127) and a 24-bit mantissa, resulting in a range of $\pm 3 \times 10^{38}$, the smallest numbers being in the order of $\pm 1.7 \times 10^{-38}$, with about 7 significant digits. The 64-bit format (IEEE standard) uses 11 bits for the exponent (–1024 to 1023) and 53 bits for the mantissa, resulting in a range of $\pm\ 2 \times 10^{308}$, the smallest numbers being in the order of 10^{-308}, with about 16 significant digits.

floating point operation: Calculation using numbers in binary scientific notation, which have a binary exponent and mantissa. *See also* floating point number.

floppy disk: Soft magnetic disk 3.5-inches in diameter. They are limited in speed and data size (up to 1.44 MB), but handy for transporting small amounts of data.

font: Character set with a specific graphic layout for every character. Dot matrix printers and sophisticated graphics terminals (such as the Sun display) can operate with a whole series of different fonts. Fonts can be of fixed width like most typewriters, where every character takes up the same space, or of proportional width, where the space allocated to each character depends on the natural width of the character.

footprint: Desk space occupied by a device such as a monitor or a CPU unit. Because many users prefer to place workstations directly on the desk, a small footprint has become important, although performance and the price-to-performance ratio are obviously most important. Sun computers combine a small footprint with good performance and price.

foreground process: Process called from some environment and remains attached to that process as a child of the calling process. Such a process is said to run in the foreground, relative to the calling process. Normally, any process can start only one foreground process at a time, and the calling process is then blocked until the foreground is finished or sent into background. Messages and output from foreground processes are sent to the calling process. *Compare with* background process.

formatting: Before a disk can be used, it must undergo a formatting process in which the disk is partitioned into cylinders, tracks, and sectors, and the information where every sector can be found is written onto the disk.

Fortran: The first high-level programming language. Its syntax looks somewhat archaic when compared to modern languages such as Pascal or C, but it still often used for scientific calculations and "number crunching" applications. For the Sun, Fortran is available as an option only.

FPU (floating point unit): Chip or integrated function of a microprocessor that performs floating point calculations. *See* CPU, microprocessor.

fragment: To avoid using too much disk space for small files, the Solaris disk file system can use fragments of 1, 2 or 4 KB for storing small files or remaining parts of large files, while large files are normally stored in 8-KB blocks. *See* block.

frame buffer: Dedicated memory for storing the pixel information for the display. Its size, in bytes, is calculated as *(number_of_pixels* × *bits_per_pixels* / 8). Sun normally uses 1 million pixels (900 × 1152) with black & white displays using 1 bit per pixel and color systems using 8 bits per pixel. Workstations with SX graphics, or those with graphics accelerators for 3D modeling (such as systems with Creator graphics) use 24 and more bits per pixel.

Free Software Foundation, Inc.: Organization "dedicated to eliminating restrictions on copying, redistribution, understanding and modification of software." Created in response to AT&T's licensing of UNIX software, the organization distributes widely used free software, such as gzip (GNU zip), GNU emacs, and GNU C/C++. This software is distributed as source code via public FTP servers such as `prep.ai.mit.edu` (for more information, read the file `/u2/emacs/GETTING.GNU.SOFTWARE` on that host). Because Solaris 2.x no longer has a C compiler included, Varian is using and distributing GNU C with VnmrX for Solaris. *See* GNU, GNU C/C++.

FTP (file transfer protocol): Protocol for transferring files via TCP/IP networks. The command `ftp` implements this protocol. FTP is typically used in interactive mode but only works if a password is defined for the remote account—unlike `rcp`, which only works if a password for the remote system is *not* required (`rcp` requires an insecure remote account or an entry in `/etc/hosts.equiv` or `~/.rhosts` on the remote machine). FTP is mostly used for transferring binary and text files and for downloading software and data over Internet. Public FTP servers usually use so-called anonymous FTP—a special FTP account that requires the user enter the name `anonymous`. There is no password definition for this account, but for logging purposes (FTP logins are logged into a file) the convention is to enter the user's e-mail address as a password. Most Web browsers have a convenient graphical interface for anonymous FTP. Some browsers also allow using FTP for downloading data from non-public FTP servers. These browsers prompt for user name and password, or alternatively these can be entered in the FTP URL. *See also* TCP/IP, anonymous FTP, Web browser, URL.

function: A subroutine in high-level programming languages. In C and related languages, functions have arguments for passing information that is specific to each function call, and functions return a value of a defined type to the calling environment.

function key: Keys (usually F1 to F12) that invoke special actions instead of entering characters. Often the functions assigned to those keys can be changed by the user. Most menu systems work in connection with function keys.

function overloading: In C, functions usually have a fixed number of arguments. Functions can be defined with a variable number of arguments, but this complicated and eliminates some security checking such as argument types. In C++, functions can be overloaded. Several functions can be defined with the same name, as long as each of the definitions, called prototypes, is different in the type or the number of arguments or both. The C++ compiler then matches the proper prototype with each of the function calls and provide full argument type checking. *See* C++.

GaAs (gallium arsenide): Potentially extremely fast chip technology. GaAs transistors switch much faster than silicon semiconductors, allowing for clock rates of several hundred MHz. This technology is not used now in workstations because current technology only allows for a rather low integration level (increasing the number of components), and expensive on-board peripherals, such as memory chips, would be required. *See also* multiprocessor system.

gateway: Machine with network interfaces that link the attached networks. Because all communication between the attached networks must pass the gateway computer, gateways are frequent targets for hacker attacks. In industrial environments, gateways are often equipped with a firewall software that restricts certain Internet protocol levels such that they can only be initiated in one direction. *See also* firewall, hacker.

GIF (graphical interchange format): Standard file format for storage of bitmap files. Identified by a `.gif` extension, GIF files are widely used for images in HTML documents. GIF files can contain sequential display of images as well as interlaced images. In interlacing, images are first displayed with low vertical resolution that is gradually enhanced over several passes. This often permits seeing the contents of an image before the entire file is displayed. GIF is limited to 256 colors, is lossless, and has limited compression capabilities JPEG, an alternate to GIF, uses a more efficient compression mechanism, but is somewhat lossy (loses some data during compression). *See* HTML, JPEG.

gigabyte (GB): 1,073,741,824 bytes. 1 GB is equal to 1024 megabytes (MB) or 1,048,576 kilobytes (KB).

GNU (GNU is Not UNIX): Label used for Free Software Foundation software, which includes GNU C/C++, GNU emacs, and GNU zip (`gzip`). *See* Free Software Foundation, GNU C/C++.

GNU C/C++: C and C++ compilers from the Free Software Foundation. Because Solaris no longer includes a C compiler, Varian includes the Solaris GNU C for use with VnmrJ and VNMR. GNU C is used only for PSG software (`psggen` and `seqgen`) and for the compilation of other source facilities distributed with standard VNMR (such as `wtgen`). The VNMR Source Code Kit requires the SunSoft C compiler (SPARCompiler C or SPARCworks Professional C) and does not work with GNU C/C++. *See* Free Software Foundation, GNU, Solaris 2.x.

graphical user interface (GUI): User-interface software that enables using a computer by addressing (pointing to, clicking, dragging, etc.) graphical objects (icons, windows, scroll bars, buttons, check boxes, etc.) on the screen with a mouse or some other graphical pointing device, instead of using the keyboard for typing commands. The GUI was invented at Xerox for the Star computer and was then taken up by Apple for the Lisa and

Macintosh computers, Microsoft for Windows operating system, OSF for Motif, Sun for SunView and OpenLook, and other companies.

graphics controller: Board (or on-board) function that contains the display frame buffer. This buffer is 1 MB on 8-bit color systems and over 3 MB on 24-bit color systems. On 8-bit color systems, this results in 256 possible numeric values per pixel. The graphics hardware can produce 16,777,216 different colors, $256 \times 256 \times 256 = 8$ bits or 256 values for each of the three colors red, green and blue. One of these 16 million combination is assigned by software to each of the 256 color numbers, and lookup table is used to translate the color numbers from the screen buffer into numeric values for the three DACs for red, green and blue. 24-bit color systems can produce all the 16,777,216 different colors simultaneously.

graphics workstation: High-performance stand-alone computer with high-resolution and fast graphics capabilities. Graphics workstations usually have multiuser capabilities but are equipped with a single high-resolution graphics terminal.

ground loop: Ideally, all grounds should originate at the same point, and ground loops should not be produced by having several grounding connections between two devices. Ground loops can lead to electrical oscillations, which can produce nasty interferences and random signal distortions that are difficult to cure. The simplest way to avoid ground loops is to have the shielding of all electrical cables only connected at one end (ideally to the central device, not to the peripherals) and to have one single, well-defined grounding connection between all devices.

GUI: *See* graphical user interface.

guru: Somebody who has worked with UNIX so much that the person can solve the most difficult problems from experience and may even understand UNIX error messages!

hacker: Real computer freak who is in a permanent competition for the “best,” most complex, most useless, and most cryptic program. This person would like to become a guru, but often ends up trying to prove to others that he or she is able to break into anybody’s data files and software.

handshaking: Most interfaces require several lines for handshaking, which is a type of hardware or software protocol for data transmissions. For example, the receiving device tells the transmitter that its buffer is empty and that the transmission can start. With most bus traffic, the handshaking is a part of the bus definition, and the user should normally not be concerned about it. An exception is that in RS-232 communication, data flow is controlled by handshaking signals, but there are several handshaking options, unfortunately. Many RS-232 devices allow selection of different handshaking options. Often, unreliable data transfer via RS-232 occurs either due to improper setting of the handshaking options on one of the communicating devices or due to improper cabling of the handshaking lines. The simplest form of RS-232 handshaking is the back-transmission of special characters XON and XOFF to start and stop the data flow. More elaborate handshaking protocols involve dedicated lines. *See* null modem, XON/XOFF, RS-232.

hangup: Describes the status of a program that got trapped in a loop and can only be accessed by stopping the program. *See* interrupt, kill.

hard disk: General expression for rigid disks, such as SCSI hard disks and similar devices, as opposed to floppy disks that are commonly used on personal computers. *See* Winchester.

hard links: UNIX allows multiple entries in a directory to point to the same file. All these entries are equivalent and are called hard links. The file pointed to is only removed when all hard links are removed. The hard links to a single file can even be in different directories, but this is not a recommended practice because it destroys the tree structure of a file system. More than one hard link to a directory file is not permitted, because this would allow

forming a loop in the file structure, which violates the tree structure and the hierarchical organization of directories. Hard links are not permitted between different partitions. *Compare with* symbolic links.

hardware: Basically, anything in a computer you can touch.

head crash: The heads writing onto or reading from hard disks fly only a few micrometers over the disk surface rotating at 5400 or 7200 rpm. A head crash occurs when any particle of dust, smoke, or fingerprint collides with the head. This can destroy the head and the disk so that all data on the disk are lost, and the disk has to be replaced. Today's hard disks are sealed; therefore, such events should occur very seldom.

help: Assistance for operators who do not know how to use a system. Help is available from manuals (hard copy and online) and from experienced users, such as a guru.

hierarchy: Hierarchical, parent-and-child structures are involved in UNIX software as well as in UNIX file systems. *See* hierarchical file system.

hierarchical file system: Storing files such that information is stored at different levels. The UNIX file system is hierarchical in that each location in the file system is called a directory and each directory is a collection of files and other directories, forming what is called a tree structure.

high-level language: A programming language, such as Fortran, Pascal, C, or C++, that takes an abstract view of programming a computer, in contrast to an assembly language that relates to certain computer systems only.

history: Stack register for the last command lines. Can be used to reexecute entire command lines without retyping them or to reconstruct the sequence of previous commands in case of problems with software.

home directory: The directory that becomes the working directory immediately after login. For normal users, it is also the directory where all or most personal files are located. The home directory is specified in the file `/etc/passwd`.

host: Computer that controls other computers. For example, on Varian spectrometers with Sun workstations, the acquisition CPU, a computer on its own, is controlled by the host, which is the Sun CPU. In UNIX, the term host is also used for any computer in a network.

host name: Each host computer is given a common but unique name, which makes easier for users to address it because they don't have to type or learn a complex sequence of numbers or funny characters when talking to other computers.

host number: A number by which the software addresses and recognizes another CPU internally. Every CPU in a network must have a unique host number and name.

HotJava: Sun's Web browser. HotJava is written entirely in Java and, of course, it fully supports Java applets. *See* Java, Web browser, HTML.

HP-GL (Hewlett-Packard Graphics Language): Internal computer terminology that defines the output on HP plotters. All HP plotters use HP-GL, and now the HP LaserJet III, 4 and 5 laser printers can interpret HP-GL/2, a terser version of HP/GL. HP-GL/2 speeds up data transfer times by a factor of over four compared to sending the entire dot matrix (about 1 MB per page at 300 dpi, or 4 MB at 600 dpi). *Compare with* Postscript, dot matrix printer, plotter.

HTML (HyperText Markup Language): A simple formatting language that permits formatting text in a Web browser, such as Netscape, HotJava, or Internet Explorer, by inserting simple ASCII formatting codes into the text. The codes can be added by any standard UNIX text editor. The formatting is *relative.* The HTML author selects styles such as heading levels, emphasized, strong emphasis, and relative font sizes, and some primitive

paragraph styles, tables, list formats, etc. The actual formatting only occurs in the viewer's browser. The font selection, width of the window, and many other formatting variables. are under the viewer's control and can be adjusted to match the viewer's screen size and viewing convenience. The real power of HTML lies in the use of hypertext links that permits incorporating any Web document that is accessible anywhere in the world into a local document. This document will have further hypertext links, building an enormous virtual library that can be reached and viewed through simple mouse clicks. HTML also permits incorporating pictures, sound, animated and interactive objects into a Web page. This can be a simple image (a GIF or JPEG file) or an animated audiovisual object defined using the Java programming language (which can be incorporated into HTML documents). *See also* WWW, Hypertext link, Web browser, URL, GIF, JPEG, Java.

HTTP (HyperText Transfer Protocol): To retrieve information or data from another machine on the Web, a Web browser needs to invoke a dedicated protocol. This protocol is specified in the URL of any remote hypertext link. For HTML documents from a Web server, the protocol is HTTP. On UNIX Web servers, HTTP is installed by starting special Web server software, such as `httpd` (the HTTP daemon). HTTP has become the dominating protocol of the total Internet traffic, overtaking the FTP and e-mail packet protocols. *See* HTML, URL, Hypertext link, Web server, FTP.

hub, Ethernet hub: Connection point for several 10BaseT or 10/100BaseT Ethernet connections, used to build star-shaped network topologies using twisted (STP or UTP) pair cabling. Hubs are typically equipped with 4 and up to 16 or more RJ-45 connectors; several hubs can also be interconnected for more complex network topologies. *See* 10BaseT, 10/100BaseT, Ethernet, RJ45, STP, UTP.

Hypertext: *See* HTML.

hypertext link: Any portion of an HTML document—including text and images—can be defined as a hypertext link for which an URL defines a different location in the same document, in a different document, or a document on any remote hypertext server that can be reached on the Internet. Clicking on the link from within a Web browser retrieves and displays the document (or the portion of a document) that the hypertext link points to. *See* HTML, URL, Web browser.

I-node (index-node): UNIX directories consist of I-nodes (index blocks) and names. Under Solaris and 4.2BSD UNIX, I-nodes are 128 bytes long and the names can be any length. Information in the I-node includes permissions, owner and group, number of hard links, dates of creation and last modification, addresses of the first 12 data blocks (8 KB each), address to the first indirect block (with the addresses of the next 1024 data blocks), and the address to a double indirect block (with 1024 addresses of further indirect blocks). Therefore, the theoretical maximum size of a single data file in 4.2BSD UNIX is about 8 GB. For small files, the I-node can point to fragments (blocks of 1, 2 or 4 KB).

I/O (input/output): Input and output of data, mostly used for describing printing and plotting (for output), and mouse and keyboard entries (for input). Input is the transfer of information into a computer; output is the transfer of information out of a computer into a form that can be read or used by human beings or computer programs. UNIX, VnmrJ, and VNMR programs, as well as shell scripts and macros, can also take input from other programs.

icon: Small picture or symbol on the screen that represents a window that is closed, or currently not shown. Unlike with most personal computer software, UNIX events can also happen inside windows. Programs, even graphics programs, can run and icons can be active windows, the window with the focus, and take keyboard input. The icon itself can act as a small graphics window, such as the clock tool, for example.

IDE (Integrated Device Electronics): Common low-cost standard hard disk types. *See also* EIDE.

indirect block: *See* I-node.

ink jet printer: Dot matrix printer that generates dots by spraying minute droplets of liquid ink onto the paper. Obtainable resolution is up to 360 or more dpi (dots per inch).

insert mode: In text editing programs, a special mode in which the characters entered on the keyboard become text and are displayed on the screen. Some editors (such as the UNIX `textedit` program) operate exclusively in the insert mode (commands are available via menus). In `vi`, the most popular editor under UNIX, the insert mode has to be exited by pressing the Esc key in order to enter commands. *Compare with* command mode.

insertion point: In text editors, the point where text typed on the keyboard is inserted into the text, after going into the insert mode.

installation: Sun workstations are normally installed by qualified service personnel in order to avoid damage to the equipment. Should you plan to install it yourself, carefully follow the Sun and Varian installation manuals.

integer: On the Sun, an integer is normally a 32-bit integer number and can have values between –2,147,483,648 and 2,147,483,647. A short integer is 16 bits and can have values between –32,768 and 32,767.

integration: The integration level refers to the complexity of a single chip as well as to the number of chips and other devices on a single computer board.

interface: Device that links two devices, provides the necessary translation, and handles the required protocols on both sides.

interlacing: The cathode ray in the CRT display switches between the even and odd display lines instead of covering each line in succession. As a result, each pixel is refreshed with half the display refresh rate, thus reducing display hardware cost. To achieve an optimum, flicker-free display, Sun CRT displays do not use interlacing.

interleave factor (IL): Number that characterizes the distance of two sectors on a disk that can be read or written subsequently. IL = 1 means that all sectors can be read one after another (1, 2, 3, 4, ...). IL = 2 means that the sequence of sectors to be read or written on a disk with 17 sectors is 1, 3, 5, 7, 9, 11, 13, 15, 17, 2, 4, 6, 8, 10, 12, 14, 16, and so on, for higher interleave factors. The size of the interleave factor is dependent on the disk controller and on the type of disk used. Obviously, IL=1 is the fastest solution, because an entire track can be read in a single disk revolution (IL is also the number of turns that are required to read a complete track). The SCSI disk drives used in Sun workstations use IL = 1.

