



UniVerse

Administering UniVerse

Version 10.2
September, 2006

IBM Corporation
555 Bailey Avenue
San Jose, CA 95141

Licensed Materials – Property of IBM

© Copyright International Business Machines Corporation 2006. All rights reserved.

AIX, DB2, DB2 Universal Database, Distributed Relational Database Architecture, NUMA-Q, OS/2, OS/390, and OS/400, IBM Informix®, C-ISAM®, Foundation.2000™, IBM Informix® 4GL, IBM Informix® DataBlade® module, Client SDK™, Cloudscape™, Cloudsync™, IBM Informix® Connect, IBM Informix® Driver for JDBC, Dynamic Connect™, IBM Informix® Dynamic Scalable Architecture™ (DSA), IBM Informix® Dynamic Server™, IBM Informix® Enterprise Gateway Manager (Enterprise Gateway Manager), IBM Informix® Extended Parallel Server™, i.Financial Services™, J/Foundation™, MaxConnect™, Object Translator™, Red Brick® Decision Server™, IBM Informix® SE, IBM Informix® SQL, InformiXML™, RedBack®, SystemBuilder™, U2™, UniData®, UniVerse®, wIntegrate® are trademarks or registered trademarks of International Business Machines Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Windows, Windows NT, and Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names used in this publication may be trademarks or service marks of others.

This product includes cryptographic software written by Eric Young (eay@cryptosoft.com).

This product includes software written by Tim Hudson (tjh@cryptosoft.com).

Documentation Team: Claire Gustafson, Shelley Thompson

US GOVERNMENT USERS RESTRICTED RIGHTS

Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Table of Contents

Preface

Organization of This Manual	xv
Documentation Conventions.	xviii
UniVerse Documentation.	xx
Related Documentation	xxii
API Documentation	xxiii

Chapter 1

Introduction

Introduction	1-2
What Is UniVerse Administration?	1-3
Who Is a UniVerse Administrator?	1-4
UniVerse Files	1-5
Nonhashed Files	1-5
Hashed Files	1-5
B-Tree Files	1-6
UV Account Directory Files	1-6
UniVerse Administration Commands	1-8
Assigning and Unassigning Peripheral Devices	1-8
UniVerse BASIC Programs	1-8
File and Account Maintenance	1-9
Managing Processes.	1-11
Monitoring Resource	1-12
Task and Record Locking	1-13
Managing Transaction Logging	1-13
National Language Support	1-14

Chapter 2

UniAdmin

Starting UniAdmin.	2-3
----------------------------	-----

Chapter 3	System Startup and Shutdown	
	Starting Up and Shutting Down UniVerse on UNIX Systems	3-3
	Configuring UniVerse Parameters at Initialization Time	3-3
	Logging On	3-3
	Initializing the UniVerse Shell	3-3
	Starting and Stopping UniVerse on Windows Systems	3-5
	Starting UniVerse Services	3-5
	Stopping UniVerse Services	3-6
	Overview of UNIX Startup and Shutdown	3-9
	UNIX Startup	3-9
	UNIX Shutdown	3-10
 Chapter 4	 Configurable UniVerse Parameters	
	The <i>uvconfig</i> and <i>.uvconfig</i> Files	4-2
	The <i>uvregen</i> Program	4-2
	Allocating Shared Memory	4-2
	Recovering <i>.uvconfig</i>	4-3
	The Configurable Parameters	4-4
	Changing Configurable Parameter Values	4-18
	Using UniAdmin to Change Parameter Values	4-19
	Editing the <i>uvconfig</i> File	4-21
	The Default <i>uvconfig</i> File	4-22
 Chapter 5	 Adding and Maintaining UNIX User Accounts	
	General Considerations	5-2
 Chapter 6	 UniVerse Accounts	
	About UniVerse Accounts	6-3
	Creating a New UniVerse Account	6-4
	Creating a New Account on a UNIX System.	6-5
	Creating a New Account on Windows Platforms	6-6
	Viewing or Modifying Account Details	6-9
	Deleting an Account	6-10
	Deleting a UniVerse Account on a UNIX System	6-10
	Deleting a UniVerse Account on a Windows System	6-11
	Customizing UniVerse Accounts	6-13
	UniVerse Account Control Files	6-13
	Essential UniVerse Files	6-16
	Controlling Access to UniVerse on UNIX Systems.	6-18
	Controlling Access to UniVerse on Windows Platforms	6-19

Customizing a UniVerse Account	6-19
--	------

Chapter 7 **Transferring Accounts**

Transferring Non-UniVerse Accounts	7-3
Manually Restoring Accounts from Tape	7-6
Restoring Accounts to UNIX Systems.	7-6
Restoring Accounts to Windows Systems	7-15
Transferring UniVerse Accounts from UNIX to Windows Platforms	7-22
Creating the Backup Image	7-22
Transferring the Backup Image	7-23
Restoring the Backup Image.	7-23
File Naming Conventions	7-24

Chapter 8 **UNIX System Security**

Security Overview	8-3
User Permissions and File Permissions	8-3
File Permission Modes	8-5
Using the <i>umask</i> Command	8-5
Protecting User Accounts with Passwords	8-6
Assigning a Password	8-6
Making a Nonlogin Account.	8-7
Using Groups Effectively	8-8
Defining Groups	8-8
VOC File Security	8-10
Security Subroutines	8-11

Chapter 9 **Managing Locks**

Record Locks and File Locks	9-3
Shared Record Lock	9-4
Update Record Lock	9-5
Shared File Lock	9-6
Intent File Lock.	9-6
Exclusive File Lock	9-7
Transactions and Locks	9-8
Managing Locks with UniVerse Admin	9-9
File and Record Locks	9-10
Group Locks.	9-11
Clearing Locks	9-12
Managing Deadlocks.	9-13
Starting and Stopping the Deadlock Manager	9-15

Configuring Deadlock Management	9-17
Using the <i>uvdlockd</i> Command	9-18
Resolving Deadlocks Automatically	9-19

Chapter 10 **Configuring Peripheral Devices**

The &DEVICE& File	10-3
Administering the &DEVICE& File	10-4
Configuring Tape Drives	10-5
Defining a New Tape Drive on a UNIX System.	10-5
Defining a New Tape Drive on a Windows Platform	10-9
Viewing and Modifying a Tape Drive Definition	10-12
Using the Test Tape Button	10-12
Deleting a Tape Drive Definition	10-12
Configuring Other Devices	10-13
Defining a New Device	10-13
Viewing and Modifying a Device Definition.	10-14
Deleting a Device Definition	10-14
Configuring Terminals on UNIX Systems	10-15
Terminal Line Naming Conventions	10-15
Setting Default Terminal Characteristics	10-16
The <i>terminfo</i> Facility	10-19
Customizing Terminal Capabilities While Logged On.	10-28
Mapping Terminals and Auxiliary Printers	10-29

Chapter 11 **Administering Printers and the UniVerse Spooler**

Configuring Printers	11-4
Defining a New Printer	11-4
Viewing and Modifying a Printer Definition.	11-7
Deleting a Printer Definition.	11-7
Defining and Administering Printer Groups	11-8
Defining a Printer Group	11-8
Adding Users or Printers to a Printer Group	11-11
Removing Users or Printers from a Printer Group	11-11
Deleting a Printer Group	11-12
Managing Printers	11-13
Mounting Forms on a Printer	11-13
Setting Printer Queuing Options	11-14
Starting and Stopping Printers	11-14
Configuring the UniVerse Spooler	11-15
Menu Bar.	11-16

Printer Information	11-17
Jobs List	11-17
Task Buttons	11-18
Changing the Spooler Configuration	11-19
Managing Print Jobs	11-22
Changing Print Job Characteristics	11-22
Controlling Print Jobs	11-25
Logging Spooler Activity	11-28
Displaying Spooler Log Files	11-28
Determining When a Job Was Printed	11-29
Starting, Stopping, and Resetting the Spooler	11-31
Starting the Spooler	11-31
Stopping the Spooler	11-31
Resetting the Spooler	11-31
About the UniVerse Spooler	11-33
What Happens When the Spooler Is Installed	11-33
Spooler Directories and Files	11-33
Spooler Processes and Commands	11-35
How the Spooler Works	11-36
Using UniVerse Spooler Printer Drivers	11-38
Using a UNIX Executable as a Driver	11-38
The Bourne Shell as a Driver	11-39
Using a Driver for Remote Printing	11-40
Complex Shell Script Drivers	11-40
Setting Interface Characteristics in a Driver	11-41
Capturing Spool Output	11-42
Using Command Line Arguments in Driver Scripts	11-42
Using the UNIX Spooler with the UniVerse Spooler	11-44
Changing the UNIX <i>lp</i> Interface File	11-44
Adding a DRIVER Option to the <i>sp.config</i> Entry	11-45
Troubleshooting the Spooler	11-46
Printing Problems	11-46
Getting Incorrect Printout	11-53
Frequently Asked Questions	11-57

Chapter 12 **Backing Up and Restoring Files**

Backup Strategies	12-3
Backing Up Individual Files	12-4
Four Ways to Back Up and Restore Files	12-6
Preserving the Integrity of Your Data	12-6

Backing Up Files	12-7
Backing Up to Multiple Tapes	12-9
Using T.DUMP to Back Up UniVerse Files.	12-11
Using <i>uvbackup</i> to Back Up Files.	12-12
Specifying the File List	12-13
Restoring Files	12-15
Choosing the Restore Device	12-15
Checking the Backup Details	12-16
The UVRestore Window	12-18
Choosing What to Restore	12-19
Listing an Index of the Backup Image.	12-20
Specifying How to Restore Files	12-21
Using T.LOAD to Restore UniVerse Files	12-23
Using <i>uvrestore</i> to Restore Files	12-24
Specifying Files and Records to Restore	12-24
Excluding Files to Restore	12-25
Display Options.	12-25
Other Options	12-25
Some UNIX Backup and Restore Commands	12-27
Using <i>cpio</i> to Back Up and Restore Files	12-27
Using <i>tar</i> to Back Up and Restore Files	12-27

Chapter 13 **Managing Data Replication**

Replication	13-4
Hot Standby	13-5
Setting Up Data Replication	13-6
The Replication Window	13-8
Menu Bar.	13-8
Toolbar	13-10
Left Pane	13-10
Right Pane	13-11
Configuring and Managing Data Replication	13-12
Managing a Publishing System	13-13
Configuring the Publishing System	13-13
Starting and Stopping the Publishing System	13-15
Publishing Files.	13-15
Managing a Subscribing System	13-24
Configuring the Subscribing System	13-24
Starting and Stopping the Subscribing System	13-25
Creating a Subscriber's List of Publishing Systems	13-25

Subscribing Files	13-27
Managing Hot Standby Operations	13-35
Configuring a Hot Standby Subscriber	13-35
Turning On Fail-Over Mode	13-37
Reconciling the Hot Standby with the Publisher	13-37
Some Restrictions.	13-40
What to Do When Disk Space Fills Up	13-41
Removing Obsolete Replication Log Files	13-42
What to Do When Replication Fails	13-43

Chapter 14 Monitoring System Activity

Listing Active UniVerse Processes and Jobs	14-3
Listing UniVerse Jobs with PORT.STATUS	14-6
Terminating a Process	14-7
Examining Shared Memory	14-8
Semaphore Table	14-9
File Lock Table	14-10
Group Lock Table	14-11
Update Record Lock Table	14-11
User Process Control Locks	14-11
Dynamic File Table	14-12
UniVerse Configuration	14-12
General System Information.	14-12
Catalog Shared Memory	14-13
Printer Memory Segment.	14-13
Examining Disk Usage on UNIX Systems	14-15
Monitoring Disk Usage on UNIX Systems	14-15
System Files that Grow	14-17
Monitoring Response Time on UNIX Systems.	14-18
Keeping Directory Files Small	14-18
Running Programs During Off-Hours	14-19
Monitoring Errors on UNIX Systems	14-20