Internet: In 1969, the U.S. Defense Advanced Research Projects Agency (DARPA) funded a project for creating an experimental packet-switched network that would still be functional after losing some individual connections or network nodes. Different from a telephone network, which is connection-oriented, the new network would send information in short packets. The network software (protocols) would automatically split and recombine these packets, which are all individually and automatically routed over the network. A packet loss would cause automatic retransmission of the packets. In 1975, the network, ARPANET, became functional, but network research and protocol development continued further, and in 1983, TCP/IP were adopted as military standard. At that time, the term Internet started being used for combination of MILNET (the unclassified part of the defense data network, DDN) and the smaller, classified ARPANET. ARPANET was discontinued in 1990, but the Internet had in the meantime grown enormously. Internet now refers to the sum of the world-wide, interconnected networks that use the TCP/IP protocol

family. Until recently, the vast majority of the packets travelling over the Internet were e-mail and FTP packets, but by 1996, HTTP packets have become the dominating portion of the overall traffic, due to the popularity of the World Wide Web. Sometimes, the term Internet is incorrectly used as synonymous with World Wide Web, which specifically refers to the part of the Internet that is accessed via Web browsers. *See* WWW, TCP/IP, FTP, HTTP.

interpreter: Program that reads source code and translates it directly into machine instructions for the CPU. There are also interpreters that read semi-compiled code such as P-code. Interpreted programs always run slower than compiled programs. A special form of interpreter is the command interpreter such as the C shell and Bourne shell.

interrupt: Signal that interrupts the execution of software. Interrupt signals might be generated from the operating system or by the operator (by pressing some special keys). An interrupt might be pressing Ctrl-c to interrupt the execution of a particular program or pressing Stop-a to interrupt the CPU.

intranet: While the World Wide Web is used more and more to do business and share information with potential and actual customers, companies have realized that Web servers are also an excellent tool to share and distribute information internally, even a tool to access internal resources such as databases. Such facilities, called an intranet, are usually installed on internal networks that are not accessible to the public. Intranets are now rapidly gaining popularity all over the world. *See* WWW, Web server, Internet.

invisible file: File that is normally not shown in directory listings except if special commands or options are used, such as the dot-files in UNIX.

IP (Internet Protocol): Network protocol that is the basis of many higher level protocols, including TCP and UDP. The IP protocol provides a basic software connection at the information transfer layer. Error checking is done at a higher levels such as TCP or UDP. *See* TCP, UDP, TCP/IP.

ISDN (Integrated Systems Data Network): A new standard for digital telephony that is rapidly being adopted and installed, especially in Europe. A basic ISDN connection offers a total bandwidth of 144 Kbps, split into two data channels of 64 Kbps each, and a control channel with 16 Kbps. There is also a primary connection option, offering 30 data channels with 64 Kbps each, almost 2 Mbps, and a control channel with 64 Kbps. An ISDN connection can be used for both digital and analog telephony. Direct digital PPP connections can be established at 64 or 128 Kbps via ISDN, provided the two computers are equipped with an ISDN interface. Such ISDN connections are faster and more reliable than standard modem connections. ISDN interfaces are available for Sun and UNIX workstations as well as PCs and Macintoshes. The ISDN interface is standard on the SPARCstation 10. For other Sun workstations it can be implemented with s a SBus expansion card. Using an ISDN port requires Solaris 2.x and a SunISDN software enabling kit. *See also* modem, PPP.

ISP (Internet Service Provider): Institutions and companies that provide Internet connectivity and operate the international Internet connections. Note that some ISPs only serve academic institutions (a part of the Internet is operated with public funding), while others operate in the commercial segment. *See* Internet.

Java: Object-oriented programming language in which the VnmrJ interface is written. Java is similar to C++ and was developed by Sun . The main objectives in creating Java were compactness of the executable file, security, and portability. For security, dangerous language elements (i.e., elements that could be used to create viruses, such as pointers) from C++ were removed, and new security features were built in. Typically, Java is compiled into very compact byte codes that are platform-independent, sent over the Web, and then interpreted by the Web browser. The intent is to download not just HTML code

over the Web, but small applications, called applets, that can animate Web pages. In the future, entire applications, including word processing and database software, might be kept on a central intranet Web server and distributed as applets that run on any platform. This dramatically simplifies the administration of desktop software. For performance-critical applications, there are also Java native compilers that produce machine language executables, but that makes the application nonportable, of course. *See* OOPS, Sun, C++, applet, WWW, HTML, Intranet.

JavaScript: JavaScript was introduced by Sun and Netscape and complements the Java programming language. Java is compiled into byte codes and downloaded over the Web, whereas JavaScript code, which resembles Java source code, can be incorporated directly into the HTML code for a Web page. The JavaScript is then interpreted along with HTML by the Web browser. *See* Java, HTML, applet, Sun, Netscape, Web page.

JPEG (Joint Photographic Experts Group): Standard file format for storing bit-mapped graphics such as scanned images. JPEG format images are often used for images in HTML documents. JPEG is has no limitation in the number of colors and includes good compression but there may be some image quality losses due to JPEG compression, which is lossy. JPEG permits storing images with two-dimensional interlacing. Such images are first displayed with very little detail and are then gradually refined in several passes. *See* GIF, HTML.

jukebox: Device that contains a series of optical disks but only a single optical disk drive. It changes disks automatically and efficiently, such that huge amounts of data (up to several terabytes) are accessible within less than a minute.

jumper: Normally a small rectangular piece of plastic that sticks on two pins to create an electrical bridge used to configure computer boards for different environments. Jumpers are often also used to set the hardware address for devices, like disks and disk controllers, that are attached to a bus (e.g., the AP bus in a spectrometer or the SCSI bus).

kernel: The program block that always resides in memory, a central part of the UNIX operating system. The kernel is never swapped out which in turn reduces the amount of RAM available to application programs.

keyboard: Sun workstations come with an American standard (QWERTY) keyboard. Solaris supports an 8-bit character set; therefore, non-US keyboards can, in principle, be used on Sun workstations.

kill: Stopping the execution of a program. Under UNIX. the `kill` command allows stopping applications programs as well as daemons or any running process.

kilobyte (KB): Corresponds to 1024 bytes.

LAN (local-area network): A computer network in the same building or same site

LaserJet: Trademark for a laser printer from Hewlett Packard. *See* laser printer.

latency: On disk drives, the time to access a sector after the right track has been accessed. This time is influenced by the interleave factor. The rotational latency is inversely proportional to the speed of rotation. *See* seek time, zero latency.

laser printer: Dot matrix printer that uses a pulsed laser beam to produce the individual dots that form the output. *See also* LaserJet.

Lexmark: Manufacturer of printers. *See also* HP/GL, PostScript.

license manager: Many licensed software packages are not licensed on a per-installation basis, but rather for a certain number of users within a network. The main reason for this is that in many locations the software will be installed on a central server only, and several users will use the same software. In such situations, license checking is done via license

manager software, a dedicated piece of software usually delivered with the licensed software package. When a user launches software that uses a license manager, this software first calls the license manager, which searches the network for other computers and users running the same software package. This process usually takes less than a minute, and then only the actual software starts. To eliminate this extra time every time a software package is needed, users can just close the window into an icon, but leave the software running for as long as they are logged into a system.

line conditioner: Device that reduces or eliminates surges and spikes in the ac power supplied to the system. A line conditioner is required where the mains power supply suffers from serious surges or high-voltage spikes.

link: *See* symbolic link, hard link.

linker: A program that links object files (precompiled program modules) to for an executable program. *See* object file.

login: Before being allowed to use UNIX and most other operating systems, the user must follow a login process by providing a user name and a password. If the user name and password are not typed in correctly, the multiuser environment cannot be entered. If the user name is not known to the system, it asks for a password anyway, so it is not obvious whether a user has not been defined or whether just the password was wrong. Only the superuser can create new users and reset forgotten passwords.

login shell: Program that handles the login (checking for user name and password), thus preventing outsiders from entering into the multiuser mode. *See* login.

logout: After finishing a working session on a multiuser system such as UNIX, a user should complete the logout process so that another person can only use the system when passed through the login shell. *See* login, login shell.

macro: VnmrJ and VNMR term for a file that consists of a series of VnmrJ or VNMR commands. A VnmrJ or VNMR macro is called by entering the name of the macro file and is interpreted like a series of commands entered on the keyboard. Macros allow for a series of programming utilities such as branching, loops, and arguments, etc. They are useful for customizing and simplifying repetitive tasks. The same is possible under UNIX with shell scripts.

MAGICAL (MAGnetics Instrument Control and Analysis Language): Varian Inc. trademark for system software that supports full automation of spectrometer operation and analysis. MAGICAL enables the spectrometer to make decisions during an NMR experiment, guiding data acquisition, processing and plotting, and even analysis of the data. Key functions are contained in macros and menus that enable the user to acquire and analyze NMR data or simplify complex or tedious operations with a single command.

mail: UNIX includes tools for mailing messages and data between users on a multiuser system or between users of different systems in a computer network. *See also* e-mail.

magneto-optical disk: *See* MOD drive.

maintenance contract: In a time of rapidly evolving software, it is important to keep system up to date. Your Sun UNIX and Varian Inc. VnmrJ and VNMR software is expected to make major progress each year, and a software maintenance contract from Varian keeps both up to date. It should also be kept in mind that UNIX and VNMR really work together. There is no guarantee that the newest VnmrJ and VNMR release will work with a much older UNIX release or that the newest UNIX will work with an older VNMR release.

manual: Manuals are important to users in at least two stages of working with the system. The beginner needs the more elementary parts of manuals to get started and to learn about the basics of the software and hardware. Later on, the manuals are valuable to the

experienced user, because nobody can learn all the software features by heart. At this stage the online manuals are even more important than those on paper. Both UNIX, VnmrJ, and VNMR have a `man` command for online help. The UNIX `man` command displays an exact copy of the required descriptions from the Sun UNIX manual if the man pages are loaded. These texts are formatted on the spot using `nroff`. Preformatting of all the manual text, at the expense of several megabytes of disk space, is possible with the `catman` command. This speeds up the display of the manual pages and reclaiming the disk space lost by `catman` can be avoided by deleting the original `nroff` files `/usr/man/man*` afterwards.

mass storage: Device such as a hard disk or tape for storing large amounts of data.

matrix printer: *See* dot matrix printer.

MC68040: 32-bit CISC CPU from Motorola, developed from the MC68030 by adding an on-chip floating point unit and increasing the instruction and data caches to 4 KB each. The MC68040 is available at clock rates up to 40 MHz. Integer performance is up to 44 MIPS. The MC68040 is used in the UNITY*INOVA* acquisition computer. *See* , Motorola.

megabyte (MB): 1,048,576 bytes, which corresponds to 1024 kilobytes (KB).

megaflops (MFlops, million floating point operations per second): An older performance measure for computers. The MFlops rating is determined by either the Linpack (typical for Fourier transformations) or the Whetstone (typical for most other tasks) benchmark test. Whetstone values are typically 6 to 8 times larger than Linpack. Because most operations in VNMR occur as floating point calculations, the performance of a computer in MFlops is more important for VNMR users than the computer speed measured in MIPS; however, the MFlops specifications are gradually being replaced by the SPECfp suite of performance tests. *See* SPEC, SPECfp.

menu: Software assistance by offering a list of options. Usually the selection of an option is made by pressing a function key, entering a character or sentence, or by selecting an option with the mouse. VnmrJ and VNMR use a menu system that allows for a wide variety of actions through menus. Menu selections in VnmrJ and VNMR can be interspersed with keyboard commands. Despite its power, the structure of the menu system is simple and safe. It offers optimum help for both the beginner and the experienced user. In addition, the menu system is flexible and can be modified by any user. *See* menu button.

menu button: With mouse-oriented user interfaces, like on Sun workstations, menu options often are button-like fields on the screen that are selected either by function key or by moving the mouse on the selected field and then pressing the left mouse button.

mezzanine board: A piggyback-type expansion board that plugs into a motherboard or expansion board via standard connectors. Mezzanine boards are mounted flat on a motherboard and are often used for VME board expansions. The UNITY*INOVA* pulse programmer can be expanded via a mezzanine board. *See* VME, piggyback board.

MFlops: *See* megaflops.

microprocessor: Complex CPU chip that powers personal computers, workstations, server systems, and even many large scale computers. A microprocessor is almost a little computer of its own in that it combines many components that before were placed in separate chips. Only the advances in chip technology and the growing degree of integration make it possible to put all this into a single integrated circuit. Twenty years ago, the first popular microprocessors were 8-bit processors such as the 8080 and the Z80. The current top-of-the-line models are 64-bit processors. Early models essentially consisted of an integer calculator (the arithmetic logical unit, ALU), a small set of registers, and I/O control functions. Modern microprocessors integrate several million transistor functions on a single chip and include several integer units for parallel execution, a large set of registers,

one or several floating point units (FPU), a set of primary cache resisters, a memory management unit (MMU), and a controller for external cache. They not only operate at clock rates up to 200 MHz and more, but they also execute several commands in parallel. *See also* CPU, ALU, FPU, MMU, 16-bit computer, 323-bit computer, 64-bit computer.

MII (Media Independent Interface): Ethernet connector and cabling standard that permits connecting transceivers for coaxial cabling via AUI connectors, fiber-optic networks, unshielded twisted pair (UTP), using the appropriate connector and transition cable. Most modern Ethernet interfaces offer both MII and RJ45 (for 10/100BaseT) connectors. *See* AUI, Ethernet, UTP.

mini UNIX: Minimal UNIX configuration for installing the full Sun software, loaded into the swap partition under Solaris 1.x. For Solaris 2.x, software is installed from an entirely memory-based version of UNIX.

MIPS (million instructions per second): Mostly determined as VAX-MIPS, where the performance of a VAX 11/780 (running a specific program, the so-called Dhrystone integer performance test) is 1 MIPS. The Dhrystone and MIPS specifications has been replaced by the SPECint performance test. *See* SPEC, SPECint.

MMU (memory management unit): Hardware function used to obtain virtual memory capability, which allows handling of very large data tables with limited RAM. On current SPARC CPU chips, this function is integrated into the CPU.

MOD drive: Magneto-optical disks are removable optical disk media (similar to the CD-ROM) that can be rewritten many times. The term “magneto-optical” describes the working principle of these storage media: for writing such disks a strong laser beam locally heats the surface in a magnetic field, locally changing the magnetization of the information layer. The magnetization changes the local optical properties of that layer, such that a weak laser beam can read back the information. Even though MOD disks are very versatile storage media with a capacity of up to 1 GByte or more per side, no format standard for MOD disks was ever defined (which lead to a diversity of incompatible formats), and no major manufacturer built such drives. The DVD-RAM may soon replace the MOD drives. *See* CD-ROM, DVD, DVD-RAM.

modem (modulator-demodulator): Transmission of binary data, most often via telephone lines, by modulation of a low frequency, usually below 10 KHz.

monitor: As hardware, a monitor is a CRT screen for displaying text and graphics information. As software, the monitor level is a built-in PROM-based diagnostics and start-up software on Sun workstations. When a Sun is switched on, the monitor diagnostic is called and started automatically. It resets the hardware, does a series of hardware tests, such as testing memory, and calls a boot program to boot up UNIX. Before switching off a Sun workstation, the system should be run down to the monitor level using the `init 0` command, which makes sure all software is terminated properly.

Mosaic: The first graphical Web browser. While hypertext and HTML had already existed for some time, the early browsers were ASCII-based only. Mosaic was developed at the NCSA (the National Center for Supercomputing Applications) and became commercially available in 1993, drawings millions of user to the Web. More recently, Netscape, a more advanced graphical browser that has originally grown out of the same NCSA environment, has become more popular than Mosaic. *See* WWW, Web browser, Hypertext, HTML, Netscape.

motherboard: The main CPU board, especially on computers that in essence consist of a single printed circuit board. The term was probably chosen because a motherboard often houses a number of additional plug-in extensions, often called daughter boards.

Motif: X Window System graphical user interface as proposed by the OSF (Open Systems Foundation). Motif is used by IBM and other companies. *See* X Window System, CDE.

Motorola: Computer and chip manufacturer that manufactures the MC680x0 family of CISC CPU chips and the PowerPC series of RISC CPU chips. *See* , MC68040, CISC, RISC,

mount directory: Normally empty directory on which other disk partitions or directories can be mounted. The mount process hides the contents of the mount directory such that after mounting, the `ls` command displays only the files of the file system that have been mounted, not those of the mount directory. *See* mounting.

mounting: UNIX facility that allows making other file systems part of the currently accessible file system. The file systems are normally entire partitions on the same disk, on other disks, or even on a remote or server disk. After mounting, these partitions can be accessed like any other part of the file system on which they are mounted. The `/usr`, `/export/home`, and other disk partitions must be mounted before they can be accessed. The mounting normally happens automatically during the bootup procedure.

mouse: Hand-held unit with one or several buttons that is moved around on the desk or on a special board for positioning a cursor on the screen. Pressing a mouse button is called clicking. Pressing a mouse button and then moving around the mouse while holding the button down is called dragging. The mouse detects movements either mechanically through a ball or rollers or optically through reflection of a light source on the mouse. Optical mice work only when moved on a special pad placed in landscape orientation. Mechanical mice operate on almost any surface. Sun has recently switched from an optical mouse to an mechanical mouse.

mouse buttons: Sun software requires a mouse with three buttons. The left button (when the cable is pointing away from the user) is the main button, which activates functions by clicking on them. The right button is usually the menu button, used to call pop-up menus, but it might also be used for other purposes. The middle button is usually used to drag graphics around or to set a size of some displayed item.

mousepad: A surface for the mouse to move on. Required for optical mice, but optional for mechanical mice. *See* mouse.

MPEG (Motion Picture Experts Group): Standard file format for compressing and storing movies as digital video data. MPEG compression can be used in multimedia Web applications and Web pages, although currently for many Web users the Internet access speed is to slow to allow for good quality online video transmission, even with MPEG. Software that plays MPEG movies is not standard on most systems yet, therefore most Web browsers only download MPEG files for playback later. *See* HTML, JPEG.

MT: *See* multithreading.

MTBF (mean time between failures): The length of continuous operation expected for hardware, usually in thousands of hours, giving a measure of the reliability of hardware.

multimedia: Applications that present text with animated visual or audiovisual information, such as a Web page using sound and a video player. *See* MPEG, WWW.

multiprocessor system: Computer that uses several CPU boards or CPU chips in parallel. Standard workstations only allow for pseudo-parallel execution of processes by extensive time-sharing on a single CPU. In its simplest form, a multiprocessor system allows for true parallel operation by distributing processes onto the different CPUs. This mode usually does not require changes in the individual programs. Special compilers or programs that are written with multithreading also allow a single process to access several CPUs in parallel. *See* multithreading

multitasking system: Operating system that can handle and run several programs simultaneously. This normally includes time-sharing between the different active tasks, with scheduling according to the individual task priorities.

multithreading: Programming technique using special multithreading utilities that takes advantage of multiple processors if present. Programs written without multithreading are not faster if run on multiprocessor systems, while applications written with multithreading can spread their load onto different processors. Of course, even without multithreading, multiprocessor systems can still distribute the different processes over several processors, which certainly helps in multiuser and server situations.

multiuser system: Operating system that allows for several users to be logged in at the same time on the same CPU. Ideally, such an operating system also handles the communication between the various users through electronic mail and other online communication utilities such as `mail` *and* `talk` on UNIX. A multiuser system also implies that a single user can operate several "pseudo-terminals" in separate windows (or on separate terminals) simultaneously.

Navigator: *See* Netscape.

NCSA Mosaic: *See* Mosaic.

network: Group of interconnected computers and peripherals communicating with each other.

Netscape Navigator: Currently, the most popular graphical Web browser. Netscape was created by a group of people who developed the Mosaic browser and who formed a new company to market a better browser. While HTML has made steady progress and has evolved over the years, Netscape has always been ahead of the crowd by new features such as tables, frames, and more font and background options. Netscape Navigator was one of the first Web browsers to support Java applets and JavaScript, and they are among the first ones to support Sun's Tcl/Tk language utility. Netscape Navigator is written in a way that permits adding new facilities easily via plug-in software, such as the Tcl/Tk plug-in. *See* WWW, Web browser, Hypertext, HTML, Java, JavaScript, Tcl/Tk, Mosaic.

NIS (Network Information Service): Sun's database software for networked workstations that facilitates the system administration, including sharing password files, network address files, and host libraries. Prior to SunOS 4.1, NIS was named "yellow pages" (this name still shows up in some command names) but the name was changed due to a trademark conflict.

NFS (Network File System): Sun's high-level protocol for disk sharing via Ethernet. Sun software allows accessing remote disks the same way as local disks. Remote disk partitions are mounted onto local directories and, thereafter, behave like a part of the local file system. NFS has been adapted by most UNIX systems.

node: Individual computer in a network. Every node has a dedicated, unique address and most have a name such as varian, fred, or jane.

non-interlacing: *See* interlacing.

null device: Receives output and then discards it. Sometimes called a bit bucket. In UNIX, the null device is the file `/dev/null`.

null modem: Serial ports, such as RS-232 and RS-423, can be configured in terminal or modem mode. In the connector for terminal mode, line 2 is the transmitting signal and line 3 is the receiving signal. In the connector for modem mode, these pins (as well as all the handshaking lines) are interchanged, such that a straight-through cable can be used. Often, both ports are configured in the terminal mode. In such a case, lines 2 and 3 have to be crossed for proper communication and the handshaking lines must be rearranged. The

device that interchanges signal and handshaking lines between two ports in terminal mode is called a null modem or null modem cable. This device can be a cable (but not with straight-through wires in this case) or a little box that is inserted between one of the ports and the cable. If printers, plotters, or terminals are hooked up to the ports on the Sun CPU board, a null modem is normally required. For exact instructions, consult Sun manuals or a Varian specialist. Often, such as on an RS-232 expansion board, serial ports can be configured such that a null modem is not required. The two most common types of null modems are those using software XON/XOFF flow control and those with hardware handshaking. An XON/XOFF null modem has pin connections 1–>1 (optional), 2–>3, 3–>2, and 7–>7. Null modems for hardware handshaking have pin connections 1–>1 (optional), 2–>3, 3–>2, 4–>5, 5–>4, 7–>7, 6+8–>20, and 20–> 6+8. Because the hardware type is fully compatible with the XON/XOFF flow control null modem, the hardware type of null modem is preferred. In rare cases, other special types of null modems with different wiring may be required. Ask Varian specialists for advice.

null modem cable: Cable or device that interchanges transmit and receive lines, and the corresponding control lines, between two equally configured RS-232 ports, both terminal or both modem. Usually a null modem cable consists of a normal RS-232, or V.24, cable and the null modem, a box that is inserted between the cable and one of the two ports. *See* null modem, RS-232.

NVRAM (non-volatile RAM): Battery-buffered RAM used to store system configuration and bootup parameters. Earlier workstations used EEPROMs instead. *See* EEPROM.

object file: A compiled program module, typically a file with a " . o" extension. In software that consists of many source files or modules, each source file is usually compiled separately, and an object file is created for each of them. These object files then are linked with each other through a linker, and with additional modules from C libraries and other libraries, such as graphics functions, in order to create an executable file. *See* OOPS, class.

object-oriented programming style: *See* OOPS.

objects: Essential element within the object-oriented programming style (OOPS). Objects in programming have internal properties, consisting of private elements that cannot be accessed by the calling program, and well-defined external properties, consisting of public elements that can be accessed by the calling program. An object is an instantiation of a class. The class defines the properties of the object, such as a type definition. An object might be something like a complex variable of the type class. Objects exist in the computer memory for a well-defined lifetime, and it is possible to define operations that are performed upon construction and destruction of an object. *See* OOPS, class.

off-line: Submitting a task for processing in which control over the task and timing is transferred to the operating system. Off-line tasks in UNIX include mail, printing, and plotting.

online: A process under direct control and access of the operator. Disk-based manuals are online help for the operator, and any processing invoked and controlled at run-time by the operator is called online.

OOPS (Object-Oriented Programming Style): Programming style that uses objects. The objective of OOPS is to make programming safer, easier, and more portable. It makes programs easier to maintain, and it facilitates cooperation between programmers and programming groups because it enforces clear definitions. Object-oriented programming is facilitated, but not necessarily enforced, in certain programming languages such as C++ and Java. *See also* object, class, C++, Java.

open access: Any user can access a file. Under UNIX, for security reasons, open access files are limited to specific disk partitions. *See* permission.

OpenGL: Application programming interface (API) for creating interactive 3D graphics visualization, simulation, and animation applications, such as molecular modeling software. OpenGL was introduced by SGI and was one of the ingredients that made SGI workstations popular in the scientific visualization market. OpenGL is now becoming available on Sun platforms (Solaris OpenGL). The first version is for systems with Creator3D graphics only, but later versions will run on any platform (with lower performance). Solaris OpenGL should bring a new range of software products, such as molecular modeling, onto the Sun platforms. *See* API, SGI.

Open Systems Foundation: *See* OSF.

OpenWindows: Sun's complete X Window System software package, containing the OpenLook graphical user interface. OpenWindows replaces SunWindows, which is no longer available with Solaris. *See* Open Look, CDE.

operating system: A program that manages the resources of a computer by handling process scheduling, input and output procedures, interprocess communication, file management, and other housekeeping duties.

operator interface: Devices that link the computer and the operator including the screen, keyboard, and mouse.

operator overloading: In most programming languages, math and logical operators, such as +, –, *, /, and parentheses, only work on a restricted and well-defined set of operands. For example, math operators require integers or floating point numbers. In C++, it is possible to define what should happen when such operators are applied to unusual operands, such as user-defined variable types, complex variable types such as entire arrays and structures, or objects. *See* C++.

OSF (Open Systems Foundation): Organization founded to counteract the UNIX International (UI) initiative Both OSF and UI are competing in the creation of a standard to succeed System V UNIX. Members of OSF are IBM, DEC, HP, and other companies. OFS's X user interface is Motif. *See* Unix International, Motif.

OSF/Motif: *See* Motif.

overloading: *See* function overloading, operator overloading.

owner: Assigned to every UNIX file is an owner, normally the person that created the file. The name of the owner can be changed with special commands that can only be performed by the superuser.

packet: Large amounts of data are not transferred at once over Ethernet, but only in small, well-defined units called packets. If the transfer of any packet fails, perhaps because of a collision with another packet, Ethernet automatically sends it again.

paging: The loading and off-loading of individual pages of 4 or 8 KB between RAM and the swap space on a virtual memory system.

parallel computing: Standard workstations allow only pseudo-parallel execution of processes by extensive time-sharing on a single CPU. True parallel computing is available on some multiprocessor computer systems or through special software that uses free CPU time in a network, such as the Varian 3D NMR processing software. It seems likely that multiprocessor systems will lead the top-end workstation market in the near future. CPUs that operate at high clock rates (100 MHz and more) can be built into faster workstations, but it turns out the faster clock rates and technologies, such as ECL, require special components and circuitry throughout the CPU board, making a system too expensive. It is cheaper to use two or more CPUs on a single CPU board. This speeds up multitasking systems immediately, but not necessarily single processes, because special compilers and

software are required to make a single program use efficiently several processors in parallel. *See also* multiprocessor system.

parallel I/O: Simultaneous transmission of multiple bits over separate parallel lines. All bus structures used in a Sun—VME, SBus, MBus, UPA, SCSI and other—use parallel I/O. Sun workstations also have a Centronics parallel port for printers, *see* Centronics.

parent: The directory above the current working directory. In UNIX systems, the parent level is accessible under the name " . . " (two periods).

parity checking: Error checking, but not correction, implemented on most RAM memories for detecting 1-bit-per-byte errors. *Compare with* ECC memory.

partition: *See* disk partition, disk slice.

Pascal: High-level programming language that is excellent for teaching purposes because it educates programmers to write clear and logically structured programs easy to read for other people. Pascal can also be used for technical or scientific programming, but lacks flexibility compared to C and C++. There are Pascal compilers, Pascal interpreters, and P-code (semi-compiled Pascal) interpreters. *See* interpreter.

password: Secret keyword that is typed, without being displayed, after entering the user name to the login shell. Although passwords are optional, a secure password keeps data protected, because only the owner and the superuser have access to the data. Changing the password frequently is recommended. Longer passwords (up to 8 characters are significant) that contain numbers and uppercase and lowercase characters are more difficult for others to find out or guess.

PCI (peripheral component interconnect) bus: 32- or 64-bit standard bus structure used in most PCs. The PCI bus operates at 33 MHz or, more recently, at 66 MHz. Because many more PCI expansion boards are sold than SBus boards, PCI boards are less expensive. On the other hand, SBus cards are more compact. Starting with the Ultra 30, Sun switched from the SBus interface to systems with PCI bus. The PCI bus offers transfer rates of 50 MB per second (32-bit bus width, 33 MHz) up to 200 MB per second (64-bit bus width, 66 MHz). See Ultra 30, SBus.

PCMCIA (Personal Computer Memory Card Interface Association): A standard that defines the layout and the electrical interface of external memory extension cards. Developed for use in portable computers, PCMCIA cards are very small and are used for portable memory cards, modems, serial port interface cards, and even Ethernet ports. PCMCIA slots can be added to Sun workstations via a SBus card.

PDF (Portable Document Format): PostScript-based file format created by Adobe that is widely adopted for online documentation and for transmitting desktop publishing (DTP) documents. Files in PDF format typically have a "`.pdf`" extension and are viewed using Adobe Acrobat Reader software. VnmrJ and VNMR online manuals are delivered in PDF format. *See* Adobe, Acrobat Reader.

performance: Computer performance is widely discussed because computing equipment power serves as a type of status symbol in today's scientific and industrial communities (as much as MHz does for NMR spectroscopists!). More than that, performance figures, besides prices, are the most important sales argument in computer industry. It has to be recognized, however, that it is very difficult to establish a reliable measure for computer performance. In fact, it is difficult if not impossible to predict how fast a program will run on a specific computer, no matter how many performance figures are at hand—the performance depends greatly on the structure of a program, whether it fits into memory, whether the data it uses are structured, whether it requires considerable disk access, whether the disk access is sequential or randomized, and even small details such as how it uses the CPU registers and how much branching its object code contains. With multitasking

systems, it is also important how quickly and easily a system can switch between different programs and how it reacts to loaded situations. Predicting the performance is even more complicated on multiprocessor systems, because it also matters whether and how well a program uses parallel processing. *See* performance measures.

performance measures: Because of the large variety of computing requirements, there are many measures for computing performance. Some tests discussed in this manual are MIPS, a test for the speed of integer calculations compared to a VAX 11/780; MFlops, a test for floating point calculations; SPECint and SPECfp, a series of tests for both integer and floating point performance; and transactions per second, a test for multiuser and network server throughput, mainly for database applications. More tests exists to quantify the disk performance from maximum and sustained transfer rates, to network server performance using NFS and other protocols, and to specify the speed of computer graphics hardware from 2D vectors per second, 3D vectors per second, shaded polygons per second, and so forth. *See* MIPS, MFlops, SPEC, performance.

peripheral: Any hardware involved with input or output including monitors, terminals, and printers.

Perl: Interpreted programming language that is optimized for processing text strings and files. Perl is often the language of choice for CGI scripts, even though it is not bundled with most UNIX operating systems. *See* CGI.

permission: UNIX distinguishes owner, group, and all other UNIX users as levels of access to a file. For each level, UNIX has the read, writer, and execute permissions. Each of these nine permissions can be set individually by the owner or by the superuser.

PGX (PCI-based GX graphics): Fast single-chip 8-bit color graphics accelerator fast windowing and 2D/3D wireframe graphics. Derived from the GX and TGX graphics accelerators, the PGX graphics controller connects to the PCI bus and provides screen resolutions up to 1280 × 1024 or 1600 × 1000. It is standard in Ultra 5 and an option for higher-level Ultra workstations. *See* GX, TGX.

piggyback board: Small printed circuit board mounted on top of a larger board. Early piggyback boards were soldered onto the larger board, but piggyback boards (such as graphics controllers, SBus, MBus or UPA expansion cards) now usually plug into a connector on the main board.

pipe: UNIX facility that permits taking the output of a program and, instead of sending it to the display or other output device, feeds it directly into another program. The pipe is called by a vertical bar between the commands, for example, `dmesg | grep Ethernet`. The data do not show up on disk—the two programs are synchronized with each other and use buffers to hold the data. Multiple pipes can be used, and the amount of data transferred via pipe is not limited.

pixel: Single displayed point. On most Sun workstations, a pixel represents 1 bit of display information for black and white, or 8 bits for 256 colors. High-performance graphics accelerators use 24 bits for 16,777,216 colors, or even more bits per pixel, such as for hidden surfaces in 3D graphics.

pizza box: Popular name for the CPU module of desktop SPARCstation, which is about as size of an American pizza box but not as flat.

plotter: Device for putting continuous information, such as drawing lines and text characters, onto paper. VnmrJ and VNMR currently support a verity of plotters.

port: Connector for bus extensions or I/O devices.

portability: The ability to take a program, or software in general, from one computer to another computer and run it there. Normally, the software has to be recompiled for that purpose. Compilers in general are not portable.

PostScript: Standardized page description language from Adobe, used mostly with laser printers. PostScript allows the description of complex page layouts with simple commands. Lines, circles, and other vector-based objects can be described with a few PostScript commands instead of sending a dot matrix, often several million bytes, to a printer. PostScript also features output independent from the technical characteristics, including resolution and aspect ratio, of the output device. PostScript is also used for displaying graphics on the screen and sending complex documents over a communications link.

power failure: Interruption in the line voltage for more than about 0.1 second, stopping CPU execution, causing loss of RAM memory contents, and corrupting data files. The files most affected are those files partially in buffer memory that were just about to be written onto disk. After the interruption, the CPU does an automatic start-up, checks the hardware and disk file systems, and boots up UNIX, ending with a login prompt.

power switch: Turn on power switches on a Sun workstation in this order: screen, CPU, and hard disk (if a separate unit). Turn off power switches in this order: shut down the software properly, switch off the hard disk (if a separate unit), the CPU, and the screen. With the exception of the screen, which can be switched off overnight, avoid excessive switching on and off because it causes thermal stress in the computer hardware.

PPP (Point-to-Point Protocol): Protocol that allows building up IP (TCP/IP) connections over serial lines, modems, and ISDN connections. This not only offers the advantage of error checking and correction, as opposed to a link established by a simple modem dial-up connection, it also permits building up a temporary full Internet connection over serial lines and modems. A similar but somewhat less popular protocol is SLIP. *See* serial I/O, SLIP, TCP/IP, Internet.

precision: Length of a data word. Sun workstations are 32-bit or 64-bit computers, numeric information is stored in 16-bit, 32-bit, or 64-bit words: integers are normally stored in 32-bit data words (`int` or `long` in C), giving 32-bit precision, or in 16-bit "short" data words (`short` in C). Floating point numbers are stored in 32-bit (`float` in C) or 64-bit format (`double` in C).

printcap: UNIX file `/etc/printcap` containing special codes that define how all possible printers must be addressed, including baud rate, handshaking, and translations (e.g., into HP/GL, or data compression). Only used in BSD (`lpr`) printing. The file `printcap` still exists in Solaris 2.x, but is no longer really used.

printer: Device for putting discrete information—characters, dots, dot matrix graphics—onto paper. VNMR supports ink jet and laser printers.

process: Single execution of a program, involving machine-interpretable code and associated data. Every process runs in its own address space, separate from other processes. Several processes run simultaneously in a time-sharing system such as UNIX.