Chapter 15 UniVerse File Utilities

Administering UniVerse Files	15-4
Listing Files in a UniVerse Account	15-5
View File Properties	15-6
View File Statistics.	15-11
Running File Diagnostics.	15-13
Repairing Damaged Files.	15-17

The Format Conversion Utility	15-21
Converting the Format of Data Files and UniVerse BASIC Code . . .	15-21
The <i>uvfixfile</i> Utility	15-28
Verifying File Integrity	15-28
Fixing a Corrupt Hashed File	15-29
Examining File Statistics	15-30
Using Interactive Mode	15-30

Chapter 16 Executing UniVerse Commands

Executing a Command	16-3
The UniVerse Command Output Window	16-6
Using the Command History	16-8
Reexecuting Commands	16-8
Editing a Command	16-8
Saving Commands	16-9

Chapter 17 Sending Messages to Users

Sending Messages with UniAdmin	17-4
Sending Messages on UNIX Systems	17-4
Sending Message on Windows Platforms.	17-5
The UNIX <i>write</i> Command.	17-6
The MESSAGE Command.	17-7
Message of the Day on UNIX Systems	17-8

Chapter 18 Adding Capabilities to UniVerse

Adding UniVerse BASIC Applications	18-3
Managing Catalog Space	18-5
Initializing System Catalog Space	18-5
Checking the Status of the Catalog.	18-6
Displaying Catalog Contents	18-6
Deleting Programs from the Catalog	18-7
Managing Catalog Shared Memory	18-9
Setting Up Catalog Shared Memory	18-9
Defining Programs to Run in Shared Memory	18-11
Adding Programs to the SHM.TO.LOAD File	18-12
Removing a Program from the SHM.TO.LOAD File	18-12
Loading Programs into Catalog Shared Memory	18-13
Using Programs Stored in Catalog Shared Memory	18-15
Modifying Programs in Catalog Shared Memory	18-16
Updating a Program in Shared Memory	18-16

Removing a Program from Shared Memory	18-17
Removing the Catalog Shared Memory Segment	18-18
Adding Commands to the VOC File	18-19

Chapter 19 Managing Network Services

Administering the UniRPC on UNIX Systems.	19-4
How the UniRPC Works	19-4
System Requirements	19-4
Defining the UniRPC Port Number and Maintaining the <i>hosts</i> File.	19-5
Starting and Stopping the UniRPC Daemon	19-8
About the <i>unirpcservices</i> File	19-9
Managing Windows Telnet Sessions	19-11
Modifying the Telnet Session Parameters.	19-12
Administering Users	19-15
Adding a New User	19-16
Starting Services on Windows Platforms	19-19

Chapter 20 Device Licensing

UniVerse Licensing Modes	20-3
Why Do I Need Device Licensing?	20-5
Device Licensing Requirements	20-5
Connection Types.	20-6
Direct Connections.	20-6
Two-Tier Connections.	20-6
Enabling Telnet Device Licensing on UNIX Servers	20-7
Using Device Subkeys	20-7
Using the License Tool <i>uvlictool</i>	20-8

Appendix A UniVerse System Administration Menus

Overview of Menus and Data Entry Screens	A-2
Moving Around the Menus	A-3
Summary of Standard Keys	A-4
The UniVerse System Administration Menu	A-6
Invoking the System Administration Menu	A-6
Package Option	A-7
Installing and Deinstalling a Software Package	A-7
Administering the UniRPC	A-8
UniVerse License Administration	A-12
Administering the Deadlock Daemon	A-12
Administering SQL Client and GCI	A-15

Accounts Option	A-16
Using the Accounts Menu	A-16
Maintaining Users and User Groups	A-17
Adding, Changing, and Deleting Individual Users	A-19
Maintaining UniVerse Accounts	A-22
Recovery Option	A-27
Backing Up and Restoring Files	A-27
Using the UVBACKUP Screen	A-30
Using the UVRESTORE Screen	A-35
Transaction Logging	A-44
Spooler Option	A-55
Spooler Status Report	A-55
Managing Print Jobs	A-57
Managing the Spooler	A-59
Defining Printers	A-61
Mounting a Form on a Printer	A-65
Configuring the Spooler	A-66
Changing the Spooler Configuration	A-67
Maintaining Printer Groups	A-69
Spooler Log Files	A-70
Shared Memory (sh Mem) Option	A-72
Designating Programs for Catalog Shared Memory	A-72
Installing Programs into Catalog Shared Memory	A-74
Modifying Catalog Shared Memory	A-76
Removing Printer Memory Segments	A-77
Import Option	A-78
Restoring Non-UniVerse Accounts from Tape	A-78
Devices Option	A-85
Updating the &DEVICE& File	A-85
Defining Printers	A-91
User Menus	A-96

Appendix B PTERM and stty Options

Appendix C terminfo Terminal Capabilities

Additional <i>terminfo</i> Entries.	C-2
<i>terminfo</i> Terminal Capabilities.	C-4
<i>terminfo</i> , <i>termcap</i> , and UniVerse	C-7

Appendix D	The Wide Zero Parameter in UniVerse	
	Number Systems	D-2
	Floating-Point Numbers.	D-3
	The UniVerse Wide Zero Feature	D-5
Appendix E	Fault Numbers and Error Codes	
	Fatal Error Codes	E-4
	Initialization Errors	E-10
Index		

Preface

This manual describes tasks specific to administering the UniVerse environment. It also describes aspects of and suggests guidelines for operating system administration that are important to the UniVerse system administrator.

This manual describes how use UniVerse System Administration menus to administer UniVerse running on UNIX systems.¹ We recommend you use UniAdmin to administer UniVerse systems running on both UNIX and Windows platforms. For information about using UniAdmin, see *Using UniAdmin*.

This manual assumes you know the structure of UniVerse. Use this manual with the administrator's guide for your operating system, and *UniVerse System Description*. These documents contain many details either not covered in this manual or mentioned only in passing. System administration for specific UniVerse applications is described in documentation supplied with those applications.

Read this entire book before you try to modify the system in any way.

For a complete information about UniVerse Commands, see the *UniVerse User Reference*.

For complete information about UNIX commands, see the *UNIX Programmer's Manual* shipped with your system. The UNIX commands described in this manual appear with a reference to the section of the standard *UNIX Programmer's Manual* where the command is defined. For example, the UNIX command *man* appears as *man(1)*, indicating that the *man* command is in Section 1 of the *UNIX Programmer's Manual*.

All examples of UNIX commands in this manual assume that you are running in the Bourne shell environment with the command interpreter */bin/sh*.

1. The UniVerse System Administration menus are not available on Windows platforms.

Organization of This Manual

This manual contains the following:

- Chapter 1, [“Introduction,”](#) gives an overview of the system administrator’s job, as well as a brief behind-the-scenes look at the organization of the system software.
- Chapter 2, [“UniAdmin,”](#) describes UniVerse Admin, a Windows program that lets you administer UniVerse running on a UNIX or a Windows NT server.
- Chapter 3, [“System Startup and Shutdown,”](#) describes how to start up and shut down UniVerse.
- Chapter 4, [“Configurable UniVerse Parameters,”](#) describes the UniVerse configurable parameters you can modify to tune the performance of UniVerse.
- Chapter 5, [“Adding and Maintaining UNIX User Accounts,”](#) describes how to add new user login accounts to UNIX systems.
- Chapter 6, [“UniVerse Accounts,”](#) describes how to add new UniVerse accounts and maintain existing UniVerse accounts.
- Chapter 7, [“Transferring Accounts,”](#) explains how to transfer accounts to UniVerse from a Pick ACCOUNT-SAVE tape or a Prime INFORMATION MAGSAV tape. It also explains how to transfer UniVerse accounts from UNIX to Windows platforms.
- Chapter 8, [“UNIX System Security,”](#) describes the basic UNIX security mechanisms. It also describes special considerations in setting up secure UniVerse accounts.
- Chapter 9, [“Managing Locks,”](#) describes how to view the UniVerse lock table, and how to manage record locks, group locks, locks held by a specific user, and deadlocks.
- Chapter 10, [“Configuring Peripheral Devices,”](#) describes the basic mechanisms by which UniVerse supports peripheral devices such as terminals and tape drives. It is included to help you support additional terminals and tape drives on your system.
- Chapter 11, [“Administering Printers and the UniVerse Spooler,”](#) describes how to configure and manage printers and the UniVerse spooler on UNIX systems. It also describes how to manipulate and control print jobs and the spooler queue. See the end of the chapter for troubleshooting information.

- Chapter 12, “[Backing Up and Restoring Files](#),” describes how to back up and restore the system. Various backup and restoration methods are covered: backing up the entire system, backing up incrementally, and backing up a single account or a single file.
- Chapter 13, “[Managing Data Replication](#),” describes how to publish UniVerse files for replication, subscribe to publications, and administer hot standby systems.
- Chapter 14, “[Monitoring System Activity](#),” describes tasks that should be performed on a regular basis, such as monitoring disk use and response time, getting information about UniVerse processes and jobs, and how to terminate UniVerse processes. Also included are descriptions of the regular maintenance tasks that should be performed on UniVerse accounts, including monitoring file size and hashing efficiency, and resizing files.
- Chapter 15, “[UniVerse File Utilities](#),” describes how to use three file maintenance utilities: the UniVerse Admin FileTool, the file format conversion utility, and the *uvfixfile* utility.
- Chapter 16, “[Executing UniVerse Commands](#),” describes how to issue a UniVerse command from UniAdmin, and how to save commands to the VOC file.
- Chapter 17, “[Sending Messages to Users](#),” describes four ways to send messages to users.
- Chapter 18, “[Adding Capabilities to UniVerse](#),” describes how to make additional programs available to UniVerse users. Programs written in UniVerse BASIC, standard UNIX programs, or additional third-party applications can be added to users’ VOC files so that they can be invoked from UniVerse. This chapter also describes how to manage catalog space used for UniVerse BASIC programs.
- Chapter 19, “[Managing Network Services](#),” describes how to manage the network services used by UniVerse. It includes how to administer the remote procedure call utility (UniRPC) on UNIX systems, and how to manage Windows telnet sessions.
- Chapter 20, “[Device Licensing](#),” describes UniVerse’s device licensing system.
- Appendix A, “[UniVerse System Administration Menus](#),” describes the structure and content of the System Administration menu system on UNIX systems. The System Administration menus are displayed when the system administrator invokes the UniVerse environment from the UV account directory or logs in to the UV account.

- Appendix B, “[PTERM and stty Options](#),” describes the terminal characteristics that can be set or displayed with the UniVerse PTERM command or with the UNIX *stty* command.
- Appendix C, “[terminfo Terminal Capabilities](#),” contains a list of terminals that UniVerse adds to *terminfo.src*, descriptions of the four capability types, and a table of *terminfo* variables with their associated names in *termcap*.
- Appendix D, “[The Wide Zero Parameter in UniVerse](#),” describes floating-point numbers and explains why UniVerse has a user-configurable wide zero.
- Appendix E, “[Fault Numbers and Error Codes](#),” lists fault numbers and error codes that are displayed when UniVerse detects certain error conditions.