PROM (programmable ROM): *See also* ROM.

prompt: Sequence of one or more characters that indicate where text can be entered from the keyboard. Prompts can include information such as the host CPU name, user name, command line number, and more. Sun UNIX, VnmrJ, and VNMR software have a few conventions for the prompt: `#` is the prompt for single-user mode, *`hostname`*`#` is the prompt for `root`, *`hostname`*`,`*`username`*`,`*`command_line_number`*`>` is the prompt for a normal NMR user, and no prompt is present in the VnmrJ and VNMR command entry window. The default shell prompt is *`hostname`*`%` for the C shell (when starting a C shell with `csh -f`, without running `.cshrc`), and `$` for the Bourne shell.

protocol: Conventions for communication over interfaces and buses. Protocols for Ethernet include TCP/IP, NFS, and limNET. On Ethernet, several protocols can coexist on the same network, but only machines that use the same protocol can talk to each other.

QIC (quarter-inch cassette): The standard definition for 1/4-inch cassettes and tape drives. Individual standards include QIC-11 (4 tracks, 26 MB with 600-foot tapes), QIC-24 (9 tracks, 60 MB with 600-foot tapes), QIC-150 (24 tracks, 150 MB with 600-foot tapes), and QIC-2000 (2.5 GB), *see* DAT, Exabyte.

quit: *See* exit.

RAID (redundant arrays of inexpensive disks): RAID stands for a number of concepts (RAID-0 through RAID-5) for improving the performance and security aspects of disk subsystems. Both these aspects have gained importance over the recent years, especially in industrial environments, and some of the RAID concepts (RAID-0, RAID-1, RAID-5) have become available for Sun users through the Solstice DiskSuite software and disk arrays hardware. *see* RAID-0, RAID-1, RAID-3, RAID-4, RAID-5

RAID-0: Concept of disk striping: For sequential disk I/O (*not* for random disk I/O), the performance of the individual disk drive is the main bottleneck. Therefore, much higher throughput can be achieved if the data are spread over multiple disks that are then accessed *in parallel*. If this is combined with a fast/wide SCSI interface, transfer rates of up to 20 MB per second can be obtained. *See* RAID, RAID-1, RAID-3, RAID-4, RAID-5.

RAID-1: Concept of disk mirroring, While RAID-0 deals with the disk performance, RAID-1 deals with security. In fact, RAID-0 increases the system vulnerability, because the failure of any single drive would affect the integrity of the data on all the disks. With RAID-1, every disk is mirrored with a second disk, and data are always written twice. While this concept is rather expensive because it doubles the number of disks, it has the advantage that any single disk failure has no effect on the data integrity. Some hardware with RAID-1 support even allows unplugging and replacing disk drives while the system is running (so-called hot spares). Because RAID-1 increases disk I/O and lowers the performance, it is often combined with RAID-1. This is then called RAID-0/1 or RAID-0+1. *See* RAID, RAID-0, RAID-3, RAID-4, RAID-5.

RAID-3: Uses disk striping (RAID-0) and tries to increase its security by adding a disk containing parity information, typically one parity disk per 4 data disks. Therefore, RAID-3 adds security without the costs of full disk mirroring, as used in RAID-1. A problem with this setup is that the parity disk becomes the system bottleneck, because every data write operation also requires a write operation on the parity disk. Also, RAID-3 is implemented at a byte level. This has the disadvantage that with virtually every I/O every disk in the array is involved. This can be overcome with RAID-5. *See* RAID, RAID-0, RAID-1, RAID-4, RAID-5.

RAID-4: Almost identical to RAID-3, except that the implementation is done in blocks instead of bytes. This allows satisfying most I/O requests from a single disk, but it still suffers from the single parity disk as a bottleneck for write operations. *See* RAID, RAID-0, RAID-1, RAID-3, RAID-5.

RAID-5: Similar to RAID-3 and RAID-4, except that the parity information is spread over all disks of the array. This balances the load, combining the security of RAID-1 and the performance of RAID-0 without the costs of RAID-1. *See* RAID, RAID-0, RAID-1, RAID-3, RAID-4.

RAM (random-access memory): Memory that the CPU can read and write to. RAM is unlike ROM, which is also random access but read-only memory.

recursion: Calling the same program or a subroutine from within itself. Under UNIX, recursion also refers to programs that recursively cover a directory and all of the

subdirectories and subfiles below the directory. For example, the `rm` command with the `-r` recursive options, `rm –r /stuff`, removes the directory named `/stuff` and all subdirectories and subfiles below `/stuff`. *Be very careful when using the recursive option with UNIX commands*. `rm -r`, for example, can delete many files, including files you don't want to delete, in a fraction of a second.

refresh rate: Rate at which the display is refreshed. The higher the refresh rate, the less flickering is obtained. Sun screens use a 66- or 76-Hz refresh rate. *See* interlacing.

release: To be useful, software must constantly evolve, but not all evolution is straightforward. The implementation of new features and sometimes even repairing old bugs can create new problems. Adding in the need to keep software costs under control makes manufacturers collect software in packages and only release such packages when they are relatively consistent, reasonably tested, and an advancement over the previous software. New software packages are called releases or revisions, and they are normally labeled with a release or revision number and a date. Since most bugs are inherently linked to the software release in which they are found, it is absolutely essential to report the software revision number when bugs are reported.

remote: Actions performed on or invoked by other computers.

revision: *See* release.

RISC (reduced instruction set computer): Reduces the number of instructions at the machine level of a CPU to make the processor simpler and usually faster. Most graphics workstations use a RISC processor. *Compare with* CISC.

RJ45: Easy to plug and unplug connector type used in telephone networks and in 10baseT (10 Mbps) and 10/100baseT (10 and 100 Mbps) Ethernet connections over shielded twisted pair cabling. *See* 10baseT, 10/100baseT, Ethernet.

ROM (read-only memory): Hardware memory that generally cannot be altered by the computer and does not lose its contents when the power is switched off. Most computers use ROM memory to permanently maintain some low-level software such as boot-up routines and system diagnostics. Types of ROMs include PROMs, which are programmable ROMs; EPROMs, which are erasable programmable ROMs; and EEPROMs, which are electrically erasable programmable ROMs. *See* PROM, EPROM, EEPROM.

root: Name of the UNIX superuser, the system manager who has access to all files and processes on the system and does not require any permissions on the system. Also, root is the name for the origin of the file system that is accessible by the slash character (`/`). The root partition is the disk partition `rsd0a` that contains the origin of the UNIX file system as well as the most important UNIX administration data and files, such as the kernel.

root menu: Menu on most graphic screens that pops up when the right mouse button is pressed with the mouse pointing to the screen background. This menu allows calling commands without typing in a shell. Some commands available from the root menu include standard and scrollable windows, special tools such as a screen blanking program, an icon editor, a mail tool, and a program that terminates the windows environment. The root menu can be customized and may contain submenus.

root partition: *See* root.

RS-232, RS-423: Standard interfaces for serial communication. RS-232 technically is a subset of RS-423. RS-232 is specified up to 19,200 baud, RS-423 allows for higher transfer rates (49,500 baud and more). The RS-232 standard defines terminal and modem ports for transmitting and receiving devices, respectively. The assignment of the most important pins in a terminal connector is as follows ("in" means that on this line a terminal port is

listening): pin 1 is frame ground (cable shielding); pin 2 is TxD, transmit data (out); pin 3 is RxD, receive data (in); pin 4 is RTS, request to send: (out); pin 5 is CTS, clear to send (in); pin 6 is DSR, data set ready (in); pin 7 is GND, ground; pin 8 is DCD, data carrier detect (in); and pin 20 is DTR, data terminal ready (out). *See* null modem, XON/XOFF, handshaking.

safety: In connection with complex multiuser operating systems, safety includes the protection of files and programs against unintended loss, against data corruption by hardware or software problems, and against access by unauthorized users. UNIX has various ways of protecting files and data.

SBus: Internal bus in desktop SPARCstation and Ultra workstations that uses 32-bit parallel, synchronous data transmission at a speed of over 50 MB per second. The SBus can be accessed though connectors on the CPU board, but one or two of these connectors are used by the graphics controller on some systems. The 64-bit version of the SBus allows for peak transmission rates of 160 to 200 MB per second.

scheduler: Program or operating system utility that assigns CPU time segments from fractions of a second up to a few seconds to the various programs running simultaneously in time-sharing operation in a multitasking system. The scheduler respects program priorities but also tries to avoid CPU blocking by excessively long processes. Except for programs with very high priority, programs are interrupted every now and then, typically once in a second, to enable small processes to get their share of CPU time.

screen: CRT display.

screen burn-in: The luminosity of a CRT display decreases in places where a bright picture is left on for a long time. Activate the screen lock from the root menu or the screen blanking mode to make the screen black, or simply switch the monitor off overnight, over the weekend, and during extended idle periods.

screen blanking: Sun Solaris CDE software automatically turns off the screen after a period of disuse. The screen then turns back on when the mouse is moved or when the keyboard is touched. Password protection is provided.

screen locking: A Sun Solaris CDE software locks the screen when it is blank. To unlock the system requires a password.

scroll bar: If the displayed information in a window exceeds the size of the window, a program such as `more` is required to scan though the information page by page or line by line or a special window is required that can keep all the information in a buffer. Such windows allow scrolling forward and backward through the information (`more` only allows forward scrolling, unless called on a file). The tool that allows scrolling with the mouse is called the scroll bar, a bar along one edge of the window that symbolizes the size of the displayed information and an indicator where within the entire buffer the current displayed information is located. On CDE, scrolling is achieved by moving the mouse onto the scroll bar and dragging the elevator symbol in the scroll bar up and down.

SCSI (Small Computer Systems Interface): Interface for high performance hard disks. EIDE is also a common interface for hard disks.

sector: Part of a circular track on a single disk surface. The number of sectors per track is in the order of 15 to 40. *See* cylinder, interleave factor.

seek time: Time required to access a track on a disk, a function of the distance between the last and the current track. Current hard disks have average seek times of 8 to 15 ms. By working in cylinder groups, Solaris can reduce seek times when reading single files, because such files are not scattered over the entire disk.

serial I/O: Data input and output, bit by bit, via a single line. Most serial interfaces use one line for sending and one for receiving. More lines may be used for grounding, handshaking, and power. Most printer interfaces use 3 to 5 lines, some terminals require 8 lines, and even more are sometimes required for synchronous transmission, *see* RS-232.

server: Computer that provides facilities, mostly disk space, for other computer systems called clients. For the X Window System (X11), the terms client and server have a different meaning. *See* client, X Window System, X client, X server.

SGML (Standard Generalized Markup Language): In the late 1960s, IBM created a Generalized Markup Language (GML) to overcome difficulties in transporting documents between different computers and operating systems. In 1986, GML became ISO 8879 and renamed the Standard Generalized Markup Language. Generalized means that it does not specify how exactly a document is to be presented or formatted, but it instead defines the document type and presentation styles. An SGML document consists of three parts: the character set and which characters are used to differentiate markup tags from standard text; the document type and which markup tags are accepted in the following document, and the document instance that contains the actual text with markup tags. The three parts don't need to be in the same document or file. HTML is an *instance* of SGML. A HTML document corresponds to the third part of an SGML document, and its markup tags are defined as the hypertext markup language. *See* HTML.

shell: Generally means a platform on which other programs are run under UNIX. Called a shell because when the UNIX functionality is usually visualized, the shells are grouped around the kernel, which is shown as a ball with a small machine-specific core. Shell also stands for a UNIX command interpreter, such as the C shell, Korn shell, or Bourne shell.

shell script: Text file with commands and control statements that are interpreted by a shell. Many UNIX commands are actually powerful shell scripts. Solaris offers different shells, including the C shell, Korn shell and Bourne shell; therefore, it allows for different kinds of shell scripts, using these shell languages. *See* shell.

shielded cable: Cable with grounded electrical shielding against rf pickup from external sources. Such shielding is woven from metal wires or made of metal foil. The shielding can surround many conductors or each conductor can be shielded individually. To avoid ground loops, the shielding should only be grounded on one side of the cable. *See* coaxial cable, ground loop.

shielding: Current computers operate in the range of common rf. Therefore, computers can generate rf emission that can potentially interfere with radio and television equipment, as well as analytical equipment including NMR machines that operate within this frequency range. For this reason, there are restrictions on the amount of rf emission from computers. Sun workstations are in fact heavily shielded (the shielding metal case at the same time ensures mechanical stability). At the same time, computers should be protected against rf pickup from external sources. Cables especially can act as antennas and should be shielded. *See* shielded cable.

SIMM (single in-line memory module): Small boards holding RAM chips that can be mounted easily onto the CPU board without soldering. A SIMM currently can contain up to 64 MB of dynamic RAM, usually with parity checking. High-end workstations, such as the SPARCstation 10 and 20, and Ultras, use SIMMs with ECC. *See* RAM, ECC memory.

single-user mode: Low-level mode of UNIX operation that does not allow for multitasking or daemons. Note that this means no printing is possible. This mode is mostly used for system administration tasks such as checking and repairing file systems, and making tape dumps from disks. *See* multitasking, daemon, multiuser system.

SLIP (Serial Line Internet Protocol): Protocol that allows building up IP (TCP/IP) connections over serial lines and modems. This not only offers the advantage of error checking and correction, as opposed to a link established by a simple modem dial-up connection, it also permits building up a temporary Internet connection over serial lines and modems. A similar protocol is PPP. *See* serial I/O, PPP, TCP/IP, Internet.

slice: *See* disk slice.

SMD (surface mounted device): On conventional printed-circuit boards, chips are mounted by soldering each leg into a dedicated hole in the board with a distance of about 2.5 mm between connections. With the SMD technique, the chip legs are soldered flat onto the surface of a printed circuit board with about two connections per mm. This allows keeping the dimensions of VLSI chips, with their large number of connections, as well as the board itself, small while maintaining high reliability. SMD chips can only be mounted by machines, and it is virtually impossible to unsolder chips. CPU boards in Sun workstations use this technique extensively.

software: Any kind of binary information used in a computer, whether a compiled, program. source code, data, or text files. Software might reside on tape, on disks, in RAM or ROM memory, or on any other temporary or permanent storage device.

software maintenance: Includes file system maintenance and software upgrading. Maintenance includes avoiding overloaded file systems by off-loading data to tape or other file systems or computers, deleting unnecessary files, and checking file systems for consistency. A software maintenance contract guarantees that the software is kept up to date by always providing the latest release.

Solaris: Solaris is a trademark for operating system software from SunSoft, a subsidiary of Sun Microsystems.

Solstice: Sun's trademark for a large suite of software products for network administration, system management, telecommunication, PC networking and Internet connectivity.

SPEC (System Performance Evaluation Cooperative): An independent organization that created a test suite for standardized CPU performance tests. SPEC performance figures are now widely accepted as more reliable and comprehensive than MIPS and MFlops numbers. The main objective of the SPEC initiative is to provide a series of tests that cannot be tuned by the computer manufacturers. With the increased number of CPU registers and cache memory, the conventional MIPS and Linpack (MFlops) performance tests produced unrealistically high numbers that had little practical meaning for the performance of everyday programs. Even the SPEC performance test suite is periodically revised to avoid tuning of compilers and computer hardware to the tests. The first test suite was introduced in 1989 as an attempt to reduce the computer performance to a single number, the geometric mean of the execution time of four integer and six floating point intensive programs, compared to the performance of a VAX 11/780. This idea received only limited acceptance, because many users wanted a test of more general UNIX performance involving integer operations, as opposed to the number crunching performance on floating point operations. For this reason, the 1992 test suite consisted of two different figures: SPECint92 was made the geometric mean of the execution time 6 integer performance tests, compared to the performance of a VAX 11/780, and SPECfp92 was made the geometric mean of 14 floating-point intensive performance tests. In 1995, the test suite was further enhanced. The basic tests are partly the same (8 integer performance tests, 10 floating point performance tests), but the test conditions are defined more vigorously, and the new figures were defined, SPECint95 and SPECfp95, based on the performance of a SPARCstation 10/41. For multiprocessor systems, a special set of

throughput tests (see SPECrate) was created. *See also* MIPS, MFlops, SPECmark, SPECint92, SPECint95, SPECfp92, SPECfp95.

SPECfp92: Performance measure for floating-point-intensive programs, typical for scientific number crunching. SPECfp92 is the geometric mean of execution times on 14 performance tests, relative to the execution times on a VAX 11/780. *See also* SPEC, SPECint92.

SPECfp95: Performance measure for floating-point-intensive programs, typical for scientific number crunching. SPECfp95 is the geometric mean of the execution times on 10 performance tests, relative to the execution times on a SPARCstation 10/41. *See also* SPEC, SPECint95.

SPECint92: Performance measure for programs dominated by integer calculations, such as typical UNIX commands. SPECint92 is the geometric mean of the execution times on 6 performance tests, relative to the execution times on a VAX 11/780. *See also* SPEC, SPECfp92.

SPECint95: Performance measure for programs dominated by integer calculations, such as typical UNIX commands. SPECint95 is the geometric mean of the execution times on 8 performance tests, relative to the execution times on a SPARCstation 10/41. *See also* SPEC, SPECfp95.

SPECmark: First performance measure created by the SPEC initiative in 1989. The measure consists of 10 tests, including 4 integer and 6 floating-point-intensive programs. The SPECmark performance number is the geometric mean of the execution times for all these tests, relative to the performance of a VAX 11/780. Although the SPECmark may be a more objective measure for general computing performance, it doesn't reflect the fact that, for example, some programs consist of only integer operations while others are dominated by floating-point calculations. Computer performance can't be reduced to a single number. It depends heavily on the type of operations involved, whether integer or floating point, or single or double precision; on the structure of the data, whether scalar or vectorized; the amount of disk operations required; and the number of competing processes. To gain more information on integer as compared to floating-point performance, the more recent performance tests (SPECint92, SPECfp92, SPECint95, SPECfp95) have been separated into integer performance, typical for UNIX commands, and floating point performance, typical for scientific number crunching.

SPECrate: Test suite defined by SPEC consortium, designed to measure performance of multiprocessor and single-processor systems. Similar to SPECmarks, SPECrate has separate integer and floating point test suites. *See* SPECmark, MIPS, MFlops.

SRAM (static RAM): Fast but expensive RAM chips, with access times of 60 nsec and less. Due to their higher complexity per storage element, these chips can currently only be made in smaller sizes like 256 kilobits or 1 megabit per chip. On high-end workstations, SRAM chips are used in the cache memory. SRAM does not require refreshing, in contrast to DRAM (dynamic RAM), which can only hold information when it is periodically refreshed. *See* cache memory, DRAM.

stack register: A last-in first-out (LIFO) buffer. When a stack gets full, the items entered first are deleted.

stand-alone workstation: Complete, functional workstation that does not require connections to other computers such as servers or clients.

static discharge: Usually generated by friction when walking around or moving on a chair. Dangerous to any electronic equipment, but can be avoided by humidifying the air and using conducting floors, carpets, and shoes.

STP (shielded twisted pair): Cabling similar to telephone cables, used for 10baseT (standard) and 100baseT (fast) Ethernet, and for ATM for transfer rates up to 155 Mbps. In NMR environments, STP cabling is preferred over UTP (unshielded twisted pair) cabling, because STP avoids rf emissions. *See also* Ethernet, ATM, UTP.

structured programming: A programming style designed to create programs easy to read and understand by persons other than the programmer. Ideally, structured programming is combined with a top-down approach. Overall, both improve programming efficiency. Certain programming languages such as Pascal or C almost automatically lead to structured programs. Low-level programming, such as assembly language, are virtually impossible to structure. *See* top-down programming.

Sun Microsystems: Founded in 1982, Sun is the leader in the workstation market. The keys to its success were SunOS and its integrated networking software, the NFS standard that Sun created, and the SPARC CPU chip technology. In 1991, Sun Microsystems Inc. was restructured by forming a number of subsidiaries such as SunSoft and Sun Microsystems Computer Corporation (SMCC). *See* SPARC, NFS, SMCC, SunSoft.

SunSoft: Subsidiary of Sun Microsystems, Inc., founded in 1991, specializing in the development and the marketing of Solaris operating system, which they have ported to the PC-compatible platform. SunSoft also develops Sun's compiler technology and creates new software for the Solaris environment. *See* Sun Microsystems, Solaris 1.0, Solaris 2.

Suntools: Earlier name for SunView. *See* SunView.

Supercache: 1-MB cache memory on some SPARCstation 10 and SPARCstation 20 models (10/41, 10/51, 20/51, 20/61, 20/71). *See* cache memory.

supermini: Former, "extra powerful" minicomputers, such as VAX 11/7xx and VAX 8xxx, used in scientific environments.

superuser: Special user who is allowed on a system to read any file from any owner, to write into any file, and to execute any executable file. With UNIX, the superuser normally has the username `root`. *See* permission, root.

surface analysis: Procedure usually performed after formatting a disk for detecting bad tracks.

SVR4 (System V, release 4): *See* System V.

swapping: Different from paging, swapping is the transfer of entire processes from RAM into the swap space. To free up memory, UNIX decides to swap out processes that have been idle for a certain time, usually a matter of seconds or minutes, or that have a low priority. When such a process is reactivated, perhaps by clicking on a window that has not been used for a while, there should only be a slight delay of a second or so until the process is ready again. The delay might be because another process has to be swapped out in turn, but mainly it is because the data have to be read into RAM again. Swapping and paging can be avoided by adding more RAM. *See* swap space, paging.

swap space: UNIX uses a special disk partition as extension of the RAM memory. The ordinary UNIX file systems are complex and, therefore, not optimized for speed of disk transfers. The swap space, on the other hand, is organized differently, so that every time the active programs request more memory than currently available, UNIX quickly moves inactive processes to the swap space on disk. In addition, if a single process requires more memory than the amount of memory physically installed, UNIX can use virtual memory capabilities to load and off-load data in pages of 8 KB or 4 KB from and to the swap space. *See* paging.

switch: *See* power switch.

symbolic link: An alias or alternate name for a file in the UNIX file system. A symbolic link is a type of pointer to a file in another directory, which can help simplify paths. For example, `/export/home/vnmr` is accessible as `/vnmr` through a symbolic link `vnmr` in the `/` directory, pointing to `/usr/vnmr`. Unlike hard links, symbolic links can be made across partition limits. When such a file is deleted, the symbolic links are not removed. On the other hand, when a file is removed is a symbolic link, only the link is removed, not the file itself. *See* hard link.

synchronous: Transmission of data over dedicated lines with synchronization signals that, in essence, determine the speed of communication. Synchronous transmission is more demanding of the hardware than asynchronous I/O, but is potentially faster, with up to 64,000 baud on the RS-232 interface. In the RS-232 standard, the lines 15, 17 and 24 for synchronization are used in addition to some of the normal handshaking lines 4, 5, 6, 8, and 20. The SBus in desktop SPARCstations uses synchronous data transfers for optimum reliability and speed. *See* asynchronous, handshaking.

System V: The current version of UNIX from the AT&T Bell Labs. System V UNIX was developed in 1984 from System III, which originated indirectly from UNIX Version 7, a version written for the PDP/11 computer. SunOS 4.x combines 4.2BSD UNIX with the most important features of release 3 of System V. System V.3 is more popular, but has some specific disadvantages over 4.2BSD, such as a file system organization that is slower and suffers from aging, getting slower and slower with the time of use. Periodic cleanup operations are necessary to maintain acceptable performance. Together with AT&T, Sun has developed System V, release 4 (often abbreviated as SVR4). This is now marketed as Solaris 2 and has become the standard Sun operating system. SVR4 combines the advantages of 4.2BSD UNIX with those of System V.4. Sun-3 and earlier systems are not compatible with Solaris 2 and can therefore not be upgraded to this level of operating system. *See* UNIX system, SunOS, Solaris 2.

T-switch: Manual switch that allows using two I/O devices alternatively, but not simultaneously. If the two devices talk a different language, such as a plotter and a printer, the user must be careful to completely terminate the communication with the device that has to be disconnected. A terminal and an output device, such as a printer, must not be connected to the same port via a T-switch, because the handshaking signals of the output device would cause the port to initiate a login procedure, which would then interfere with any attempt to send data to the plotter.

tape: Magnetic medium to store digital information, mostly for backup purposes or for transporting large amounts of data. Current Sun workstations support 1/4-inch tapes (150 MB and 2.5 GB), DAT tapes (5 GB), and 8 mm Exabyte tape drives (2.5, 5, 10, and 14 GB). Early versions of the Sun operating system were delivered on QIC-24 tapes. All Sun-3 systems were equipped with a QIC-24 (60 MB) drive. Many SPARCstations are equipped with the QIC-150 tape drive, which can also read QIC-24 and QIC-11 data. Software from Sun and Varian is now all delivered on CD-ROM. This means that tape drives are used mostly today for backup and mass storage purposes. *See* QIC, Exabyte, DAT.

tape archive: *See* tar.

tape cassette: Cassette with 1/4-inch tape. Recommended quality is 3M DC600A (600 feet), or equivalent, for maximum (60 MB) storage capacity. For SPARCstations with QIC drive, 3M DC6150 (600 feet, 150 MB), or equivalent, must be used. Other tape formats also come in cassettes: 8-mm 5-GB Exabyte tapes and DAT cartridges. *See* QIC, Exabyte, DAT.

tape drive: Device for reading and writing magnetic tapes. Most Sun systems are equipped with one or two tape drives. The drives come in various formats and can also be used remotely. *See* tapes, QIC, Exabyte, DAT.

tap kit: Device that connects a transceiver to an Ethernet cable. Three different kinds of tap kits are in use: thin Ethernet tap kits that connect via BNC-type T-piece, N-type tap kits that connect to thick Ethernet cable via N-type connectors, and vampire tap kits that connect onto standard thick Ethernet cable without disrupting the network. Mounting a vampire tap kit onto Ethernet coaxial cable requires special tools and careful operation because the cable can be inadvertently shorted. With the newer twisted-pair Ethernet, standard tap kits are no longer required. *See* Ethernet, twisted pair.

tar: UNIX tape archive program for reading and writing multiple files stored on a magnetic tape. Useful also for transferring data between different file systems.

task: Single execution of a program. Under UNIX, the term "process" is normally used instead of task, although both terms mean the same thing.

Tcl/Tk (Tool command language /Tool kit): Tcl/Tk (pronounced "tickle tee-kay") is an application programming interface (API) used in the development of interactive applications or graphical user interface (GUI) under UNIX. Tcl is a simple interpreted scripting language, similar to a UNIX shell or a macro language. Tk is a toolkit that provides facilities, such as buttons, menus, and event handlers, for creating an interactive GUI.

TCP (transport control protocol): Networking transport protocol that uses the IP network protocol for the transfer of information over Internet. TCP provides error-free, in-order, and two-way data IP packet transfer. TCP is used by higher-level services, such as `rcp`, `rlogin`, `rsh`, and `ftp`. TCP is often named in connection with IP (TCP/IP), even though the two protocols are two totally different entities and IP is used by other services, such as UDP. *See* IP, TCP/IP, Internet, UDP.

TCP/IP (**transport control protocol/internet protocol)**: Networking protocols for communication and data transfer via Ethernet. Available on all Sun systems, most UNIX systems, and many PC compatibles and Apple Macintoshes. *See* Ethernet, NFS.

terabyte (TB): 1,099,511,627,776 bytes, which corresponds to 1024 gigabytes (GB).

terminal: Operator interface consisting of at least a screen and a keyboard.

terminator: Device used on buses and open rf cables to avoid signal reflections and erroneous signal transmission. *See* bus terminator, Ethernet terminator.

text file: File that contains ASCII characters only and can be accessed by text editors, such as `vi` and `textedit`, as well as many UNIX commands, including `cat`, `more`, and `nroff`. Each character in a text file occupies 1 byte in memory or on the disk. VNMR was written to work with text files as much as possible, because such files can be accessed, modified, and inspected easily, as opposed to binary files, which are normally only accessed by special commands or dedicated programs. *See* binary file, directory file.

text processing: Off-line text formatting program, as opposed to word processors. With text processors, the formatting instructions are inserted into the text in the form of special character sequences and whenever the formatted text is requested, the text processing scans through the entire text and interprets the formatting instructions, producing the formatted document as output on a suitable printer or on the screen. Standard UNIX software includes the text formatting programs `nroff` and `troff`.

textedit: An easy-to-use mouse- and window-oriented UNIX text editor. Textedit is always in the insert mode and has features such as drag-and-drop and window splitting, which allow working on various portions of the text simultaneously. Experienced UNIX users probably prefer the `vi` text editor, which has a less sophisticated user interface but is faster and more powerful, especially for repetitive tasks. *See* vi, text file.

thin Ethernet: Ethernet based on 50-ohm BNC cable connections. The cables must have terminators on both ends and no branching is allowed. *See* Ethernet.

TIFF (**tagged image file format):** Popular format for storing graphics. First created by Aldus Corp., it was later adopted by most PC and Macintosh painting, imaging, and desktop publishing programs. TIFF can be used to store both grayscale and color graphics. On PCs, TIFF files usually have a .tif extension. *See* GIF, JPEG.

time sharing: An operating system that shares CPU time between several tasks, or processes, running simultaneously. This involves priority handling and interrupting long processes as frequently as every second to let other processes have their share of CPU time. All this is completely transparent to the user, and if only one task is mainly active, there is be no significant slow-down in performance as compared to a system without time sharing. *See* process, task, multitasking system.

tool: Program that can be used to build complex programs. Most UNIX programs can be regarded as tools for building powerful shell scripts, while MAGICAL commands are tools for building macros. A shell tool within OpenLook is just a window with a command interpreter for performing any UNIX task.

top-down programming: An approach in programming that describes the sequence or the timing in which the various parts of a program are written. First, a very general main program is written, for example:

```
#include <stdio.h>
main()
{
   getdata();
   calculate();
   showresult();
}
```

Then, the subroutines, such as `getdata()`, `calculate()`, and `showresult()` in the example, are defined, details are filled in for defining further subroutines, and so on, down to the smallest details. *See* structured programming.

transceiver: On Ethernet networks, a combined transmitter and receiver that converts serial binary information into modulated 10 MHz rf and vice versa. The transceiver is directly hooked up to the Ethernet cable via a tap kit. Branching from the Ethernet cable to the transceiver is not allowed. The transceiver is not a terminator.

transfer rate: Speed at which data are transferred continuously over an interface, or to and from a hard disk. SCSI disks transfer data at 2 to 4 MB per second. The SCSI-2 bus in the SPARCstation 10 and 20 operates at up to 10 MB per second, the fast/wide SCSI in Sun Ultra systems with Creator graphics supports transfer rates of up to 20 MB per second. The transfer rate depends on a number of factors: bus speed, average seek time to access tracks, latency and interleaving, and amount of fragmentation in the file or in free space when writing a file. To minimize track-to-track access time and increase file transfer rates, the fast BSD 4.2 file system used in Sun workstations tries to keep files in subsequent tracks, or cylinder groups. *See* access time, latency, interleave factor, cylinder group.

tree structure: Hierarchic structure of a directory file with subdirectories, subdirectories of subdirectories, and so forth, and their subfiles. To avoids loops, a proper tree structure does not allow for two links to the same subfile. *See* directory file, link

troubleshooting: Activities such as rebooting the software, checking electrical connections, and consulting manuals or course notes. If nothing helps, call service for hardware problems, an applications chemist for software problems, or your salesperson and your boss for monetary problems, if additional hardware is required.

twisted-pair Ethernet: Ethernet standard that allows Ethernet communication over telephone cable-like hardware (unshielded twisted pair, UTP). Newer SPARCstation and Ultra workstations come with a twisted-pair Ethernet connector, besides a standard Ethernet (transceiver cable) connector. Unlike traditional Ethernet, which always requires a linear backbone without branching, twisted-pair Ethernet also allows for star-shaped network topologies. *See also* UTP.

UC Berkeley: *See* 4.2BSD.

UDP (user datagram protocol): A simple data transport protocol that builds on the IP network layer and protocol. UDP is faster but less robust than TCP and is used for services such as HTTP, where information is transported primarily for visual presentation and data integrity and error-checking are less important than with other services, such as `ftp` and `rcp`. *See also* IP, TCP, Internet, HTTP.

UI: *See* Unix International.

UNIX International (UI): Non-profit organization originally founded by Sun Microsystems and AT&T to create and promote UNIX System V, release 4 (SVR4) and associated software. Over 150 other companies have joined Sun and AT&T in UI. Fearing a monopolization of the UNIX market, a number of other companies created the Open Systems Foundation (OSF), which also created a UNIX standard. *See* System V.

UNIX system: Operating system invented by Ken Thompson and Dennis Ritchie at AT&T Bell Laboratories for the PDP-7 minicomputer. Initially written in assembly language, Version 3 UNIX was translated into C in 1973. Distribution to the public outside Bell Labs started in 1974 with Versions 5 and 6. Version 7 in 1979 was the basis for all of the following UNIX implementations. Since then, UNIX has split up into a family of operating systems such as XENIX and HP-UX. AT&T developed System III in 1982 and System V in 1984. System V is probably the most popular version today. In 1980, the U.S. Department of Defense chartered the University of California in Berkeley to redesign UNIX for distributed computing on networks. The results of this cooperation were 4.1BSD and 4.2BSD (Fourth Berkeley Software Distribution), which featured a faster and better file system, network support, largely expanded process address space, and new utility programs such as a fullscreen editor (`vi`) and an alternate command interpreter (C shell). 4.2BSD became the starting point for SunOS. *See* System V.

URL (uniform resource locator): Hypertext address for a remote resource in HTML. The principal format of a URL is as follows:

protocol`://`*user*`:`*password*`@`*host*`.`*domain*`:`*port*`/`*path*`/`*file*`#`*anchor*

where *protocol* is most often `http` (for HTTP) or `ftp` (for FTP), *user* and *password* are the user name and password for non-anonymous FTP, *host*`.`*domain* is an Internet host address (such as `www.nmr.varian.com`), *port* specifies a non-standard port in cases where such port addresses are used (e.g., where a server has both a public and a private port or interface for the same protocol), *path* is the directory containing the resource (if *path* is not specified, the user to led into the default directory), *file* is the local file path to the resource (if *file* is not specified, the user is shown a directory listing, if FTP, or a document named `index.html` or `index.htm` is transferred, if HTTP), and *anchor* (HTTP only) is a tag or label within an HTML document that the user jumps to when the document is displayed. Some examples:

```
http://www.varian.com
http://www.nmr.varian.com/products/software/
http://www.nmr.varian.com/news/events.html
ftp://prep.ai.mit.edu/u2/emacs/
ftp://usergroup@vnmrnews.nmr.varian.com/pub/userlib/README.FIRST
```

See also HTML, WWW.

user: Person that has direct access to the UNIX software. For each new user, the superuser `root` installs the necessary files in the `/home` directory and starts an account for the user. For safety reasons, `root` should assign an initial password for new users. Users other than `root` have only limited write access to files outside their home directory. Each user has a home directory in `/home`. *See* superuser, root.

user partition: General-purpose partition for users, typically with home directories. On the Sun host, the `/usr` directory normally contains only software and global data files for UNIX. The home directories and VNMR are stored in the `/export/home` directory on a separate disk partition of the main disk.

utility: UNIX programs—including editors, command interpreters, and compilers—that run on top of the kernel. Basically, any UNIX program that is not a kernel function can be called a utility.