Documentation Conventions

This manual uses the following conventions:

Convention	Usage
Bold	In syntax, bold indicates commands, function names, and options. In text, bold indicates keys to press, function names, menu selections, and MS-DOS commands.
UPPERCASE	In syntax, uppercase indicates UniVerse commands, keywords, and options; BASIC statements and functions; and SQL statements and keywords. In text, uppercase also indicates UniVerse identifiers such as file names, account names, schema names, and Windows file names and paths.
<i>Italic</i>	In syntax, italic indicates information that you supply. In text, italic also indicates UNIX commands and options, file names, and paths.
Courier	Courier indicates examples of source code and system output.
Courier Bold	In examples, courier bold indicates characters that the user types or keys the user presses (for example, <Return>).
[]	Brackets enclose optional items. Do not type the brackets unless indicated.
{ }	Braces enclose nonoptional items from which you must select at least one. Do not type the braces.
itemA itemB	A vertical bar separating items indicates that you can choose only one item. Do not type the vertical bar.
...	Three periods indicate that more of the same type of item can optionally follow.
ä	A right arrow between menu options indicates you should choose each option in sequence. For example, “Choose File ä Exit ” means you should choose File from the menu bar, then choose Exit from the File pull-down menu.
␣	Item mark. For example, the item mark (␣) in the following string delimits elements 1 and 2, and elements 3 and 4: 1␣2F3␣4V5

Manual Conventions

Convention	Usage
F	Field mark. For example, the field mark (F) in the following string delimits elements FLD1 and VAL1: FLD1 F VAL1 V SUBV1 S SUBV2
V	Value mark. For example, the value mark (V) in the following string delimits elements VAL1 and SUBV1: FLD1 F VAL1 V SUBV1 S SUBV2
S	Subvalue mark. For example, the subvalue mark (S) in the following string delimits elements SUBV1 and SUBV2: FLD1 F VAL1 V SUBV1 S SUBV2
T	Text mark. For example, the text mark (T) in the following string delimits elements 4 and 5: 1 F 2 S 3 V 4 T 5

Manual Conventions (Continued)

The following conventions are also used:

- Syntax definitions and examples are indented for ease in reading.
- All punctuation marks included in the syntax—for example, commas, parentheses, or quotation marks—are required unless otherwise indicated.
- Syntax lines that do not fit on one line in this manual are continued on subsequent lines. The continuation lines are indented. When entering syntax, type the entire syntax entry, including the continuation lines, on the same input line.

UniVerse Documentation

UniVerse documentation includes the following:

UniVerse Installation Guide: Contains instructions for installing UniVerse 10.2.

UniVerse New Features Version 10.2: Describes enhancements and changes made in the UniVerse 10.1 release for all UniVerse products.

UniVerse Basic: Contains comprehensive information about the UniVerse BASIC language. It is for experienced programmers.

UniVerse BASIC Commands Reference: Provides syntax, descriptions, and examples of all UniVerse BASIC commands and functions.

UniVerse BASIC Extensions: Describes the following extensions to UniVerse BASIC: UniVerse BASIC Socket API, Using CallHTTP, and Using WebSphere MQ with UniVerse.

UniVerse BASIC SQL Client Interface Guide: Describes how to use the BASIC SQL Client Interface (BCI), an interface to UniVerse and non-UniVerse databases from UniVerse BASIC. The BASIC SQL Client Interface uses ODBC-like function calls to execute SQL statements on local or remote database servers such as UniVerse, DB2, SYBASE, or INFORMIX. This book is for experienced SQL programmers.

Administering UniVerse: Describes tasks performed by UniVerse administrators, such as starting up and shutting down the system, system configuration and maintenance, system security, maintaining and transferring UniVerse accounts, maintaining peripherals, backing up and restoring files, and managing file and record locks, and network services. This book includes descriptions of how to use the UniAdmin program on a Windows client and how to use shell commands on UNIX systems to administer UniVerse.

Using UniAdmin: Describes the UniAdmin tool, which enables you to configure UniVerse, configure and manage servers and databases, and monitor UniVerse performance and locks.

UniVerse Transaction Logging and Recovery: Describes the UniVerse transaction logging subsystem, including both transaction and warmstart logging and recovery. This book is for system administrators.

UniVerse Security Features: Describes security features in UniVerse, including configuring SSL through UniAdmin, using SSL with the CallHttp and Socket interfaces, using SSL with UniObjects, UniObjects for Java, and UniObjects for .NET, and automatic data encryption.

UniVerse System Description: Provides detailed and advanced information about UniVerse features and capabilities for experienced users. This book describes how to use UniVerse commands, work in a UniVerse environment, create a UniVerse database, and maintain UniVerse files.

UniVerse User Reference: Contains reference pages for all UniVerse commands, keywords, and user records, allowing experienced users to refer to syntax details quickly.

Guide to Retrieve: Describes Retrieve, the UniVerse query language that lets users select, sort, process, and display data in UniVerse files. This book is for users who are familiar with UniVerse.

Guide to ProVerb: Describes ProVerb, a UniVerse processor used by application developers to execute prestored procedures called procs. This book describes tasks such as relational data testing, arithmetic processing, and transfers to subroutines. It also includes reference pages for all ProVerb commands.

Guide to the UniVerse Editor: Describes in detail how to use the Editor, allowing users to modify UniVerse files or programs. This book also includes reference pages for all UniVerse Editor commands.

UniVerse NLS Guide: Describes how to use and manage UniVerse's National Language Support (NLS). This book is for users, programmers, and administrators.

UniVerse SQL Administration for DBAs: Describes administrative tasks typically performed by DBAs, such as maintaining database integrity and security, and creating and modifying databases. This book is for database administrators (DBAs) who are familiar with UniVerse.

UniVerse SQL User Guide: Describes how to use SQL functionality in UniVerse applications. This book is for application developers who are familiar with UniVerse.

UniVerse SQL Reference: Contains reference pages for all SQL statements and keywords, allowing experienced SQL users to refer to syntax details quickly. It includes the complete UniVerse SQL grammar in Backus Naur Form (BNF).

Related Documentation

The following documentation is also available:

UniVerse GCI Guide: Describes how to use the General Calling Interface (GCI) to call subroutines written in C, C++, or FORTRAN from BASIC programs. This book is for experienced programmers who are familiar with UniVerse.

UniVerse ODBC Guide: Describes how to install and configure a UniVerse ODBC server on a UniVerse host system. It also describes how to use UniVerse ODBC Config and how to install, configure, and use UniVerse ODBC drivers on client systems. This book is for experienced UniVerse developers who are familiar with SQL and ODBC.

UV/NET II Guide: Describes UV/Net II, the UniVerse transparent database networking facility that lets users access UniVerse files on remote systems. This book is for experienced UniVerse administrators.

UniVerse Guide for Pick Users: Describes UniVerse for new UniVerse users familiar with Pick-based systems.

Moving to UniVerse from PI/open: Describes how to prepare the PI/open environment before converting PI/open applications to run under UniVerse. This book includes step-by-step procedures for converting INFO/BASIC programs, accounts, and files. This book is for experienced PI/open users and does not assume detailed knowledge of UniVerse.

API Documentation

The following books document application programming interfaces (APIs) used for developing client applications that connect to UniVerse and UniData servers.

Administrative Supplement for Client APIs: Introduces IBM's seven common APIs, and provides important information that developers using any of the common APIs will need. It includes information about the UniRPC, the UCI Config Editor, the *ud_database* file, and device licensing.

UCI Developer's Guide: Describes how to use UCI (Uni Call Interface), an interface to UniVerse and UniData databases from C-based client programs. UCI uses ODBC-like function calls to execute SQL statements on local or remote UniVerse and UniData servers. This book is for experienced SQL programmers.

IBM JDBC Driver for UniData and UniVerse: Describes UniJDBC, an interface to UniData and UniVerse databases from JDBC applications. This book is for experienced programmers and application developers who are familiar with UniData and UniVerse, Java, JDBC, and who want to write JDBC applications that access these databases.

InterCall Developer's Guide: Describes how to use the InterCall API to access data on UniVerse and UniData systems from external programs. This book is for experienced programmers who are familiar with UniVerse or UniData.

UniObjects Developer's Guide: Describes UniObjects, an interface to UniVerse and UniData systems from Visual Basic. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with Visual Basic, and who want to write Visual Basic programs that access these databases.

UniObjects for Java Developer's Guide: Describes UniObjects for Java, an interface to UniVerse and UniData systems from Java. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with Java, and who want to write Java programs that access these databases.

UniObjects for .NET Developer's Guide: Describes UniObjects, an interface to UniVerse and UniData systems from .NET. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with .NET, and who want to write .NET programs that access these databases.

Using UniOLEDB: Describes how to use UniOLEDB, an interface to UniVerse and UniData systems for OLE DB consumers. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with OLE DB, and who want to write OLE DB programs that access these databases.

Introduction

Introduction	1-2
What Is UniVerse Administration?	1-3
Who Is a UniVerse Administrator?	1-4
UniVerse Files	1-5
Nonhashed Files	1-5
Hashed Files	1-5
B-Tree Files	1-6
UV Account Directory Files	1-6
UniVerse Administration Commands	1-8
Assigning and Unassigning Peripheral Devices	1-8
UniVerse BASIC Programs	1-8
File and Account Maintenance	1-9
Managing Processes	1-11
Monitoring Resource	1-12
Task and Record Locking	1-13
Managing Transaction Logging	1-13
National Language Support	1-14

Introduction

This chapter describes the role of a UniVerse system administrator and gives a brief overview of the relationship between the operating system and UniVerse. UniVerse administrators must know the operating system, UniVerse, and the applications developed in UniVerse.

- UniVerse is a database management environment that runs on the Windows platforms and UNIX operating systems. Most end users interact with commercial applications developed in this environment. However, many users also interact directly with the UniVerse environment and use its commands to store and retrieve data in UniVerse tables and files. Some users, and most application developers, also work in the operating system environment.

What Is UniVerse Administration?

The UniVerse administrator is responsible for the UniVerse system and for performing any tasks needed to maintain it. These tasks include the following:

- Starting up and shutting down the system
- Creating new UniVerse accounts
- Transferring accounts to UniVerse
- Implementing system security
- Configuring peripheral devices
- Rehashing and resizing UniVerse files for better performance
- Backing up and restoring files
- Managing network services
- Loading shared memory and defining the programs to run in it
- Configuring and managing printers
- Configuring and managing the UniVerse spooler (on a UNIX system)
- Managing locks

UniVerse administrators can also do the following:

- Configure and manage transaction logging
- Administer data sources

For information about administering the UniVerse Transaction Logging System, see *UniVerse Transaction Logging and Recovery*. For information about administering data sources, see *UniVerse BASIC SQL Client Interface Guide*, *UCI Developer's Guide*, and *UniVerse ODBC Guide*.

Who Is a UniVerse Administrator?

To perform most administration tasks, you must be logged on as a UniVerse Administrator. On UNIX systems, UniVerse administrators must log on as *root* or as *uvadm*. On Windows systems, UniVerse administrators must log on as a member of the Administrators group. You can then use UniVerse Admin and UniVerse administration commands to perform UniVerse administration.

UniVerse Files

UniVerse files are described in detail in *UniVerse System Description*. This section summarizes the main points of the UniVerse file system.

UniVerse provides the following kinds of file organization:

- Nonhashed files
- Hashed files
- B-tree files

Nonhashed Files

Nonhashed files store text, program source code, and other data that does not have much structure to it. A nonhashed file is implemented as an operating system directory. The records in a nonhashed file are operating system files.

Hashed Files

Hashed files use a hashing algorithm to distribute records in one or more groups of the file. The algorithm is applied to the record ID to generate the address of a group buffer where the record is stored. To find a record in a hashed file, UniVerse locates its group address on the disk. Within that group, individual record IDs are examined to identify the record.

Hashed files allow rapid access to records regardless of the number of records in the file. There are two kinds of hashed files: static and dynamic. A static file does not change its size to accommodate changes in the amount of data stored in it. A dynamic file resizes itself by changing the number of groups required to store records.

B-Tree Files

B-tree files store data for rapid access. In a B-tree file, records are stored in sorted order. To find a record, its record ID is compared to the value at the center of the tree. If the value is greater than the record ID, the search continues with the subtree to the left of the center value. If it is less, the search continues with the subtree to the right of the center value. This process continues until the record is found.

A B-tree file is the most efficient file structure to use when frequent searches are made for partially specified keys, such as a key that comprises a region code and an invoice number. A search for the last 100 orders placed in the “NE” region would search only the branches with a key beginning with “NE”. Using partially specified keys to find records in a hashed file is inefficient because it involves reading the entire file.