UTP (unshielded twisted pair): Telephone cable-like cabling used for 10baseT (standard) and 100baseT (fast) Ethernet, and for ATM. Transfer rates can be as high as 155 Mbps. *See also* Ethernet, ATM, STP.

vi: Powerful and frequently used fullscreen UNIX text editor. `vi` (pronounced "vee-eye") is compatible not only with the Sun graphics interface, but also with RS-232 terminals, because it does not rely on the mouse. The cursor is moved around with cursor keys or with special commands instead. `vi` is powerful, but definitely more complex than `textedit`, the other major UNIX text editor. `vi` operates in three modes: insert mode, `vi` command mode, and last line or `ex` command mode (`ex` is a line-oriented text editor often used for global changes). As the standard text editor on the Sun host, `vi` is called by many VNMR commands. *See* textedit, text file.

virtual memory: By dividing every process into pages of 8 KB or 4 KB, individual pages can be loading and unloading instead of processes. This allows working with a processes up to 1 GB, much more than the actual RAM memory. *See* process.

virus: Generally a small, but sometimes quite dangerous, software bug that can multiply within the software, infecting files—especially executables. A virus can destroy an entire operating system, but fortunately, viruses are mainly an issue in the area of PCs and Macintoshes, and UNIX has not so far suffered from viruses.

VIS (Visual Instruction Set): The UltraSPARC instruction set has been complemented by a special visual instruction set (VIS) that dramatically accelerates graphics and imaging. Pixel manipulations that otherwise would take many steps can be executed in a single clock cycle. VIS also accelerates image processing, real-time video compression, such as JPEG and MPEG, and image animation. *See also* UltraSPARC, MPEG, JPEG.

VME: Industry standard bus structure. Implemented as part of the backplane, VME is the main expansion bus on Sun 3/xxx and 4/xxx computers. As used in Sun computers, the VME bus is 32 bits wide and allows continuous data transfer rates of up to 8 MB per second between different computer boards. The VME bus is also used in the current Varian NMR acquisition computers.

VnmrJ: Varian Inc. research NMR software package.

VRAM: Fast random-access memory (RAM) chips that are optimized for use as the hardware frame buffer in video controllers: VRAM chips are dual-ported with one port used for updating and changing the display, and the other port is used to read out the display information. *See* RAM, 3D-RAM.

VRML (Virtual Reality Modeling Language): An extension to HTML, introduced by SGI, that enables defining and manipulating 3D objects within hypertext documents across the Web. *See* HTML, SGI, WWW.

W3: *See* WWW.

wait states: CPU cycles that are spent waiting for data to arrive in the CPU registers from the RAM memory. Because RAM memory is considerably slower than the CPU clock rate, often the CPU is idling for a few cycles. Modern CPU architectures try to avoid such wait cycles by using an intermediate, fast-cache memory between the RAM, the hard disk, and the CPU. *See* cache memory.

WAN (wide-area network): A large-scale computer network across several cities or countries

Web: *See* WWW.

Web authoring: The process of creating and designing HTML documents. *See* hypertext, HTML, WWW.

Web authoring tool: Software for creating and designing HTML documents. *See* hypertext, HTML, WWW.

Web browser: Program that requests, receives, formats and displays hypertext (HTML) documents from a Web server, using the HTTP protocol, allows for user interaction with HTML documents, and provides a convenient access to the FTP protocol for downloading text and binary data. The first browsers were pure ASCII text interfaces, but graphical browsers such as Netscape Navigator, HotJava, and Mosaic are most popular on UNIX systems now. These browsers offer true multimedia access to the Web, with on-screen animation and audio support. *See* Mosaic, Netscape, HotJava, Java.

Web page: A HTML document in general or the entry point into a series of HTML documents on a Web server. A large number of companies have now opened up Web pages and do business over the Web, and many Web users have defined personal Web pages. *See also* HTML, WWW.

Web server: Networked machine that provides access to hypertext (HTML) documents via HTTP protocol. In the case of a UNIX system, this is achieved with the HTTP daemon, `httpd`. Web server software is available from companies that also market Web browsers. See also HTML, HTTP, Hypertext, WWW.

Winchester disk: General term (not a trademark) for a sealed hard disk, mostly with several internal disk platters. *See* hard disk.

windows: Dividing the screen into separate areas for different tasks. With windows, the screen is no longer reserved for a single task, such as a command interpreter or a text editor or a graphics program. Each window is clearly limited by a frame and usually has a title bar that indicates the kind of task running in the window. Windows can overlap and can even be completely covered by other windows. It is also possible to close a window but not destroy it. The window is then shown as a small icon. Invariably, windows also imply that some kind of graphics pointer must be used to indicate to the software which window the operator currently wants to work in or what size and location the operator desires for a window. The most popular graphics pointer is the mouse. Because UNIX is a full multitasking and timesharing operating system, all windows shown on the screen are continuously active, even if they are hidden behind other windows or if they are shown as icons. Only one window can have the focus for keyboard entries. This implies that the operator has to move the mouse onto that window and click on it with a mouse button (in VNMR, the left mouse button is used). The keyboard focus is then indicated by a solid

frame around the window. In OpenLook (as in X in general), windows are also allowed to move outside the display area. With SunView, this was not possible. *See* icon.

word: The number of bits handled by the computer at a time: 16, 32, or more bits.

word processing: Text formatting software, such as Microsoft Word, that displays the formatting online. *See* text processing, WYSIWYG.

working directory: *See* home directory.

workstation: Complete stand-alone computer, usually with a single operator interface.

WORM drive: "Write once, read many" optical disk: optical disks that can be recorded once, but not rewritten. WORM drives are used for archiving purposes in some (industrial) GLP environments, as WORM disks are certified to hold the information for over 10 years, and the information on WORM disks cannot be altered. Even though WORM disks are very reliable storage media with a capacity of up to 1 GByte or more per side, no format standard for WORM disks was ever defined (which lead to a diversity of incompatible formats), and no major manufacturer built such drives. The DVD-R may once replace the WORM drives. *See* CD-ROM, DVD, DVD-R.

WWW (World Wide Web, W3, Web): Widely used terms referring specifically to the wealth of interconnected systems communicating using protocols, mainly HTTP, the hypertext transfer protocol. Basically, the Web is a world-wide collection of hypertext (HTML) documents with references (hypertext links) to each other, forming a single, cross-linked pool of information, a world-wide information resource. The idea of using hypertext to link documents on different computers, or in general in different parts of the world, was first brought up in 1989 by a group of researchers led by Tim Berners-Lee at the CERN (Centre Européen pour la Recherche Nucléaire) in Geneva. This defined the origin of the World Wide Web. The most convenient way of accessing documents on the Web is via Web browser. With the advent of the first graphical browser, Mosaic, in 1993, the Web rapidly gained popularity. The term "World Wide Web" is sometimes incorrectly taken as synonymous to Internet. *See* HTTP, HTML, Web browser, Web server, Internet.

WYSIWYG (what you see is what you get): Word processing software, such as FrameMaker, that formats the text online so that as the operator types, the text and appearance of the document are shown on the screen as they will look like when the document is printed. WYSIWYG software gives the tightest and fastest possible control over the final text format. The disadvantage is that such software requires a faster CPU to display pages at the same speed as a text editor. *See* word processing.

X: *See* X Window System.

X client: Program that delivers X Window System display information for an X server. The X client can run locally or on a remote computer, in which case the display information is transferred via Ethernet. Note that the X terminology for servers and clients is totally different from the rest of the UNIX world. *See* X Window System, X server.

XON/XOFF: Special characters that can be transmitted back over RS-232 lines by the receiving device, to start and stop the data flow. If XON/XOFF is the only handshaking option selected, a three-wire cable using pins 2, 3 and 7 is sufficient. Optionally, a shielded cable with the shield connected to pin 1 on one end only can be used. If a null modem is required, lines 2 and 3 must be crossed. *See* null modem.

x-ray emission: Emission from the CRT display. Sun workstations fulfill FCC and other local regulations and, therefore, should be of no harm to the operator.

X server: System that acts as an X Window System terminal. It can be software running on a Sun, an IBM RS/6000, a Silicon Graphics computer, a DECstation, or other graphics workstation with X software. An X server can obtain its display information from a local

X client or from an X client running on a remote machine through Ethernet. Apart from UNIX workstations, X server software is also available for the Macintosh and for IBM PC compatibles equipped with Ethernet. Dedicated X terminals are also available. *See* X client, X Window System, X terminal, VnmrX.

X terminal: Special terminal containing X server software built into its firmware.

Xt (X Toolkit): API (application programming interface) used for creating applications under the X Window System. *See* API, X Window System.

X Window System: Standardized windowing software, adapted by Sun, DEC, IBM, Silicon Graphics, and many other computer companies. MIT, the originator of X, defined how graphics software (X client) communicates with the window driver (X server), which can be local software or a remote X terminal (*see* X terminal). The actual X software has to be generated on top of this definition. Sun propagated its own standard for that graphics surface, called OpenLook. X is currently supported on Sun computers by VnmrX, on IBM RS/6000 Powerstations by VnmrI, and on Silicon Graphics IRIS Indigo and Crimson workstations by VnmrSGI. *See* X terminal, X server, X client, VnmrX.

yellow pages: *See* NIS

zero latency: On a disk drive, reading an entire track directly into an intermediate buffer instead of taking time to seek and read a requested sector in the track. This helps in the frequent case where adjacent sectors are also going to be read. With zero latency drives, these additional sectors are then already in the intermediate fast RAM buffer. *See* latency.

zero wait-state: If the memory chips are slower than the CPU chip and the CPU wants to read in or write some information, probably it performs some wait-states. This can drastically affect the performance of a CPU. Some workstations (such as SPARCstations and Ultra workstations) use fast primary and secondary cache memory as an intermediate buffer to reduce the number of wait-states to almost zero. *See* wait states, cache memory.

Symbols

! shell command, 80
!! shell command, 80
# notation, 95, 103
$ notation, 96, 104
& notation, 74
(...) parentheses notation, 74
* wildcard notation, 76
. file name notation, 75
.. file name notation, 76
.cshrc file, 77, 89, 111, 112, 175, 202
.dt directory, 91, 114, 124
.exrc file, 113
.indent.pro file, 113
.login file, 89, 90, 110, 111, 111–112, 205
.mailrc file, 111, 113
.netrc file, 151, 152
.profile file, 113
.rhosts file, 137, 140
.tar file name extension, 172
.tiprc file, 159
.Xdefaults file, 113
.xinitrc file, 113
.xlogin file, 114, 194, 195
/ notation, 75
/var slice
    insufficient space in, 123
/var slice in sufficient space in, 38
> notation, 74
> prompt, 84
>> notation, 74
? wildcard notation, 76
@ notation, 98
[...} wildcard notation, 76
^ notation, 187
^ shell command, 80
`...` notation, 75
| notation, 75
~ notation, 159
~ symbol, 76

Numerics

1/4-inch tape cartridges, 173
10/100baseT Ethernet port, 132
10/100baseT standard, 229
100baseT standard, 229
10baseT standard, 229
16-bit computer, 229
32-bit computer, 229
32-bit format, 70
3D-RAM technology, 229
4.2BSD software, 229
4-track mode, 173
64-bit computer, 229
64-bit operating system, 17, 29
68881 math coprocessor, 61
9-track mode, 173

A

aa command, 213
abort action, 230
absolute paths, 75
access time, 230
acoustic coupler, 230
acq_errors file, 206
acqpresent file, 93
acqproc command, 213
Acqproc process, 155
Acqproc process family, 57
acqstatus parameter, 206
acquisition
    daemon, 94
    status codes, 206
acquisition bus trap, 208
acquisition computer shutdown, 93
acquisition console hostname, 136
acquisition diagnostics terminal, 191
acquisition lock, 125
Acrobat Reader, 230
Acrobat Reader color flashing, 91
Acroread file, 214
active SCSI terminators, 220
active window, 230
ADC overflow warning, 206
adding
    disk space in /var, 38, 123
    printer ports, 191
address, 230
address in Ethernet network, 133
adm directory, 122
admintool command, 126
Adobe Acrobat Reader, 214
AIX software, 194
alias command, 74, 230
alias generation, 112
alias shell command, 80
ALU (Arithmetic Logical Unit), 230
ampersand notation, 74
angled brackets (< or >) notation, 16
anonymous FTP, 230
ANSI C, 230
API (Application Programming Interface), 230
app-defaults file, 114, 124, 199
    dt, 114
    openwin, 114
Apple Macintosh computers, 153
applet program, 231
argument accessibility, 102, 108
argument to command, 231
argument types, 73
argv variable, 102
arithmetic lines in C shell, 98
arithmetic with variables, 104
ARPANET, 231
array processor, 231
artificial shoulders in printout, 186
ASCII (American National Standard Code for Information Interchange), 231
ASIC (application-specific integrated circuit), 231
askbcc variable, 113
askcc variable, 113
assembly language, 231
asterisk wildcard notation, 76
asynchronous serial communication, 190
asynchronous transmission, 231

ATM (Asynchronous Transfer Mode), 231
AUI (attachment universal interface) connector, 232
auto_home file, 149, 154
auto_master file, 149
automatic gain errors, 208
automatic lock errors, 208
automatic mounting, 146–148
automation lock, 125
automount directory, 148
automountd daemon, 149
automounter, 148–149, 154
automounting directory, 121
awk command, 79, 105

B

back straight quotes notation, 75
back up file systems, 171
backbone on Ethernet, 129
background command execution, 74
background job, 81
background lock, 125
background menu, 91
background process, 232
backplane, 232
backslash notation, 16, 73
backup copies, 65
backup files, 232
backup tape, 232
bad track on disk, 232
bandwidth, 232
bar command, 183
basename command, 105
batch mode, 232
batch of data, 232
battery on Sun CPU, 232
baud rate, 158, 186, 233
beeping control, 76
Berkeley file system, 116, 118
beta test, 233
bg shell command, 81
bin directory, 110, 120, 123, 125, 233
binary file, 233
binary file compression, 70
bit, 233
bit bucket, 255
bit operators, 99
block devices, 115, 143
block of data, 233
block size complete, 206
blocking factor, 67, 68, 173, 174, 177
BNC cable, 233
BNC tap kit, 130
BNC T-connector, 131
board, 233
boot blocks, 118
boot monitor command, 84
boot up system, 84
bootup messages, 233
bootup process, 233
Bourne shell, 233
Bourne shell command interpreter, 103
Bourne shell customization, 113
Bourne shell scripts, 103–109
branch cable, 130, 138, 139
break statement, 101
bs parameter, 206
BSD file system, 116
BSD UNIX commands, 124
buffer memory, 233
bug in software, 233
bus arbiter, 233
bus hardware, 233
bus master, 233
bus terminator, 233
button, 234
byte, 234

C

C compiler, 123
C programming bibliography, 225
C programming language, 234
C shell, 89, 234
C shell command interpreter, 95
C shell scripts, 95–103, 112
C source file format, 113
C++ programming language, 234
cache file system, 118
cache memory, 234
cachefs system, 118
caddy for CD-ROM drive, 234
calendar manager, 91
cancel command, 189
capacity of disks or memory, 234
card cage, 234
case construction, 101, 107
case-sensitive, 73, 234
cat /etc/netmasks command, 51
cat command, 79
catman command, 74
ccs directory, 123
cd command, 77
CDE, 45, 90, 234
  console message files, 122
  copying text, 81
  customization files, 114
  definition files, 114
  dtpad text editor, 163
  graphical user interface, 90
  running vi text editor, 168
  setup files, 124
  shortcuts, 81
  software directory, 123
  style manager, 91
  toolbar, 91
CDE toolbar, VnmrJ, VNMR icons in, 27
CD-R disk, 235
cdrom directory, 122
CD-ROM disk, 235
CD-ROM drives, 41
  installing VNMR 5.2, 41
  mounting local, 41
  mounting remote, 42
CD-ROM mount point, 122
CD-ROM mounting, 146
Centronics port, 235
CGI (common gateway interface), 235

change directory, 77
change file ownership, 55, 78
change file permissions, 78
change group ownership, 56
change user-ID to become root, 79
character size, 186
check Ethernet connection, 212
checking the system, 201
chgrp command, 56
child process, 235
chmod command, 54, 56, 78, 110
chown command, 55, 78
CISC (complex instruction set computer), 235
class, 235
classes of Internet addresses, 133
cleaning the system, 201
clicking the mouse, 236
client system, 236
clients, 153
clock, 91, 236
clock frequency, 236
cmdtool command, 92
coaxial cable, 236
Codonics dye sublimation printer, 152
collision between devices, 236
collision of packets, 139
color flashing problem, 91, 214
color names, 114
comma as command separator, 73
command, 236
command grouping, 74
command interpreter, 236
command line, 236
command line interface, 91
command mode, 236
command mode of vi, 164
command output redirection, 75
command separator, 73
command substitution, 75
command tool, 92
comment lines, 95
Common Desktop Environment, 45
Common Desktop Environment. See CDE
compiled programs on remote disks, 145
compiler program, 236
compress a file, 80
compress command, 70, 71, 80, 172
compressfid command, 70
compressing data, 172
compressing files, 70
computer, 236
conditional execution, 100, 106
conditional looping, 100, 107
config macro, 213
configuration, 236
connecting Ethernet hardware, 130
connection timed out, 211
connector hardware, 236
console, 236
console hardware configuration parameters, 212
console messages, 122
constructor, 236
controller ID, 115
conventions used in manual, 16
converged UNIX, 236
coprocessor chip, 236
copy directories between file systems, 175
copy file on a remote host, 140
copy files, 77
core dumps, 202
core files, 205
core memory, 237
cp command, 64, 77, 112, 174, 175
cpio command, 182
CPU (central processing unit), 237
CPU host name, 80
crash from bug, 237
creating new directory, 77
creating shell scripts, 110
Creator graphics, 46
cron command, 65, 69, 205
cron file, 122
crontab command, 205
crontab file, 69, 205
CRT display, 237
csh command, 89
Ctrl-c key combination, 80
Ctrl-d key combination, 76, 77, 80, 92, 112, 148, 158, 159
Ctrl-q key combination, 81
Ctrl-s key combination, 80
Ctrl-u key combination, 77, 194
Ctrl-z key combination, 81
current experiment, 125
current processes, 80
current user name, 80
currently defined aliases, 80
cursor on screen, 237
customization files, 111, 124
customized files, 68
cylinder groups, 237
cylinder on disk, 237
cylinder tracks, 117