UV Account Directory Files

During installation, the UniVerse creates a master account in the UV account directory. The following files and directories are unique to the UV account directory:

Name	Description
bin	Directory containing UniVerse system programs.
catdir	Directory containing cataloged UniVerse programs. Also referred to as the system catalog space, or (more simply) the catalog.
errlog	On UNIX systems, an error-logging file containing the most recently logged errors.
gcidir	Directory containing General Calling Interface (GCI) files.
nls	Directory containing UniVerse NLS files.
sample	Directory containing sample programs, demonstration files, and other sample UniVerse files.
sql/catalog	Directory containing the UniVerse SQL catalog.
sqlclient.config	File containing definitions of BASIC SQL Client Interface data sources.
terminfo	Directory containing terminal definitions.
uvconfig	File containing UniVerse configurable parameter values.
uvdr.config	File containing the configuration of the data replication utility.

UV Account Directory Files

The following UniVerse system files are unique to the UV account:

Name	Description
&DEVICE&	File containing definitions and configurations for all peripheral devices.
APP.PROGS.O	File containing optional UniVerse application programs.
BP	File containing system BASIC programs.
B.P.O	File containing the executable code for system BASIC programs.
CAT	File containing lists of globally cataloged programs.
DICT.DICT	File containing the dictionary definitions for all file dictionaries on the system.
NEWACC	File containing master copies of the default VOC files that are copied into new UniVerse accounts.
PTERM.FILE	File containing tables used by the PTERM command for setting terminal characteristics.
SYS.MESSAGE	Master file containing system messages and prompts used in the UniVerse environment. You can edit this file to change the prompts.
UV.ACCOUNT	File containing the names and paths of UniVerse accounts.
UV.FLAVOR	File specifying the flavors of UniVerse accounts.

UniVerse System Files

UniVerse Administration Commands

This section describes UniVerse commands that administrators will find useful. Many of them can be used only by a user logged on to the UV account as a UniVerse Administrator. UniAdmin is an interface to some of the commands listed here. For ease of reference, the commands are organized in functional groups and are listed alphabetically.

Assigning and Unassigning Peripheral Devices

The following table describes commands for assigning and unassigning peripheral devices.

Command	Description
ASSIGN	Assigns a device for your exclusive use. Usually used for assigning tape, printer, and disk devices.
UNASSIGN	Relinquishes control of a physical device that has been assigned to you.

Peripheral Device Commands

UniVerse BASIC Programs

The next table describes commands for UniVerse BASIC programs.

Command	Description
BASIC	Compiles a UniVerse BASIC program.
CATALOG	Copies the compiled object code to the system catalog space.
DELETE.CATALOG	Deletes programs from the catalog space.
ED	Creates UniVerse BASIC source programs. Also edits the contents of data files, file dictionaries, and select lists.
INITIALIZE.CATALOG	Initializes the catalog space.

BASIC Program Commands

Command	Description
LIMIT	Sets the maximum size of memory storage for a user's active UniVerse BASIC routines.
MAKE.MAP.FILE	Creates the &MAP& file from the catalog contents.
MAP	Displays information about the contents of the catalog space.
RAID	Debugs a UniVerse BASIC program.
VCATALOG	Compares the object code of a program in the catalog to object code in the original file.
VLIST	Lists UniVerse BASIC object code.

BASIC Program Commands (Continued)

File and Account Maintenance

The following table describes commands for file and account maintenance.

Command	Description
ACCOUNT.FILE.STATS	Gathers file statistics on the current state of selected files.
ANALYZE.FILE	Displays statistics about a dynamic file.
CLEAN.ACCOUNT	Performs routine maintenance and verifies the condition of files in an account.
CONFIGURE.FILE	Changes the parameters of a dynamic file.
FILE.STAT	Displays statistical information about the file size and record partitioning in a file.
FILE.USAGE	Displays statistics on the use patterns of a file.
FILE.USAGE.CLEAR	Resets statistics displayed by the FILE.USAGE command.
FORMAT.CONV	Changes the storage format of UniVerse files or BASIC object code.
GROUP.STAT	Displays information about the record distribution in a file.

File and Account Maintenance Commands

Command	Description
GROUP.STAT.DETAIL	Displays a detailed record distribution summary for a file.
HASH.AID	Displays statistical information about the hypothetical file size and record partitioning in the file.
HASH.HELP	Displays a recommendation for a file type, modulo, and separation based on the current record IDs and file size.
HASH.HELP.DETAIL	Displays the same information as HASH.HELP, and also includes details on the record ID sizes and record size in bytes.
HASH.TEST	Displays how the record distribution would work with a hypothetical file type, modulo, and separation.
HASH.TEST.DETAIL	Displays the same hypothetical information as HASH.TEST, and also includes the hypothetical number of bytes per record, and the number of bytes per group.
LIST.FILE.STATS	Displays file statistics gathered by ACCOUNT.FILE.STATS.
RECORD	Determines the group that a record should reside in, or, if the record exists, verifies that the record is there.
RESIZE	Changes the structure of a file with a new file type, modulo, or separation.
SUSPEND.FILES	Suspends UniVerse processes that make changes to files, without terminating user processes.
VVOC	Compares the contents of the VOC file in the current account to those of NEWACC, and reports the differences.

File and Account Maintenance Commands (Continued)

Managing Processes

The following table describes commands for managing processes.

Command	Description
AUTOLOGOUT	Logs the user out of UniVerse after a period of inactivity.
CHAP	On UNIX systems, changes the execution priority level for tasks.
ENVIRONMENT or ENV	Sets and displays environment variables.
JOBS	Lists active phantom processes.
MAIL	On UNIX systems, lets you send and receive messages from other users on the system.
MESSAGE	Sends a message from your terminal to another user.
PASSWD	On UNIX systems, sets or changes the password for the account you are using.
PHANTOM	Starts a phantom process.
SLEEP	Suspends a process.
UMASK	On UNIX systems, sets default file permission modes for an account.
Commands for Managing Processes	

Monitoring Resource

The following table describes commands for monitoring resources.

Command	Description
ANALYZE.SHM	Displays statistics about the disk and printer shared memory segments.
AVAIL	Displays statistics about the disk records.
CONFIG	Displays information about current authorization parameters and configurable parameter values.
CORE	On UNIX systems, displays statistics about UniVerse's current memory usage.
LISTU	Displays information about the users currently on the system.
PORT.STATUS	Displays information about UniVerse processes and jobs currently running on the system.
STATUS	Displays information about the files that are open, the network, assigned devices, the operating system version, and the users.
TANDEM	On UNIX systems, displays input and output of another user's terminal.
USERS	Displays the number of users on the system.
Commands for Monitoring Resources	

Task and Record Locking

The following table describes commands for task and record locking.

Command	Description
CLEAR.LOCKS	Clears a specific lock number or all the locks set by an account.
LIST.LOCKS	Lists the 64 task synchronization locks, informing you which are still available.
LIST.READU	Displays a list of locked files and records.
LOCK	Reserves one of the 64 task synchronization locks to guarantee that you can process a file record without interference from others.
MASTER	Releases task synchronization locks set with the LOCK command.
RELEASE	Releases record locks that were set by the UniVerse BASIC commands READU, MATREADU, and READVU.
SEMAPHORE.STATUS	Displays information about the status of system semaphores.
UNLOCK	Clears file, group, and update locks.

Commands for Task and Record Locking

Managing Transaction Logging

On UNIX systems, the following commands let you manage the transaction logging and recovery system.

Command	Description
ACTLIST	Activates lists of recoverable files for logging.
CREATE.LDIR	Creates the log directory.
CREATE.LFILE	Creates log files.
DEACTLIST	Deactivates lists of recoverable files for logging.

Commands for Managing Transaction Logging

Command	Description
DEL.RFILE	Deletes a series of log files once they have been rolled forward.
DELETE.LFILE	Deletes empty log files from the log directory.
ENABLE.RECOVERY	Starts up the log daemon.
LOG.RESTORE	Restores log files from tape to a log directory on disk.
LOG.SAVE	Saves log files from a log directory on disk to tape.
MKFILELIST	Creates and saves a select list of all files in an account.
RECOVERY.CHECKPOINT	Finds the numbers of the first log file you need for a roll-forward recovery.
RECOVERY.CONSISTENT	Clears a file's inconsistency flag.
RELEASE.LFILE	Releases a full log file for reuse.
SET.LOG.ATTR	Sets the archive and checkpoint modes to ON or OFF.
SHUTDOWN.RECOVERY	Shuts down the log daemon.
SUSPEND.RECOVERY	Suspends the log daemon.

Commands for Managing Transaction Logging (Continued)

National Language Support

UniVerse has a special mode that offers National Language Support (NLS). With NLS mode enabled, you can use UniVerse in various languages and countries. You can do the following:

- Input data in any character set using your local keyboard
- Retrieve data and format it using your own conventions, or those of another country
- Output data to a screen or printer using the character sets and display conventions of any country
- Write programs that run in different languages and countries without source changes or recompilation

NLS mode works by using two types of character sets:

- The UniVerse internal character set
- External character sets that cover the world's different languages

In NLS mode, UniVerse maps between the two character sets when you input data to the database, or output data from the database.

For complete information about installing and administering NLS and using NLS commands, see the *UniVerse NLS Guide* and the *UniVerse User Reference*.

UniAdmin

Starting UniAdmin 2-4



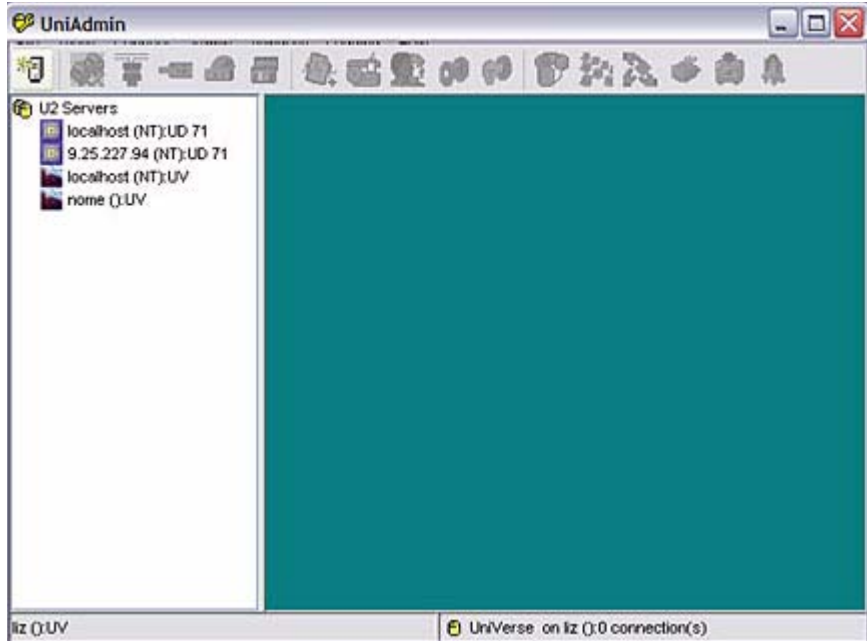
UniAdmin

UniAdmin is a Windows-based program that lets you administer UniVerse running on a UNIX or Windows server.

Note: To use *UniAdmin*, you must be logged on as a *UniVerse Administrator*.

Starting UniAdmin

To display the UniAdmin main window, choose **Start -> Programs -> IBM U2 -> UniAdmin**. A window similar to the following example appears:



UniAdmin enables you to perform the following tasks:

- Administer Accounts
 - Create a new account
 - View details of an existing account
 - Delete an Account
- Create Backups
 - Backup UniVerse accounts
- Change UniVerse configuration parameters

- Administer UCI or UniVerse ODBC data sources
 - Create a data source
 - Delete an existing data source
 - View or modify information about a data source
- Administer UniVerse deadlocks
 - Start or stop the UniVerse Deadlock Manager
 - Configure the Deadlock Manager
 - Manually resolve file locking conflicts
- Administer Devices
 - Configure tapes
 - Configure printer
 - Configure other device
- View file system usage
 - Define and update log configuration table
 - Define and update archive configuration table
- File Tools
- View files in UniVerse accounts
- Import
 - Import non-UniVerse accounts from Prime Information or Pick systems
 - Manually import non-UniVerse accounts from tape
- License UniVerse or UV/Net
 - View information about licenses in use
 - Clean UniVerse license
 - Update UniVerse license
- Administer Locks
 - Monitor locks
 - Clear file locks
 - Clear record locks
 - Clear group locks

- Administer network services
 - Administer telnet services
 - Administer other services
- Monitor UniVerse processes
 - Monitor performance
- Administer Replication
- Restore files from backup
- Manage shared programs
 - Define programs to load into shared memory
 - Modify programs in shared memory
 - Load program into catalog shared memory
 - Remove catalog shared memory segment
- Configure SSL through UniAdmin
- Administer Transaction Logging
- Execute UniVerse commands
- Administer UniVerse users
 - View user and background processes
 - Terminate a UniVerse user process
 - Send a message to a UniVerse user

For detailed information about UniAdmin, see *Using UniAdmin*.

System Startup and Shutdown

Starting Up and Shutting Down UniVerse on UNIX Systems	3-3
Configuring UniVerse Parameters at Initialization Time	3-3
Logging On	3-3
Initializing the UniVerse Shell	3-3
Starting and Stopping UniVerse on Windows Systems	3-5
Starting UniVerse Services	3-5
Stopping UniVerse Services	3-6
Overview of UNIX Startup and Shutdown	3-9
UNIX Startup	3-9
UNIX Shutdown	3-10

This chapter describes how to start up and shut down UniVerse.

- The first part of the chapter describes how to start up and shut down UniVerse on a UNIX system.
- The second part of the chapter describes how to start up and stop UniVerse on a Windows system.
- The last part of the chapter gives an overview of what happens when a UNIX system is started up and shut down, so that you know what to do if you need to modify the standard procedure.

Starting Up and Shutting Down UniVerse on UNIX Systems

You start up and shut down UniVerse from a UNIX shell. The *uv* command with the *-admin* option runs the script that provides for the orderly startup and shutdown of UniVerse. The syntax is as follows:

```
uv -admin { -start | -stop }
```

Use the *uv -admin* command with the *-start* option to start up UniVerse. Use *uv -admin* with the *-stop* option to shut down UniVerse.

Configuring UniVerse Parameters at Initialization Time

You can configure certain UniVerse parameters at initialization time to provide some flexibility in tuning the performance of UniVerse. See Chapter 4, [“Configurable UniVerse Parameters,”](#) for more information.

Logging On

Once you log on using a particular name, certain account parameters are automatically set for you. These parameters include a home directory (the starting point of your personal file hierarchy), the type of command interpreter that will be used to issue system commands (a UNIX shell or the UniVerse command processor), file protection, and so forth.

Initializing the UniVerse Shell

If you specify the UniVerse command processor as the account’s command interpreter, UniVerse executes the UV.LOGIN entry in the VOC file of the UV account. You can set system-wide defaults by putting appropriate commands in the UV.LOGIN entry.

After executing UV.LOGIN, UniVerse executes the LOGIN entry in the VOC file of the user’s account. LOGIN can be a paragraph, sentence, proc, menu, or UniVerse BASIC program.

In a PICK, IN2, or REALITY flavor account, the UniVerse command processor first executes an entry with the account name. If none exists, it executes an entry with the login name. If none exists, it executes an entry named LOGIN.

Starting and Stopping UniVerse on Windows Systems

You must be a domain administrator or a UniVerse Administrator to start up or shut down UniVerse on a Windows system. You do so by starting or shutting down the UniVerse Resource service. In addition to the UniVerse Resource service, you may also want to start up or shut down two other services:

- UniVerse Telnet service
- UniRPC service

Starting UniVerse Services

You can start UniVerse in one of three ways:

- From the UniVerse Control Panel
- From the Windows Control Panel
- At the MS-DOS prompt

Use the UniVerse Control Panel when you want to start all four UniVerse services together. Use either the Windows Control Panel or an MS-DOS prompt when you want to start one or more of the UniVerse services individually.

Normally you should start both the UniVerse Resource service and the UniVerse Telnet service. If you want UniVerse to accept connections from client programs or through UV/Net, you must start the UniRPC service.

If you start the UniVerse Telnet service, the UniVerse Resource service is automatically started if it is not already running.

From the UniVerse Control Panel:

1. Choose **Start ä Programs -> IBM U2 -> UniVerse -> UniVerse Control**.
2. Click **Start All Services** to start all UniVerse services.

From the Windows Control Panel:

1. Double-click the **Services** icon.

2. Scroll down the list of services until you find three entries for UniVerse:

- UniRPC Service
- UniVerse Resource Service
- UniVerse Telnet Service.

3. Choose **UniVerse Resource Service**, then choose **Start**.

4. Choose **Startup**, then choose **automatic**. This ensures that UniVerse is started automatically when the server is rebooted.

5. Repeat the previous step to start any of the other UniVerse services.

At the MS-DOS Prompt:

Enter one or more of the following commands:

```
D:\users>net start universe
```

```
D:\users>net start uvtelnet
```

```
D:\users>net start unirpc
```

```
D:\users>net start hsrexec
```

The system reports the name of the service it is starting and whether the startup is successful.



Note: *The UniVerse services are started automatically when the operating system is loaded unless you clear the automatic startup boxes during UniVerse installation.*

Stopping UniVerse Services

You can shut down UniVerse in one of three ways:

- From the UniVerse Control Panel
- From the Windows Control Panel
- At the MS-DOS prompt



Note: *If users are connected to the services when they are shut down, the users do not lose their connections; the connections remain active until the users terminate them. However, it is not possible for new users to connect to UniVerse.*

If you want to do a complete shutdown of UniVerse to restart the services, be sure you first terminate all connections.

From the UniVerse Control Panel:

1. Choose **Start -> Programs -> IBM U2 -> UniVerse -> UniVerse Control**.
2. Click **Stop All Services** to stop all UniVerse services. Wait for all services to stop.
3. Click **OK** to exit the UniVerse Control Panel. All four services are shut down.

From the Windows Control Panel:

1. Double-click the **Services** icon.
2. Scroll down the list of services until you find three entries for UniVerse:
 - UniRPC Service
 - UniVerse Resource Service
 - UniVerse Telnet Service
3. Choose **UniVerse Resource Service**, then choose **Stop**.

If the UniVerse Telnet service is also running, a message appears prompting you to confirm that in addition to the UniVerse Resource Service, you also want to stop the UniVerse Telnet Service.

4. Click **OK**.

At the MS-DOS Prompt:

1. Enter one or more of the following commands at the MS-DOS prompt:

```
D:\users>net stop universe  
D:\users>net stop uvtelnet  
D:\users>net stop unirpc
```

If the UniVerse Telnet service is also running when you stop UniVerse, a message appears prompting you to confirm that you also want to stop the UniVerse Telnet service.

2. Enter **Y** at the prompts. Each service is shut down in turn.

Preserving Data Integrity When You Shut Down Windows Platforms

Windows platforms allow only a short time for applications to close during system shutdown. If applications do not respond quickly, the shutdown process closes the applications immediately. If your UniVerse application is writing data to disk during the shutdown, you could lose data.

This can have severe consequences for the UniVerse transaction logging service. If a shutdown request is issued during a transaction, the transaction logging service must flush the buffered transaction to disk. If the operating system shuts down before the write to disk finishes, the data integrity of the transaction is lost.

To ensure data integrity, always shut down the UniVerse Resource service before you shut down Windows. If you have a transaction running, check that the final entry in the *uvlogd.info* file in your transaction log directory reads as follows:

```
Logging system shut down consistently.
```

If you see this message, you can safely shut down Windows.

Overview of UNIX Startup and Shutdown

When you start up a UNIX system, the file systems are automatically checked, a number of automatic processes (such as error logging, process accounting, and print spooling) are initiated, and all terminals are prepared for user logins. When you shut down the system, automatic processes must be terminated, and the file systems closed down gracefully so there is no disk activity when the power is turned off. Most of the system startup and shutdown processes are done automatically by a number of shell scripts and programs shipped with the system.

You must do the startup and shutdown procedures described in this chapter from a UNIX shell.

UNIX Startup

Initializing UNIX: The /etc/init Program

The *init* program runs a series of initialization scripts that check and mount the file systems, start various accounting processes and system daemons (automatic processes such as spoolers), and continuously spawn *getty* processes. One of the initialization scripts, */etc/rc*, starts UniVerse.

The UniVerse Startup Script

The main script for the UNIX system initialization process is in the */etc/rc* file. To see the processes invoked when the system is booted to multiuser mode, you can print this file. Among other things, */etc/rc* calls other command files that vary from system to system.

The UniVerse installation procedure modifies the UNIX startup procedure to execute the commands in a file copied from */usr/ibm/uv/sample/uv.rc*. You can see where this file is on your system by entering the following command from the UV account directory:

```
# cat .uvrcloc
```

The *uv.rc* file contains the script that provides for the orderly startup and shutdown of UniVerse when UNIX changes run levels. You can run this script from the UNIX prompt either to start up or to shut down UniVerse. The syntax is as follows:

uv.rc [start | stop]

Use the *uv.rc* command with the *start* option to start up UniVerse. Use *uv.rc* with the *stop* option to shut down UniVerse. If you use *uv.rc* with no options, the startup procedure executes.

UNIX Shutdown

Use the following three steps to shut down your system:

1. Run the UniVerse shutdown script.
2. Return the system to single-user mode from multiuser mode.
3. Shut down your system, or reboot it.

Running the UniVerse Shutdown Script

You must be in the root directory in order to shut down the system. If you are not in the root directory, enter the following command to change to the root directory:

cd /

To shut down UniVerse from the root directory, use the *uv.rc* command with the *stop* option at the UNIX prompt:

uv.rc stop

The UniVerse shutdown script does the following:

- Shuts down the spooler
- Shuts down all active UniVerse processes, freeing all resources allocated to them

After this is done, the following message appears:

UV has been brought down.

You can now bring the UNIX system down to single-user mode from multiuser mode.

Returning to Single-User Mode

You execute the *shutdown* command to return the system to single-user mode. The *shutdown* command */etc/shutdown* provides an automated shutdown procedure which notifies users that the system is about to be shut down after a specified interval. After the interval has elapsed, any users still on the system are automatically logged off, and the various system daemons are terminated. From the root directory enter the following command to bring the system to single-user mode:

```
# shutdown
```

Wait until *shutdown* notifies users and finishes running. When the shutdown is completed, the system is left in single-user mode. Depending on the reason for the shutdown, you may want to perform file system maintenance, reboot, or power down the system entirely.



Warning: *You must run shutdown before turning off the power, or you risk corrupting the file system. If you are working in single-user mode and you want to turn off the power, first issue the sync command twice, as follows:*

```
# sync; sync
```

See the *UNIX Programmer's Manual* for a detailed description of the *sync* command.

Shutting Down and Rebooting a Running System

Rebooting the system is not a cure-all. Nonetheless, there are many cases in which rebooting (that is, rerunning the initialization scripts that take the system from single-user to multiuser mode) can clear various error conditions.

To notify users that you intend to reboot the system, run the */etc/shutdown* command as described in [Returning to Single-User Mode](#). Wait until *shutdown* notifies users and finishes running, then perform the reboot as directed by the UNIX manuals for your system.

Configurable UniVerse Parameters

The <i>uvconfig</i> and <i>.uvconfig</i> Files	4-2
The <i>uvregen</i> Program	4-2
Allocating Shared Memory	4-2
Recovering <i>.uvconfig</i>	4-3
The Configurable Parameters	4-4
Changing Configurable Parameter Values	4-17
Using UniAdmin to Change Parameter Values	4-18
Editing the <i>uvconfig</i> File	4-20
The Default <i>uvconfig</i> File	4-21

When UniVerse starts, configurable parameters are used to specify certain UniVerse settings and limits. This chapter describes these configurable parameters.