D

d1 parameter, 213
daemon program, 237
damaged root partition, 68
DAT (digital audio tape) drives, 237
data, 237
data compression, 69
data directory, 126, 162
data storage disk slices, 126
data transfer testing on Ethernet, 139
date command, 80
daylight-savings time, 51
DC-6150 tapes, 67, 173
DC-9250 tapes, 67, 173
dd command, 171, 176, 177
debugging, 237
DEC VT-100 terminal definition, 168
DECNET for the Sun, 153
decomp command, 178
dedicated phone line, 157, 158
default input, 237
default interactive shell, 95
default operating mode, 86

defaultdomain file, 141
defaultrouter file, 141, 155
defaultsedit command, 113
degaussing a tape, 173
degaussing the screen, 201, 237
deleting files, 238
desktop storage pack, 238
desktop switching, 91
desktop utilities, 91
desktop workstation, 238
destructor, 238
dev directory, 115, 120
developer software, 123
device driver, 238
devicenames file, 185
devices directory, 120
devicetable file, 186
df command, 79, 119, 161, 201
df file prefix for mail, 160
dfs directory, 121
dfstab file, 144, 154
diagnostic software, 238
dialog box, 238
dial-up connections, 157, 158
diff command, 79
differences between files, 79
direct data transfer, 158
directory commands, 77
directory files, 60, 238
directory information, 53
directory levels, 75
directory owner, 55
disabling Ethernet, 140
disk address, 115
disk data storage media, 238
disk drives, 238
disk file errors, 208
disk free percentage, 79
disk full problems with mail, 161
disk ID, 116
disk label, 118
disk layout, 117
disk partitions, 115, 238
disk platters, 117
disk slice, 238
disk space, 119, 238
  adding in /var, 123
  usage for mail, 161
disk space, adding in /var, 38
disk synchronization, 80
diskless clients, 156
DiskSuite software, 60, 117
display, 239
  aliases, 80
  background jobs, 81
  command lines, 80
  CPU host name, 80
  current user name, 80
  date and time, 80
  disk free percentage, 79
  disk information, 80
  file names, 76
  group ID, 80
  process-IDs, 80
  size of file, 79
  system information, 80
  text from text files, 79
  text string, 79
  user ID, 80
  user name, 80
  users logged in, 80
  working directory, 77
  xterm window, 197
DISPLAY variable, 111
DNS (Domain Name Service), 239
documentation for Sun and UNIX, 221
dollar-sign notation, 96, 104
domain name, 50
  finding, 50
domainname command, 50
done codes, 206
dos2unix command, 180
dot files, 75, 111, 239
dot matrix printer, 239
dots per inch (dpi), 239
double straight quotation marks notation, 73
dpi (dots per inch), 239
dpp0 port, 186
dps command, 213
DRAM (dynamic RAM), 239
ds command, 213
dsk directory, 115
dt directory, 123
dtlogin graphical login, 89
DTP (desktop publishing), 239
dtpad command, 163
dtpad text editor, 91
dtterm shell tool, 91
dtterm shell window, 92
du command, 79, 174, 201
dumb terminal, 112
dump command, 177, 202
dump schemes, 67
DVD (digital versatile disk), 239
DVD-R (DVD-Recordable), 239
DVD-RAM, 239
DVD-ROM, 239

E

EBus (external bus), 240
ECC (error checking and correction) memory, 240
echo command, 79, 158
ed text editor, 169
edit text file, 79
edit text stream, 79
editor program, 240
EDITOR variable, 69, 205
EEPROM (electrically erasable programmable ROM), 240
EEPROM settings, 84
EIDE (enhanced IDE), 240
eject command, 178, 180
ejecting a floppy disk, 180
electronic mail (e-mail), 240
Elite graphics, 46
Elite3D graphics accelerator, 240
else statement, 100

emacs text editor, 169
e-mail (electronic mail), 240
e-mail setup, 160
emergency UNIX shutdown, 204
emulation software, 240
encrypted password, 89
end statement, 100
end-of-file mark, 173
end-of-text (EOT) character, 158
environment variables, 96, 104
EOT (end-of-text) character, 158
EPROM (erasable programmable ROM), 240
ergonomics, 240
error codes, 206
error during acquisition, 206
error message, 240
error messages for Ethernet, 138
Esc key, 76
etc directory, 121
Ethernet address, 241
Ethernet boards
    Thinnet Coax Ethernet board, 47
    X453A, 47
Ethernet controller boards, 155, 211
Ethernet hardware, 240
Ethernet interface chip, 136
Ethernet node, 130
Ethernet software, 133
Ethernet structure, 129–130
Ethernet terminator, 241
ethers file, 212
EUnet national branches, 135
ex command mode, 166
ex text editor, 113, 167, 169
Exabyte tape compression, 70
Exabyte tape drives, 241
execkillacqproc script, 57
executable program, 241
execute command on a remote host, 140
execute permission, 53
execution in separate shell, 74
exit command, 92
exit statement, 102, 107
exiting the computing environment, 241
exp1 to exp9, 125
expand command, 187
expansion board, 241
expansion memory, 241
expansion slot, 241
experiment locking, 125
export command, 104
export directory, 121
expr command, 104

F

fast Ethernet, 132, 241
fastboot command, 86
fasthalt command, 86
fb shell command, 81
fdformat command, 178, 179
Federated Naming Service, 122
fiber-optic cable, 241
FID complete, 206
FID file compression, 70
FIFO underflow error, 208
file, 242
file access mode, 242
file compression, 70
file editors customization, 113
file handling commands, 77
file information, 53
file length, 53
file manager, 91
file name rules, 75
file name suffix, 75
file owner, 53, 55
file sizes, 79
file system, 242
file system backup, 171
file system checks, 85
file system mounting, 142
file system overhead, 119
file system scan, 65
file system sharing administration, 121
file system tree, 58
file transfer protocol (FTP), 243
file type, 53
filec option, 76–77
filter program, 242
find command, 78, 147
find files, 78
firewall, 242
firewall software, 142
firmware, 242
flash PROM update, 18, 29
flickering on CRT display, 242
floating point coprocessor chip, 242
floating point number, 242
floating point operation, 242
floating point unit (FPU), 243
floppy directory, 122
floppy disk, 242
floppy disk drives, 178
floppy disk mount point, 122
floppy disk mounting, 143
flow controls in Bourne shell, 104
flow controls in C shell, 99
flush command, 125
fmt command, 79, 189
fn parameter, 213
fold command, 79, 187
font, 242
fonts under OpenWindows, 114
footprint, 243
for statement, 107
foreach statement, 101
foreground command, 80
foreground lock, 125
foreground process, 243
Format Menu, 217
format text, 79
formatting a disk, 243
formatting a text file, 189
formatting floppy disks, 178
Fortran programming language, 243
FPU (floating point unit), 243
fragmentation, 59

fragments for storing files, 243
frame buffer memory, 243
free disk space, 79, 202
Free Software Foundation, Inc., 243
fsck command, 65–66, 85, 118, 121, 122, 123, 146, 147
fsflush daemon, 66
fstab file, 147, 148
FTP (file transfer protocol), 149–152, 243
ftp command, 150
ftptool command, 150
full-duplex terminal session, 159
full-screen editor, 164
function key, 244
function overloading, 244
functions, 244

G

GaAs (gallium arsenide) chip technology, 244
gain parameter, 213
gallium arsenide (GaAs) chip technology, 244
gateway machines, 135, 155, 244
gateway stations, 130
GB (gigabyte), 244
GCC_EXEC_PREFIX variable, 111
gemcon host name, 136
Gemini tape format, 177
gets command, 102
GIF (graphical interchange format), 244
gigabyte (GB), 244
global parameter file, 209
global parameters, 125
GNU (GNU is Not UNIX) software, 244
GNU C compiler, 123
GNU C compiler variables, 111
GNU C/C++ compilers, 244
GNU variable, 111
GNUDIR variable, 111
go monitor command, 84
graphical user interface (GUI), 90, 244
graphics
    hardware, 46
    PGX, 46
graphics controller, 245
graphics variable, 111, 112
graphics workstation, 245
GraphOn terminals, 112
grep command, 79
gripper abort, 207
ground loop, 245
group file, 153
groups file, 55
GUI (graphical user interface), 244
guru, 245

H

hacker, 245
halt command, 86
handshaking, 190
handshaking lines, 245
hanging (stopped) system, 210
hangup status, 245
hard disk drives, 245
    partitions space defaults, 182
hard interrupt, 93
hard links to files, 61–62, 245
hardware, 246
hardware address, 133
hardware diagnostics, 83
hardware maintenance, 201
head command, 79
head crash, 246
header page, 188, 189
help for operators, 246
help monitor command, 84
here documents, 103, 109
Hewlett-Packard Graphics Language (HP-GL), 246
hierarchical file system, 246
hierarchy, 246
hierarchy of processes, 90
high-density floppy disks, 178
high-level programming language, 246
high-noise signal, 206
High-Sierra File System (hsfs), 146
history command, 112
history length variable, 112
history of command lines, 246
history shell command, 80
history standard variable, 96
home directories, 124, 126
home directory, 76, 89, 121, 126, 246
home file system, 117
host address, 133
host address file, 198
host computer, 246
host computer name, 246
host computer number, 133, 246
host disk errors, 208
host name
    finding host name information, 49
    restrictions, 49
host name for acquisition CPU, 136
host name for Magnet and Sample Regulation board, 136
host window, opening a, 45
host workstation requirements, 46
hostname command, 80
hostname, changing the, 211
hostname.* file, 136
hostname.ie0 file, 140
hostname.ie1 file, 155
hostname.le0 file, 139, 140
hosts file, 121, 133, 135, 139, 141, 153, 155, 160, 211, 212
hosts.equiv file, 136, 138, 139, 140
HotJava Web browser, 246
HP-GL (Hewlett-Packard Graphics Language), 246
HP-GL output, 186
HP-GL plotter output, 162
hsfs (High-Sierra File System), 146
HTML (HyperText Markup Language), 246
HTTP (HyperText Transfer Protocol), 247
hub, Ethernet, 247
hypertext link, 247
HyperText Markup Language (HTML), 246
HyperText Transfer Protocol (HTTP), 247

I

I/O (input/output), 247
ib_initdir directory, 124
IBM RS/6000
    using as X server, 194
icon, 247
id command, 80
ID of print request, 189
IDE (Integrated Device Electronics), 248
if statement, 100, 106
ihwinfo command, 208
image browser setup files, 124
include directory, 123
increasing space on floppies, 182
indent command, 113
individual tape backups, 69
init 0 shutdown command, 45
init command, 85, 86, 204
init.d directory, 88
inittab. init file, 88
ink jet printer, 248
inline tap kit, 130
I-node (index-node), 247
I-node numbers, 60
I-node table, 118
INOVA IP address, 135
INOVAAUTO IP address, 135
insert mode, 166, 248
insertion point, 248
installation of Sun workstations, 248
installboot command, 118
installing
    printers, 185
installing Solaris patches, 38
integer value, 248
integration level, 248
interactive input, 102, 109
interactive mode for commands, 112
interface device, 248
interlacing the CRT display, 248
interleave factor (IL), 248
Internet address, 133–135, 211, 212
Internet bibliography, 224–225
Internet network, 248
Internet Protocol (IP), 249
Internet Protocol network address, 50
Internet Service Provider (ISP), 134, 249
internetworked network, 134
interpreter program, 249
interrupt signal, 249
intranet network, 249
invisible file, 249
IP (Internet Protocol), 249
IP address, 20, 32, 50
    finding, 50
IP address of router, 141
ISDN (Integrated Systems Data Network), 249
ISP (Internet Service Provider), 249

J

Java programming language, 249
JavaScript, 250
jexp command, 126
jobs shell command, 81
JPEG (Joint Photographic Experts Group), 250
jukebox device, 250
jumper, 250

K

KB (kilobyte), 250
Kermit protocol, 162
kernel directory, 121
kernel program block, 250
keyboard, 250
keyboard input, 74
keyboard layouts, 124
keyboard shortcuts, 76
kill command, 80, 203
killing a program, 250
kilobyte (KB), 250

L

L1 key, 204, 210
LAN (local-area network), 130, 250
laser printer, 250
LaserJet printer, 186, 250
last line mode of vi, 166
latency time, 250
LD_LIBRARY_PATH variable, 114
left-shift all arguments, 102
level 0, 83–84
level 0 and level 1 dumps, 66
level 1 run mode, 85
level 2 run mode, 85
level 3 and level 5 dumps, 68
level 3 run mode, 86
Lexmark color printer, 250
lf command, 77
lf file prefix for mail, 161
lib directory, 121, 123, 124
license manager, 250
limNET Ethernet protocol, 153
line conditioner, 251
line-oriented text editors, 168
link to a file, 78
linked network, 134
linker program, 251
linking to VNMR software directory, 124
list files, 77
ll command, 77
ln command, 61, 78
local area network (LAN), 250
local host, 133
local-area network (LAN), 130
localhost host name, 135
lock type, 125
lock_1.primary file, 125
log file, 122
log out of system, 80
loghost host name, 135
login process, 56, 89, 251
login shell, 251
login shell script, 111
logout command, 159, 205

logout process, 251
looping, 100, 107
lost+found directory, 66, 121, 122, 123, 182
low-density floppy disks, 178
low-noise signal, 206
lp command, 187, 189
lp device, 186
lp directory, 121, 122
lp script, 88
LPDEST variable, 189
lpr command, 188
ls command, 53, 74, 75, 77, 111, 112, 180

M

Macintosh as X server, 195
maclib directory, 62, 126
maclibpath parameter, 127
macro file, 251
macroedit command, 163
macrovi command, 163
MAGICAL (MAGnetics Instrument Control and Analysis Language), 251
MAGICAL command language, 92
Magnet and Sample Regulation (MSR) board, 122
    host name, 136
magnetic tape, 172
magneto-optical disk (MOD), 251
mail administration files, 121
mail client machine, 160
mail command, 113
mail directory, 121, 122
mail problems, 161
mail software configuration, 113
mail tool, 251
Mail Tool program, 113
mailhost, 160
mailtool command, 113, 160
maintenance contract, 251
makesuacqproc command, 57
makeuser command, 56, 111
makeuser file, editing, 27
man command, 74
man directory, 124
MANPATH variable, 111
manual directory, 62, 206
manual notational conventions, 16
manuals, importance to users, 251
mass storage device, 252
Master NIS file server system, 211
master server, 153
MB (megabyte), 252
MC68040 CPU, 252
mean time between failures (MTBF), 254
megabyte (MB), 252
megaflops (million floating point operations per second), 252
memory amount, 46
memory buffer exceeded, 210
memory management unit (MMU), 253
memsize variable, 111
menu, 252
menu button, 252
mezzanine board, 252
MFlops (million floating point operations per second), 252
microprocessor, 252
MII (Media Independent Interface), 253
million instructions per second (MIPS), 253
mini UNIX, 253
minus sign notation, 73
MIPS (million instructions per second), 253
mkdir command, 77
mkfile command, 209
MMU (memory management unit), 253
MOD drive, 253
modem (modulator-demodulator), 253
modem connections, 157
modem types, 159
monitor diagnostic, 253
monitor mode, 83–85
monitor screen, 253
more command, 79, 112
Mosaic Web browser, 253
motherboard, 253
Motif graphical user interface, 254
Motif interface, 91
Motorola CPU chips, 254
mount
    local CD-ROM, 41
    remote CD-ROM, 42
mount command, 143, 146, 180, 181
mount directory, 142, 254
mountall command, 147, 154
mountd daemon, 145
mounting, 254
mounting file systems, 142
mounting floppy disks, 143, 180
mounting limitations, 145
mounting removable media, 57
mouse, 254
mouse buttons, 254
mousepad, 254
move files, 78
MPEG (Motion Picture Experts Group), 254
mqueue file, 160, 161
MS-DOS floppy disk format, 179
MTBF (mean time between failures), 254
multimedia applications, 254
multiple hard links to a file, 61
multiprocessor system, 254
multisegment tapes, 176
multitasking system, 255
multithreading, 255
multiuser backups, 69
multiuser mode, 83
multiuser system, 255
mv command, 78, 145

N

n monitor command, 84
name servers, 141
name service, 50
    finding the type, 51
name variables, 104
nd.local file, 212
net directory, 121, 148

netmask, 134
netmasks file, 135
Netscape Navigator Web browser, 255
netstat command, 212
network, 255
Network File System (NFS), 143, 255
Network Information Service (NIS), 153, 255
Network Information Services (NIS), 211
network number, 133, 134
network security bibliography, 224
network system security, 136
networking, 129–155
networking bibliography, 223–224
networks file, 153
newfs command, 66, 119, 181, 182
NFS (Network File System), 153, 255
NFS client machine, 144
nfs directory, 145
NFS mounting, 121, 154
NFS mounting options, 144–145
nfsd daemons, 145
NFS-type mounting, 143
NIS (Network Information Service), 255
NIS clients, 153
NIS or NIS+ server, 50
nmr group, 56
NMR system administrator, 126
node in network, 255
nodename file, 136
notational conventions, 16
notrouter file, 155
np parameter, 208, 213
nroff command, 74, 79
nsswitch.conf file, 142
nt parameter, 213
N-type connectors, 130
null device, 255
null modem, 158, 159, 189–190, 255
null modem cable, 256
number-sign notation, 95, 103
NVRAM (nonvolatile RAM), 256