The *uvconfig* and *.uvconfig* Files

The current settings for the configurable parameters are stored in the *uvconfig* file, located in the UV account directory. A command called *uvregen* uses the *uvconfig* file to create another file in the UV account directory named *.uvconfig*, which contains an encrypted version of the current configurable parameter settings. The *.uvconfig* file is used during the startup of UniVerse.

The *uvregen* Program

When you run the *uvregen* program, it does the following:

- Verifies that the values in the *uvconfig* file are reasonable
- Creates a new *.uvconfig* file in the UV account directory
- Resets the master key if you are using encryption

Some *uvconfig* values may be reasonable but invalid for the current kernel configuration. *uvregen* cannot detect such inconsistencies.

Allocating Shared Memory

When you start up UniVerse, the settings in the *.uvconfig* file are used to determine how much space to allocate for disk shared memory. Since changing the configurable parameters often changes the amount of shared memory you need, you should always restart UniVerse whenever you change any of the configurable parameters.

Recovering *.uvconfig*

If something happens to the *.uvconfig* file, there is a file named *.uvconfig.bak* in the UV account directory that is a backup copy of the *.uvconfig* file as shipped with the release. Copy it to *.uvconfig* to restore a usable UniVerse environment. To restore the *uvconfig* file, use the default values for the parameters.

The Configurable Parameters

The following table lists the UniVerse configurable parameters. The default values shown may be different on your system. Please consult the *UniVerse Installation Guide* for changes to the default values.

Parameter	Description
64BIT_FILES	Specifies whether UniVerse uses 32-bit or 64-bit file systems. A value of 0 means all UniVerse files are created and resized as 32-bit files. A value of 1 means all UniVerse files are created and resized as 64-bit files. The default value is 0.
ALLOWMARKS	<p>The value of this parameter can be either 0 or 1. A value of 1 indicates to allow marks (SVM, VM, AM, or TM) in record IDS for non-SQL tables. A value of 0 indicates not to allow marks in record IDs.</p> <p>Warning: Using multivalued record IDs can cause adverse effects on the database, and may corrupt index data. Reports may print improperly and the TRANS function will not work properly.</p>
ALLOWNFS	Specifies whether UniVerse files stored on remote systems can be opened without UV/Net. A value of 0 means UV/Net is required to access remote files. Any other value allows remote files to be opened, but no locks are maintained on the remote system. Turn this parameter on when UniVerse is not running on the remote system. The default value is 0.
BGINPUTTIMEOUT	The amount of time, in seconds, a background process, including PHANTOMS, waits for input before it terminates.
BLKMAX	Sets the maximum block size for UVBACKUP and UVRESTORE. It must be greater than, and a multiple of, 512.
CENTURYPIVOT	Sets the century pivot year, which determines how 1 and 2 digit years are interpreted by the ICONV function. A value of 1930 means that 30 through 99 are interpreted as the 1900s, and 00 through 29 are interpreted as 2000 - 2029. If the century pivot value is two digits, the century pivot year is based on the current year, as follows: a value of 30 means the century pivot year is 1930 in 2000, 1931 in 2001, 1932 in 2002, and so forth. The default value is 1930.

UniVerse Configurable Parameters

Parameter	Description
CONVERT_EURO	If this value is set to 0 (the default), UniVerse does not do any Euro conversion. If this value is set to 1, UniVerse performs Euro conversion.
CPLOGRLS	The amount of time, in seconds, UniVerse waits before releasing a log file. If this value is 0 (the default), the uvchkd process calls the fsync function for all data files that have updates in the current log file. If this value is greater than 0, uvchkd does not call the fsync function and waits to release the log file for the number of seconds you specify.
CSHDISPATCH	Defines the full path name for the <i>cs</i> shell command. On UNIX systems the default is <i>/usr/bin/csh</i> . On Windows systems the default is NOT_SUPPORTED.
DOSDISPATCH	Defines the full path for the DOS shell command. On UNIX systems the default is NOT_SUPPORTED. On Windows systems the default is CMD.EXE.
EXACTNUMERIC	Specifies the number of digits of precision before rounding occurs. It can be between 15 through 57 digits. The default value is 15 digits.
FLTABSZ	Sets the number of file lock entries in a file lock semaphore set. The default value is 11.
FMT_TEXTMARK	Dictates how the UniVerse BASIC FMT function treats a string when you specify the field width and the input string is larger than that field width. <ul style="list-style-type: none"> n 1 - If the input string is larger than width n, UniVerse inserts a text mark (CHAR (251)) every n characters. UniVerse pads each field with the fill character to the width definition. n 0 - If the input string is larger than width n, UniVerse truncates the string at width n.
FSEMNUM	Sets the number of file lock semaphore sets used for concurrency control. The default value is 23.
GLTABSZ	Sets the number of group lock entries in a group lock semaphore set. The default value is 75.
GSEMNUM	Sets the number of group lock semaphore sets used for concurrency control. The default value is 97.

UniVerse Configurable Parameters (Continued)

Parameter	Description
HISTSTK	Specifies the maximum number of sentences in a user's sentence stack. The default value is 99.
ISOMODE	Sets the SQL isolation level. See <i>UniVerse BASIC</i> for an explanation of the possible values. The default value is 1.
JOINBUF	Specifies the size of the cache the optimizer uses for joins with explicit record IDs or indexes. The default value is 4095.
LAYERSEL	Determines whether a select list remains active when returning from a higher EXECUTE level. A value of 0 maintains an active select list. Any other value clears any select lists before returning to the previous layer, unless K mode is active. The default value is 0.
LOGBLNUM	Specifies the size of the log data buffer, in file system blocks. The default value is 8.
LOGBLSZ	Specifies the log buffer block size should be the same as the block size of the file system where the log directory is mounted. The default value is 512.
LOGSYCNT	Specifies the maximum number of commits allowed between log file <i>syncs</i> . If the specified value is less than 2, it is changed to 0. This parameter has no effect on a nontransactional environment. The default value is 0.
LOGSYINT	Specifies the maximum time interval allowed between log file <i>syncs</i> . The minimum value (other than 0) you can specify is 5. If the LOGSYCNT parameter is set to a value other than 0, LOGSYINT defaults to 120.
MALLOCTRACING	Turns on <i>malloc</i> tracing for UniVerse support analysis. 1 turns tracing on, 0 turns tracing off. The default value is 0.
MAXERRLOGENT	Specifies the maximum number of log entries that can be written to the <i>errlog</i> file. The default is 100.
MAXKEYSIZE	Specifies the maximum number of characters for a primary key. It must be any multiple of 64 between 256 and 2048. The full record ID is stored in the record lock entry. The default value is 255. Do not change MAXKEYSIZE from the default value without understanding its effect on the record lock table entries.

UniVerse Configurable Parameters (Continued)

Parameter	Description
MAXRLOCK	Sets the maximum number of record locks that can be held by an SQL transaction on a physical file (a device or an i-node) before a file lock is requested. The default is 74.
MFILES	Specifies the size of the UniVerse rotating file pool. The value of MFILES must be at least 8 less than the system's open files per process limit. The default value is 12. Count UV/Net connections as files. Pipes are not part of the rotating file pool, but they do count as files.
MODFPTRS	Specifies whether file pointers in the VOC file can be modified by the COPY, DELETE, and EDIT commands. A value of 0 disallows modifications of VOC file pointers. The default value is 1.
NETTIME	Sets the timeout value in minutes for UV/Net. The default value is 5.
NLSDEFDEVMAP	Specifies the name of the default map to use for device input or output. This map is used for all devices except printers that do not have a map specified in the &DEVICE& file. The ASSIGN MAP command overrides this setting. The default value is ISO8859-1+MARKS.
NLSDEFDIRMAP	Specifies the name of the default map to use for type 1 and type 19 files without assigned maps. This occurs if a type 1 or type 19 file was not created on an NLS system and has not had a map defined for it by the SET.FILE.MAP command. This map applies only to the data in records, not to record IDs. The default value is ISO8859-1+MARKS.
NLSDEFFILEMAP	Specifies the name of the default map to use for hashed files without assigned maps. This occurs if a hashed file was not created on an NLS system and has not had a map defined for it by the SET.FILE.MAP command. The default value is ISO8859-1+MARKS.
NLSDEFGCIMAP	Specifies the name of the default map to use for string arguments passed to and from GCI subroutines. This map is used if the GCI subroutine does not explicitly define a map. The default value is ISO8859-1+MARKS.

UniVerse Configurable Parameters (Continued)

Parameter	Description
NLSDEFPTRMAP	Specifies the name of the default map to use for printer output. This map is used if a printer does not have a map defined for it in the &DEVICE& file. The default value is ISO8859-1+MARKS.
NLSDEFSEQMAP	Specifies the name of the default map to use for sequential input or output for files or devices without assigned maps. The SET.SEQ.MAP command overrides this setting. The default value is ISO8859-1+MARKS.
NLSDEFSRVLC	Specifies the name of the default locale to use for passing data to and from client programs. This locale is used if the client program does not specify a server locale. The default value is ISO8859-1+MARKS.
NLSDEFSRVMAP	Specifies the name of the default map to use for passing data to and from client programs. This map is used if the client program does not specify a server map. The default value is ISO8859-1+MARKS.
NLSDEFTERMMAP	Specifies the name of the default map to use for terminal input or output. This map is used if a terminal does not have a map defined for it in its <i>terminfo</i> definition. The SET.TERM.TYPE MAP command overrides this setting. The default value is ISO8859-1+MARKS.
NLSDEFUSRLC	Specifies the default locale. The default value is OFF.
NLSLCMODE	Specifies whether locales are enabled. A value of 1 indicates that locales are enabled; a value of 0 indicates that locales are disabled. The default setting is 0. This parameter has no effect unless NLSMODE is set to 1.
NLSMODE	Turns NLS mode on or off. A value of 1 indicates NLS is on, a value of 0 indicates NLS is off. If NLS mode is off, UniVerse does not check any other NLS parameters.
NLSNEWDIRMAP	Specifies the name of the map to use for new type 1 and type 19 files created when NLS mode is on. This map applies only to the data in records, not to record IDs. The default value is ISO8859-1+MARKS.

UniVerse Configurable Parameters (Continued)

Parameter	Description
NLSNEWFILEMAP	Specifies the name of the map to use for new hashed files created when NLS mode is on. A value of NONE (the default value) indicates that data is to be held in the internal UniVerse character set.
NLSOSMAP	Specifies the name of the map to use for file names or record IDs visible to the operating system. This chiefly affects CREATE.FILE and record IDs written to type 1 or type 19 files. The default value is ISO8859-1.
NLSREADELSE	Specifies the action to take if characters cannot be mapped when a record is read by a READ statement. A value of 1 indicates that the READ statement takes the ELSE clause. A value of 0 indicates that unmappable characters are returned as the Unicode replacement character 0xFFFD. The default value is 1.
NLSDEFSOCKMAP	The name of the map to associate with sockets that are either explicitly created through UniVerse BASIC APIs, or implicitly created through other APIs, such as CallHTTP.
NLSWRITEELSE	Specifies the action to take if characters cannot be mapped when data is written to a record. A value of 1 indicates that the write aborts or takes the ON ERROR clause (if there is one). A value of 0 indicates that unmappable characters are converted to the file map's unknown character (for example, ?) before writing the record. When this happens, some data may be lost.
OCVDATE	Specifies whether UniVerse accepts partially bad internal dates. A value of 0 rejects any value that is not wholly numeric and supplied as an internal date to the D conversion code; the date is not converted. Any value other than 0 accepts a number followed by nonnumeric data (for example, 9199-f); the number is treated as an internal date, and the STATUS function is set to 3. The default value is 0.
OPENCHK	Modifies the behavior of operations on files opened with the BASIC OPEN statement. When set to 0, no integrity constraints are observed. This parameter does not affect files opened using the OPENCHECK statement. The default value is 1.
OPTMEM	Specifies the amount of memory allocated for the query optimizer's workspace. This is specified in 1K units. The default value is 64.