O

object file, 256
object libraries, 123
Object-Oriented Programming Style (OOPS), 256
objects, 256
off-line task, 256
ok prompt, 84
old-mode monitor command, 84
online process, 256
OOPS (Object-Oriented Programming Style), 256
open access files, 256
Open Look
    configuration file, 113
    screen background, 113
    shortcuts, 81
    window manager, 113
Open Systems Foundation (OSF), 257
OpenGL application programming interface, 257
opening
    a host window, 45
openwin directory, 123
OpenWindows, 257
    configuration file, 113
    fonts possible, 114
    software directory, 123
OPENWINHOME variable, 114
operating system program, 257
operator interface devices, 257
operator overloading, 257
opt file system, 117
opt slice, 123
optical media mounting, 57
optional software slice, 123
OSF (Open Systems Foundation), 257
owner of file, 257

P

p1 parameter, 213
packet collision information, 212
packets, 139, 257
page command, 79
paging, 257
parallel computing, 257
parallel I/O, 258
parallel port, 186
parent directory, 76, 258
parentheses notation, 74
parity checking, 258
partition, 258
partition map, 118
partitioning the disk, 209
partitions on hard disks, 115
Pascal programming language, 258
passive mode, 196
passwd file, 55, 89, 121, 126, 153
password, 258
password access, 136
password aging mechanism, 210
password checking, 89
password file, 126
password invalid, 210
password protection, 202
password selection, 138
passwords, 52
patchadd command, 38
patches, 37
patches installation log, 122
path for shell scripts, 110
path to a file, 75
PATH variable, 168
path variable, 111, 114
PC computers as X server, 195
PC Workstation IP address, 136
pcfs directory, 180
PCI bus, 258
PCL output, 186
PCMCIA (Personal Computer Memory Card Interface Association), 258
PCs, running Microsoft Windows, 153
PDF (Portable Document Format), 258
performance measures, 259
performance meter, 91, 132
performance, computers, 258
peripheral, 259

peripheral component interconnect. See PCI
Perl programming language, 259
permission flags, 54
permission to access file, 259
personal computers
  starting VnmrJ, 195
  X server software, 195
PGX graphics, 46
PGX graphics accelerator, 259
piggyback board, 259
ping command, 132, 138, 212
pipe, 75, 259
pixel, 259
pizza box CPU module, 259
platform directory, 121
plotter definition, 185
plotter device, 259
plotter parameter, 186
plotters characteristics, 186
PlotView software package, 162
Point-to-Point Protocol (PPP), 260
Point-to-Point Thinnet Network, 130
port, 259
portability, 260
PostScript output, 186
PostScript page description language, 260
power failure, 65
power failure effects, 260
power switch, 260
power switch turnoff order, 203
ppmm parameter, 186
PPP (Point-to-Point Protocol), 260
precision of data word, 260
preinstallation worksheet, 49
primary group membership, 55
primary network interface, 21, 31, 49
Printcap entry, 186
printcap file, 260
printenv monitor command, 84
printer administration files, 121
printer definition, 185
printer device, 260
printer installation, 185
printer ports, 191
PRINTER variable, 111
printers characteristics, 186
printing and plotting under VNMR, 187–188
printon, printoff commands, 188
proc directory, 121
procedures definition, 109
process families, 90
process in program execution, 260
processes list, 121
procpar file, 70
PROM-based software level, 83
prompt, 260
prompt format, 112
prompt standard variable, 96
props button, 171
protection bits, 53, 53–55
protocol, 261
prtvtoc command, 80, 118
ps command, 80
public data networks, 157
public e-mail access, 135
pw parameter, 208, 213
pwd command, 75, 77

Q

qf file prefix for mail, 161
QIC (quarter-inch cassette), 261
QIC tape format, 173
Qtune file, 199
quitting the shell window, 92

R

RAID (redundant arrays of inexpensive disks), 261
RAM (random-access memory), 261
RAM requirements, 46
raster entry, 186
raw devices, 115, 143
rc files, 111
rc.boot file, 212
rc0 to rcS startup scripts, 87
rcp command, 136, 137, 139, 140, 149, 153
rdsk directory, 115
read files to tape, 171
read permission, 53
read tape, 78
reading from a tar file, 174
reading tapes of any format, 177
reboot command, 86
rebooting UNIX, 204
receiver overflow warning, 206
receiving mail, 160
recovering damaged files, 68
recursion, 261
recursive copy of a file, 78
reexecute last command line, 80
refresh rate, 262
rehash command, 110
relative paths, 75
release of software, 262
remote computer actions, 262
remote control files, 111
remote file, 159, 191
remote file system mounting, 143
remote home directories, 154
remote host commands, 140
remote login, 137, 211
remote login on a remote host, 140
remote printing and plotting, 140
remote tape drive, 176–177
remote X server, 112
remote X terminals, 114
removable magnetic tape, 172
removable media mounting, 57
remove
  alias, 80
  directory, 77
  files, 78
reset monitor command, 84
resetting the console, 210
resolv.conf file, 141
restarting UNIX, 204
restore command, 177

Return key, 16, 73
REXEC command protocol, 195
RG/58U coaxial cable, 131
rgb.txt file, 114
RISC (reduced instruction set computer), 262
RJ45 connector, 262
rlogin command, 137, 139, 140, 153, 194, 195, 211, 212
rm command, 61, 78, 112
rmdir command, 77
ROM (read-only memory), 262
root access to NFS mounted systems, 145
root menu, 262
root partition, 117
root partition size, 161
root password, 85
root slice, 120
root superuser, 262
rooted environment, 195
rotational latency, 118
route command, 141
RS-232 cable, 206
RS-232 expansion board, 189
RS-232 system connection, 157
RS-232, RS-423 serial communications, 262
rsh command, 136, 137, 139, 140, 153, 194
RSH command protocal, 195
run levels, 83
run-time library, 123

S

sadm directory, 122
safety concerns, 263
sample changer errors, 207
sbin directory, 122, 124
SBus internal bus, 263
scaning the file system, 65
scheduler program, 263
screen blanking, 263
screen burn-in, 201, 263
screen locking, 263
scroll bar, 263
SCSI (Small Computer Systems Interface), 263
SCSI bus, 116
SCSI bus, using an active terminator, 220
SCSI disks, 115
SCSI errors, 208
SCSI terminators, active, 220
    problems, 220
SCSI-ID, 116
search for pattern in file, 79
secondary lock file, 125
sector of disk, 263
security patches, 57
sed command, 79, 187
seek time, 263
semicolon command separator, 73
serial I/O, 264
serial link between systems, 157
serial port, 186
server computer, 264
set command, 76
set statement, 96
setenv command, 84, 98, 195
set-GID bit, 56
setnoether command, 132
set-UID bit, 56
SGML (Standard Generalized Markup Language), 264
shadow file, 89, 153
share command, 144, 146
share directory, 124
shareall command, 145
sharetab file, 43
shell, 264
shell for commands, 74
shell script, 264
shell scripts, 95
shell tools, 91, 92
shielded cable, 264
shielded twisted pair (STP), 267
shielding for rf, 264
shift command, 102
shimming errors, 208
showconsole command, 212
showrev command, 38, 57
shutdown
    command, 45
    procedure, 44
shutdown command, 86, 93
shutting down
    acquisition computer or cabinet, 93
    entire system, 93
    UNIX, 202
shutting down UNIX, 44
    immediately, 45
    to install Solaris, 44
SIMM (single in-line memory module), 264
single-user mode, 83, 85, 264
slash notation, 75
slave servers, 153
slice, 265
slice number, 116
slices of hard disks, 115
Slim Box
    IP address, 136
SLIP (Serial Line Internet Protocol), 265
SMD (surface mounted device), 265
software
    compatibility, 46
software group, 46
software handshaking, 190
software installation log, 122
software maintenance, 201, 265
software types, 265
Solaris
    collecting system information, 49
    patch installation, 38
    patches, 37
    preinstallation worksheet, 30, 49
    shutting down after wait, 45
    shutting down immediately, 45
    shutting down the system, 44
    upgrade from SunOS, 49
    version 2.6, 17, 29
    version 8, 17, 18
    versions compatible with VNMR, 46

Solaris 1.1 operating system, 265
Solaris 9
preinstallation worksheet, 18
Solaris patches installation log, 122
Solaris software groups, 46
Solstice network software, 265
sort command, 79
sort lines in text file, 79
source command, 114
source files, 123
space requirements, 46
SPEC (System Performance Evaluation Cooperative), 265
SPECfp92 performance measure, 266
SPECfp95 performance measure, 266
SPECint92 performance measure, 266
SPECint95 performance measure, 266
SPECmark performance measure, 266
SPECrate performance measure, 266
spectrometer host, 155
spinner errors, 206
spool directory, 122
spooled data storage, 122
spooling directory, 161
spray command, 139
square brackets wildcard notation, 76
SRAM (static RAM), 266
stack register, 266
staff group, 56
standalone system, 49
stand-alone workstation, 266
standard input, 74
standard printer, 186
STAPE command, 177
startacqproc command, 93, 205
startup commands, 122
static discharge, 266
sticky bit, 56
Stop key, 204, 210, 211
Stop-a key combination, 84, 93
storing directories on tape, 171
STP (shielded twisted pair), 267
streaming tape mode, 173
structured programming, 267
stty command, 158
Style Manager, 91
su acqproc command, 93, 213
su command, 79
subnet mask, 51
subnet mask, finding the number, 51
suffix to filename, 75
summary block, 118
sum-to-memory error, 208
Sun computer
models, 46
Sun FPA (floating point accelerator), 61
Sun Microsystems, 267
Sun patch FTP server, 37
Sun terminal, 112
Sun workstation documentation, 221
Sun workstations
architecture and SunOS versions, 46
suninstall command, 133, 135
SunOS
shutting down after wait, 45
shutting down immediately, 45
SunSoft division, 267
SunSolve Online Web site, 37
Suntools, 267
superblock, 118
Supercache memory, 267
supermini computers, 267
superuser, 267
surface analysis, 267
surface mounted device (SMD), 265
sw parameter, 213
swap command, 122, 209
swap space, 117, 122, 209, 267
swapping processes, 267
symbolic links, 63–64, 78, 174, 268
sync command, 65, 80, 84
synchronize system disk, 80, 84
synchronous data transfer, 158
synchronous transmission of data, 268
syslog file, 122
system administration commands, 79, 124
system administration files, 122
system processes list, 121
system recovery commands, 122
system shutdown, 93
system startup scripts, 87
System V UNIX, 116, 268
sys-unconfig command, 133, 211

T

T junction, 130
tail command, 79
tap kit, 130, 269
tape backup, 65, 66, 202
tape cassette, 268
tape catalogs, 174
tape command, 177
tape device configuration, 172
tape drive on remote system, 176
tape drives, 268
tape lengths, 67
tape magnetic medium, 268
tape rewinding, 176
tape storage, 171
Tape Tool program, 171
TAPE variable, 111, 172, 175
tar command, 64, 69, 70, 73, 78, 171, 179, 202
tar file, 172, 173
tar file system, 179
tar tape archive program, 269
target address or ID, 115
task, 269
TB (terabyte), 269
Tcl/Tk (Tool command language /Tool kit), 269
Tcl/Tk variables, 111
TCL_LIBRARY variable, 111
TCP (transport control protocol), 269
TCP/IP (transport control protocol/internet protocol), 153, 269
Tektronics terminals, 112
telnet, 195
client startup, 197

session startup of X client, 196–197
temporary files, 122
terabyte (TB), 269
TERM signal, 80
TERM variable, 168
term variable, 92, 96, 112, 168
terminal emulation software, 162
terminal information database, 124
terminal interface, 269
terminal types, 112
terminate processes, 80
terminator, 269
terminator device, 131
test function, 99, 104, 106
text commands, 79
text editing programs, 163–169
text file, 269
text file compression, 70
text processing, 269
textedit command, 79, 81
textedit text editor, 269
tftpboot directory, 122
then statement, 100
thick Ethernet cabling, 130, 131
thin Ethernet, 270
thin Ethernet coaxial cable, 131
thinnet coax second Ethernet board, 47
Thinnet Ethernet cable standard, 130
Thinnet network, 131
TIFF, 270
tilde sign notation, 76, 159
time sharing operating system, 270
time zone definitions, 124
tip command, 159
TK_LIBRARY variable, 111
tmp directory, 122, 187
tmp file system, 122
tmp partition fills up, 210
tof parameter, 213
tool, 270
toolbar, 91
top-down programming, 270
touch command, 78, 93
tpwr parameter, 213
tr command, 79
transceiver, 130, 270
transceiver for Ethernet, 131
transfer rate for Ethernet, 139
transfer rate over interface, 270
transferring directories, 172
transferring large files via floppies, 182
translate characters, 79
tree structure for a file system, 59
tree structure of directories and files, 270
troff command, 79
troubleshooting, 206
acquisition status codes, 206
Ethernet, 211
system hangs, 210
troubleshooting activities, 270
trusted hosts, 137
T-switch, 268
ttya file, 158
ttya, ttyb ports, 186
ttymon process, 89
tunefs command, 119
twisted-pair Ethernet, 271
twisted-pair Ethernet (TPE), 129, 132
two dots file name notation, 76

U

ucb directory, 124
ucbinclude directory, 124
ucblib directory, 124
UDP (user datagram protocol), 271
ufs (UNIX file system), 116, 118
parameters, 119
ufsdump command, 66, 67, 171
ufsrestore command, 68, 69, 171
umount command, 143, 144, 147, 180
umountall command, 147
unalias command, 80
uname command, 80
uname -n command, 49
uncompress command, 70, 71, 80
uncompress file, 80
unconditional looping, 101, 107
unformat floppy using stray field, 179
UNIX, 271
administration files, 121
argument separator, 73
bibliography, 221–222
command names, 73
command separator, 73
documentation, 221
emergency shutdown, 204
file name rules, 75
file system, 116, 180
file system generation, 181
file system tree, 58
global environment variables, 111
important commands, 77
kernel, 121
manuals, 124
online manuals, 221
programming bibliography, 225–226
restarting, 204
security bibliography, 224
shell interface, 91
shutting down the system, 44
software file system, 123
tools bibliography, 222–223
UNIX International (UI), 271
unix2dos command, 180
UNIX-to-UNIX copy, 159
unknown host message, 198
unlock command, 126
unset command, 76
unshare command, 145
unshielded twisted pair (UTP), 272
update daemon, 66
updates file dates, 78
URL (uniform resource locator), 271
usable disk space, 119
user file, 69
user of UNIX software, 272
user partition, 67, 272

user standard variable, 96
user_templates directory, 111
users currently logged in, 80
usr file system, 117, 123
utility programs, 272
UTP (unshielded twisted pair), 272
uucp command, 159, 160

V

vampire tap kit, 130
var file system, 117
var slice, 122
Varian NMR User Pages, 37
varian.xicon file, 113
VAX computer under VMS, 153
vertical bar notation, 75
vfstab file, 143, 146, 154, 180, 182
vi command, 79
vi text editor, 92, 113, 164, 272
    command mode, 164
    insert mode, 166
    last line mode, 166
    problems, 167
    running remotely, 168
virtual file system, 118
virtual memory, 272
virtual terminals, 194
virus software bug, 272
VIS (Visual Instruction Set), 272
VME bus structure, 272
vmunix file, 121
vn command, 112
VNMR, 272
    application configuration files, 114
    argument separator, 73
    compatibility with Solaris versions, 46
    customization files, 111
    file name extensions, 75
    programs, 92
    shell, 92
    system variables, 111
vnmr directory, 126
VNMR Online, 91
vnmr1 file ownership, 55
vnmr1 NMR system administrator, 80
vnmr1 user name, 126
vnmreditor variable, 111, 163
VnmrJ
    multiuser setup, 126
    software directory, 124
    startup from personal computer, 195
    user's home directories, 124
    workspaces, 125
vnmrj, 194, 195
vnmrj command, 194, 195
vnmrj directory, 124
vnmrmenu variable, 111
vnmrplot shell script, 188
vnmrprint shell script, 186, 187–188
vnmrsys directory, 125, 126
vnmrsystem variable, 111, 124
vnmrtext variable, 111
vnmruser variable, 111
vol directory, 122
vold process, 179
vold volume daemon, 57
volmgt file, 183
volmgt script, 88
volume management, 88, 182
volume manager, 122
VRAM memory chips, 272
VRML (Virtual Reality Modeling Language), 273
VT errors, 207
vttime parameter, 207
VXR-4000 tape format, 177
VxWorks operating system, 122

W

WAN (wide-area network), 273
warning error codes, 206
wbs command, 206
wc command, 74
Web (WWW, World Wide Web, W3), 274
Web authoring, 273
Web authoring tool, 273
Web browser, 273
Web browsers and color flashing, 91
Web page, 273
Web server, 273
werr command, 206
wexp command, 206
while statement, 100, 107
who am i command, 80
who command, 80
wide-area networking (WAN), 162
wildcard characters, 76
Winchester disk, 273
windowless multiuser mode, 90
windows, 273
wnt command, 206
word count, 74
word data size, 274
word processing, 274
working directory, 75
workstation, 274
world time zones, 51
World Wide Web (W3, Web, WWW), 274
WORM drive, 274
wrap text lines, 79
write files, 78
write files to tape, 171
write permission, 53
WWW (World Wide Web, W3, Web), 274
WYSIWYG (what you see is what you get), 274

X

X client, 193, 274
X host, 193
X server, 112, 193–??, 274
    setting up, 193
X terminal, 114, 193, 275
X Toolkit (Xt), 275
X Window System, 275
    bibliography, 226–227
    customizing the X environment, 199

X453A Thinnet Coax Ethernet board, 47
second, 47
xcharp1 parameter, 186
xdm graphical login, 86
xf file prefix for mail, 161
xfn directory, 122
xhost command, 112, 193
xlsfonts command, 114
XON/XOFF data control, 274
x-ray emission from CRT, 274
Xt (X Toolkit), 275
xterm window, 195–??
controlling appearance, 198
displaying, 197

Y

ycharp1 parameter, 186
yellow pages service, 153
ypcat command, 50
ypwhich command, 211
ypwhich command, 51

Z

zcat command, 72
zero latency drive, 118, 275
zero wait-state, 275