UniVerse Configurable Parameters (Continued)

Parameter	Description
PAKTIME	Specifies the number of seconds the system waits at the <code>Press Any Key to Continue</code> message before releasing a pending group lock. The default value is 300.
PHANTOMSAMEUV	The value of this parameter can be either 0 or 1. If you specify 1, UniVerse enables the phantom using the same uv executable. A value of 0, the default, uses UVHOME/bin/uv.
PI_MATCHFIELD	If you set the value of this parameter to 1, the UniVerse BASIC MATCHFIELD function performs as it does in PI/open.
PICK_MT	The value of this parameter can be either 0 or 1. 1 indicates the MT conversion works the same as the Pick flavor in INFORMATION. A value of 0 indicates UniVerse MT conversion handling.
PICKDATE	The value of this parameter can be either 0 or 1. A value of 1 indicates that UniVerse will accept Pick-style dates, and handle data in the same way as Pick. A value of 0 indicates the default UniVerse date format of <code>yyyymmdd</code> .
PICKNULL	Sets the masked decimal conversion for empty data. A value of 1 turns on Pick-style conversions, where empty data is converted to an empty string. A value of 0 indicates UniVerse-style conversions, where empty data is converted to 0.00.
PIOPENDEFAULT	Sets the INFO.CONVERT and PIOPEN.EXECUTE options of the BASIC \$OPTIONS statement as defaults in PIOPEN flavor accounts.
PKRJUST	A value of 1 gives Pick-style right-justified behavior for LIST and SORT in all flavors. Pick-style right-justified behavior can overwrite data in previous columns if the data exceeds the column or the column header width. The default value is 0.
PROCACMD	Defines the action of the ProVerb A command. A value of 0 specifies that the A command quits when <i>m</i> characters are moved, or when a field mark or the end of the input buffer is reached. A positive nonzero value causes the A command to ignore the field mark. The default value is 0.

UniVerse Configurable Parameters (Continued)

Parameter	Description
PROCPRMT	Determines the effect of the UniVerse PROMPT on the ProVerb prompt. A value of 0 lets the PROMPT keyword change the ProVerb prompt. Any other value retains the ProVerb prompt, which can be changed only by the IP ProVerb command. The default value is 0.
PROCRCMD	Determines the behavior of the ProVerb RI command. A value of 0 clears the input buffer and removes the preceding field mark. Any other value does not remove the field mark. The default value is 0.
PSEMNUM	Sets the number of BASIC user process control locks. The default value 64.
QBREAK	Selects the function of the keys Q and Ctrl-X at the Press Any Key to Continue message. A value of 0 means that the Q and Ctrl-X keys are ignored while in BREAK OFF mode. A nonzero value allows the Q and Ctrl-X keys to quit at the Press Any Key to Continue message even in BREAK OFF mode. There are security implications to selecting the latter mode of operation. The default value is 1.
QDEPTH	Specifies the maximum depth of nesting allowed in Q-pointer references. The default value is 16, and the minimum value is 0.
QSBRNCH	Specifies the number of runs which cause a sub-merge to be performed by the query processor's sorting algorithm. This is sometimes referred to as the branching factor of the sort. The efficiency of the sorting algorithm is very sensitive to the value of QSBRNCH. The default value is 4, and the minimum value is 2.
QSDEPTH	Specifies the maximum depth of the sort tree used by the query processor's sorting algorithm. The efficiency of the sorting algorithm is very sensitive to the value of QSDEPTH. The default value is 8, and the minimum value is 2.
QSMXKEY	Specifies the maximum number of sort key components. The efficiency of the sorting algorithm is very sensitive to the value of QSMXKEY. The default value is 32.

UniVerse Configurable Parameters (Continued)

Parameter	Description
QSRUNSZ	Specifies the size of the initial sorting run used by the query processor's sorting algorithm. The efficiency of the sorting algorithm is very sensitive to the value of QSRUNSZ. The default value is 2000, and the minimum value is 2.
RLOWNER	Sets the number of lock owner entries maintained for shared record locks in a group semaphore set. The default value is 300.
RLTABSZ	Sets the number of update record lock entries in a group lock semaphore set. The default value is 75.
SCRMAX	Specifies the maximum size of the UniVerse scratch buffer pool. SCRMAX must be larger than SCRMIN, and must be specified after SCRMIN. The default value is 5.
SCRMIN	Specifies the minimum size of the UniVerse scratch buffer pool. SCRMIN must be at least 1 and must be specified before SCRMAX. The default value is 3.
SCRSIZE	Specifies the initial size of a scratch buffer. SCRSIZE must be from 512 through 2048. The default value is 512.
SELBUF	Specifies the size of the in-memory select list buffer. It is the amount of locally cached select data which can be stored before the select list starts using disk storage. It is specified in 1K units. The default value is 4.
SHDISPATCH	Defines the full path for the <i>sh</i> shell command. On UNIX systems the default is <i>/usr/bin/sh</i> . On Windows systems the default is NOT_SUPPORTED.
SMISDATA	Specifies whether UniVerse should treat a segment mark found in data as a data character (1) or a data terminator (0). This parameter affects data access in the file subsystem as well as BASIC functions such as EXTRACT and REPLACE when working with a dynamic array that contains one or more segment marks. If you set the value of this parameter to 0, UniVerse inserts data containing a segment mark, but truncates at the mark when retrieved.

UniVerse Configurable Parameters (Continued)

Parameter	Description
SPINSLEEP	On an HP multiprocessor system, when spinlocks are in use, this value determines the time to sleep between successive attempts to lock a spinlock. The value, expressed in microseconds, should be between 50 and 50000. If spinlocks are in use, and the value of this parameter is 0, UniVerse defaults to a value of 5000. Misuse of this tunable can drastically affect system performance.
SPINTRIES	On an HP multiprocessor system, this value determines whether spin locking is used instead of regular semaphore locking. This value determines the number of attempts to obtain the spinlock before the process sleeps. If you want to use spinlocks, choose a value between 5 and 500. A value of 0 switches spinlocks OFF. Misuse of this parameter can drastically affect system performance.
SQLNULL	You can assign this parameter any NULL value the system wants to use. The default value for SQLNULL is 128, which interferes with the EURO symbol value on Windows platforms. Change this value to another value if you are using EURO symbols.
SYNCALOC	A value of 1 causes creation of new UniVerse files to occur as soon as they are requested. The default value is 1.
SYSTEM_EURO	You can set this value to a system Euro codepoint value. The default is 128 on Windows platforms, and 164 on UNIX systems.
T30FILE	Specifies the number of dynamic files that can be opened. This is used to allocate shared memory concurrency control headers. The default value is 200.
TERM_EURO	This parameter can be set to a terminal Euro codepoint value. The default is 128 on Windows platforms, and 164 on UNIX systems.
THDR512	Specifies whether DR-type tapes are written with 512- byte labels. The default value is 0.
TIMEACCURACY	If the value of TIMEACCURACY is set to 1 (the default), the TIME() function displays time in millisecond/microsecond format, depending on the platform you are using. If the value of this parameter is set to 0, the TIME() function displays the time in 4 decimal format.

UniVerse Configurable Parameters (Continued)

Parameter	Description
TSTIMEOUT	Sets the number of seconds the UniVerse device licensing shell (<i>uvdls</i>) waits for a connection from a telnet client. The default value is 60.
TXMEM	Specifies the amount of memory allocated for the private transaction cache. This is specified in 1K units (1024 bytes). The default value is 32.
TXMODE	Sets the transaction mode observed by the system. When set to 0, transactions are not logged by the log daemon. The default value is 0.
TXNEXTHOLD	<p>Determines how the record controlling the next hold entry number is treated when using transaction processing.</p> <p>1 - If there is an active transaction, the handling of the NEXT.HOLD record in the D_&HOLD& file is fully compliant with transaction semantics, and the UniVerse file will hold the lock until the transaction is committed or aborted.</p> <p>0 - The handling of the NEXT.HOLD record is treated as in nontransaction mode, regardless of the current transaction status.</p>
TXWMFSYNCMODE	<p>Determines how warmstart transactions are synchronized to disk while the checkpoint mode is enabled on UNIX platforms. Valid values are:</p> <p>0 - UniVerse uses the UNIX fsync() system call for both the file header and data updates.</p> <p>1 - UniVerse sets the O_SYNC flag on the UNIX file descriptor for the file header updates, and uses the UNIX fsync() system call for data updates.</p> <p>2 - UniVerse uses the UNIX fsync() system call for file header updates and sets the O_SYNC flag on the UNIX file descriptor for data updates.</p> <p>3 - UniVerse sets the O_SYNC flag on the UNIX file descriptor for both file header and data updates.</p>
ULIMIT	Sets the maximum file size set by UniVerse. UniVerse uses either the value set by ULIMIT or the value set by the UNIX <i>ulimit</i> , whichever is larger. The default value of ULIMIT is 128000.

UniVerse Configurable Parameters (Continued)

Parameter	Description
UDRBLKS	Specifies the size of the internal buffer used for caching replicated data before it is written to the log file. One block is equivalent to 4096 bytes. The minimum size is 10, the maximum size is system-dependent. A larger size can improve performance on larger systems. The default size is 10.
UDRMODE	Sets the data replication mode. When set to 0, replication is not activated. When set to 1, replication is activated. The default value is 0.
UVLOGRLS	This parameter determines the amount to wait, in seconds, before releasing a log file. A value of 0, the default, implies that the uvchkd process calls fsync for all data files with updates in the current log file. A value greater than 0 implies that uvchkd does not call fsync, and waits to release the log file for the period of time you specify as the value of this parameter.
UVNET_CONNECT	The value of this parameter is either 0 or 1. A value of 1 implies to connect to the remote machine regardless of EURO data settings. A value of 0 implies communication takes place only if EURO settings match on both machines.
UVSPOOL	Specifies the name of the directory to be used as the UniVerse spooler directory. This should be a fully qualified path of 112 characters or less.
UVSYNC	Determines if UniVerse uses the UNIX <i>sync</i> () call. A nonzero value allows a <i>sync</i> () to be performed if a leading process exits. Data loss can occur if <i>sync</i> () is not executed often enough.
UVTEMP	Specifies the name of the directory used to contain UniVerse select lists and other temporary files. This should be a fully qualified path of 112 characters or less.

UniVerse Configurable Parameters (Continued)

Parameter	Description
UVTSORT	The value of this parameter can be either 1 or 0. A value of 1 enables multithreaded sort. A value of 0 disables multithreaded sort.
VDIVDEF	Selects the default action of the vector divide operator when the divisor values are exhausted prematurely. A nonzero value returns the dividend. A zero value returns 0. The default value is 1.
WIDE0	Specifies the mask used internally to decide when the difference between two numeric values is to be considered 0. The default value is 0x3dc00000. For more information about the wide zero parameter, see Appendix D, “The Wide Zero Parameter in UniVerse.”

UniVerse Configurable Parameters (Continued)

Changing Configurable Parameter Values

You can use UniAdmin to change configurable parameter values, or you can edit the *uvconfig* file manually.

When you change configurable parameter settings, you must save them in the *uvconfig* file. You must restart UniVerse for the new settings to take affect.

Changing the value of any of the following parameters changes the size of the shared memory segment:

FLTABSZ	LOGBLSZ	T30FILE
FSEMNUM	MAXKEYSIZE	UDRBLKS
GLTABSZ	PSEMNUM	UDRMODE
GSEMNUM	RLOWNER	
LOGBLNUM	RLTABSZ	



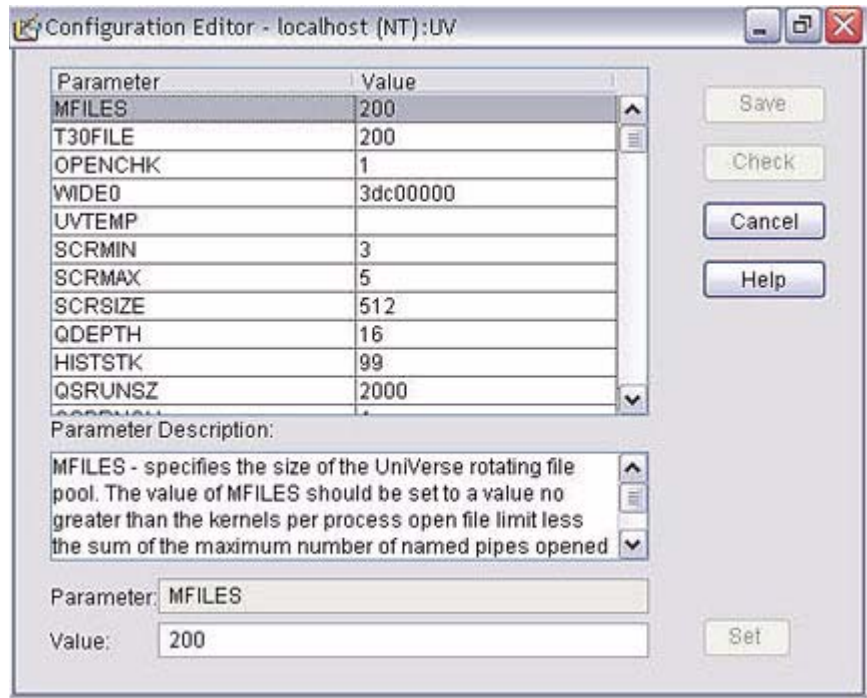
Note: Make sure you understand the effect of any changes you make to the default values. A small change can have a significant impact on your system. It is impossible to document the impact of each of these parameters so that you can predict the effect of a value change. If you are not sure about how to change the value of any parameter, please consult with an IBM Support Specialist.



Warning: The performance of the disk I/O subsystem can be profoundly affected by the concurrency control parameters (*FSEMNUM*, *GSEMNUM*, *PSEMNUM*, *FLTABSZ*, *GLTABSZ*, *RLTABSZ*, and *RLOWNER*). Use caution when changing any of these parameters.

Using UniAdmin to Change Parameter Values

To edit values for the UniVerse configurable parameters, choose **Configuration Editor** from the UniAdmin main menu.. The UniVerse Configuration Editor window appears with a list of configurable parameters and their current settings, as shown in the following example:



Use this window to change configurable parameter values.

Changing a Parameter

To change the value of a parameter:

1. Choose the parameter you want to edit from the list. A description of the parameter appears under **Parameter Description**, and the **Parameter** and **Value** fields are updated with the parameter name and value.
2. Enter a new value in the **Value** field. To return to the previously saved setting, choose the parameter from the list again.



3. Click **Set** to accept the new setting. The parameter value is updated in the list.
4. Repeat step 1 through step 3 to change other parameters before you finally save the new settings.
5. Click **Save**. A message box confirms that your changes have been saved and prompts you to restart UniVerse.
 - **On a UNIX System:** Choose a suitable response to the message box:
 - **Yes** to run *DBsetup* now. The *DBsetup* routine is executed and UniVerse is reinitialized with the new settings.

***Warning:** Great care must be taken with this option, as it affects any users still logged on to UniVerse.*
 - **No** to run *DBsetup* later. A second message box reminds you to restart UniVerse manually at a more convenient time. Click **OK** to acknowledge this message.
 - **On a Windows System:** Click **OK** to acknowledge the message. Before restarting the UniVerse service, make sure all users have logged out of UniVerse.

Editing the *uvconfig* File

To see a list of configurable parameter settings, use the CONFIG command at the UniVerse prompt with either the ALL or the DATA option.

To change the value of a configurable parameter, complete the following steps:

1. Make sure all users are logged out of UniVerse.
2. Stop all UniVerse processes.
3. Change the working directory to the UV account directory (make sure the UV account directory is in the current path).
4. Shut down UniVerse:
bin/uv -admin -stop
5. Edit the *uvconfig* file in the UV account directory.
6. Execute the UniVerse program *uvregen*:
bin/uvregen
7. Start up UniVerse:
bin/uv -admin -start

The Default *uvconfig* File

A partial sample of the default *uvconfig* file appears as follows:

```
#####
#
#   UniVerse tunable parameters
#
#   Version <number>   Date <date>
#
#   (c) Copyright 2003 IBM Corporation
#       All Rights Reserved
#   This is unpublished proprietary source
#   code of IBM Corporation.
#   The copyright notice above does not
#   evidence any actual or intended
#   publication of such source code.
#
#####

# MFILES - specifies the size of the
# UniVerse rotating file pool. The
# value of MFILES should be set to a
# value no greater than the kernels
# per process open file limit less the
# sum of the maximum number of named
# pipes opened by a user application
# and the 8 files reserved for internal
# UniVerse use.
MFILES 12
.
.
.
# UVSPOOL - is the name of the directory
# where the UniVerse printer routines are
# to build print files. This should be a
# fully qualified pathname of at most 112
# characters.
UVSPOOL /usr/spool/uv
.
.
.
# UVSYNC - This boolean if set will change the
# behavior of UniVerse calling the UNIX sync()
# call when exiting the environment. A non-zero
# means UniVerse will do a UNIX sync() if a job
# leading UniVerse process exits. This value should
# only be modified if you know exactly what you are
```

```
# doing. Data loss may occur if UNIX sync() is not
# executed frequently enough.
UVSYNC 1
.
.
.
```

Adding and Maintaining UNIX User Accounts

General Considerations	5-2
----------------------------------	-----

This chapter describes how to add new user accounts and how to maintain existing accounts on UNIX systems. It covers the following topics:

- Issues to consider when creating UniVerse and UNIX accounts
- How to create new user groups and user login accounts, and how to modify existing user accounts

General Considerations

When you first install UniVerse, the UV account is created. You use the UV account for both UNIX and UniVerse system administration. Probably the first task you perform after starting up the system is to add new user accounts. Before discussing the details of how to add new user accounts, it is worth considering a few general issues.

UniVerse users can work in either of two environments, the UNIX programming environment or the UniVerse database management environment. The difference between the two environments has some implications for how you assign user accounts, particularly if you plan to implement a protection scheme in which all files and commands are not available to all users.

User Accounts Differ from UniVerse Accounts

A UNIX user account and a UniVerse account are not identical.

UNIX user accounts are actually more like personal working environments that stay with users no matter what else they may be doing or where they may be working on the system. A user account is defined by an entry in the */etc/passwd* file that sets the user's login name and password and defines the user's home directory. Once UNIX users log on to the system, they have access to all directories and files on the system, except those protected by file permissions.

UniVerse accounts, on the other hand, are more self-contained. The user's working environment in UniVerse is determined primarily by the UniVerse account directory into which the user is currently logged. It is not determined, as in UNIX, by the user's login account. When users log on to a UniVerse account, they generally remain in that account and have access only to commands and files that are defined as available in that account. To access other commands and files, the user may need to log to the account containing the command or files, leaving the UniVerse account in which they have been working.

In the UNIX environment, each user is generally given a personal user account, which includes a home directory under which the user can create his or her own hierarchical directory tree of private files. Access to other parts of the system for UNIX users is easy. They can change their current working directories without changing other aspects of their account environment. They can access files and commands in other accounts simply by entering the full path that identifies the proper location in the file system's complete directory tree.

In the UniVerse environment, the account directory a user is logged on to and the user's working environment are more or less identical. The VOC file in each UniVerse account defines the account environment, including all the files and all the commands that are available to the user. Files in other UniVerse accounts are much less available than they are in the UNIX environment, although files in other accounts can be referenced in the VOC file. Generally, to use files in another account, you must log to that account.

Setting Up User Environments

If your site plans to make both environments available to users, it is probably best to create a UNIX login account for each user.

However, if you plan to have users working primarily in the UniVerse environment, and you are concerned about limiting access to the data in that UniVerse account to members of a selected group, you may want to create group UniVerse accounts (such as for members of a department).

All users of a particular UniVerse BASIC application can share a UniVerse account, since they all need access to the same commands, data files, and file dictionaries. For example, a UniVerse account might be defined for Sales rather than for an individual user.

Implement a group user account by giving each user his or her own login name at the UNIX level, but assign the same home directory (and therefore the same UniVerse account) to all users in the group.

Maintaining User Groups

All UNIX users must be assigned to at least one *user group*. User groups are used for setting file access permissions. You define user groups on the system by giving each user group a name and an ID number. When you add new users to the system, you give them a group ID number as well as a user ID number. For information about adding, changing, and deleting user groups, see [Maintaining Users and User Groups](#) in Appendix A, “UniVerse System Administration Menus.”

Maintaining User Accounts

To gain access to UniVerse, each UNIX user must have a unique login name that identifies him or her to the system. Users may also be required to enter a password as a security precaution to prevent unauthorized access to the system.

In addition to the required login name (and an optional password), each user must also have the following:

- Unique user ID number
- Group ID number
- Login shell
- Home directory

You provide this information when adding a new user to the system. You can use the UniVerse System Administration menus to add, change, or delete UNIX users. For details, see [Adding, Changing, and Deleting Individual Users](#) in Appendix A, “UniVerse System Administration Menus.”

UniVerse Accounts

About UniVerse Accounts	6-3
Creating a New UniVerse Account	6-4
Creating a New Account on a UNIX System.	6-5
Creating a New Account on Windows Platforms	6-6
Viewing or Modifying Account Details.	6-9
Deleting an Account.	6-10
Deleting a UniVerse Account on a UNIX System	6-10
Deleting a UniVerse Account on a Windows System	6-11
Customizing UniVerse Accounts	6-13
UniVerse Account Control Files	6-13
Essential UniVerse Files	6-16
Controlling Access to UniVerse on UNIX Systems.	6-18
Controlling Access to UniVerse on Windows Platforms	6-19
Customizing a UniVerse Account	6-19

This chapter describes how to add new UniVerse accounts and how to maintain existing accounts. It covers the following topics:

- How to create a UniVerse account, and how to carry out additional procedures that make the account function properly in either the operating system or the UniVerse environment
- How to delete a UniVerse account
- How to customize a UniVerse account

When you first install UniVerse, the UV account is created. One of your first tasks after starting up the system is to add new UniVerse accounts.

About UniVerse Accounts

You always enter UniVerse through a UniVerse account. A UniVerse account includes a directory containing the files required to run UniVerse in that directory. An established UniVerse account can also contain database files and program files.

The VOC file in each UniVerse account defines the account environment, including all the files and commands that are available to users who are logged on to the account.

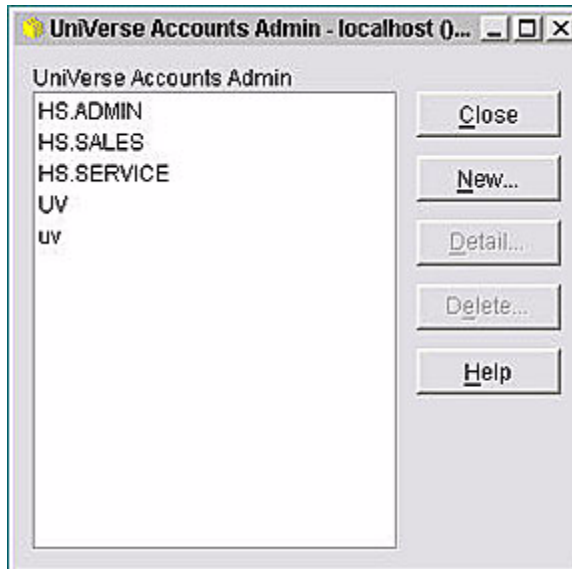
For example, a UniVerse account might be defined for a department rather than for an individual. Each user of the SALES account might be given his or her own login name at the operating system level but be assigned the same home directory and share the same UniVerse account.

On UNIX systems you can assign or change the user ownership and group ownership of files in an account, and you can set or change the file access permissions. These ownerships and permissions apply to all of the files and subdirectories contained in a UniVerse account directory.

UniVerse file permissions are managed by UNIX or Windows file permissions. For information about setting file permissions, refer to the documentation that comes with your operating system.

Creating a New UniVerse Account

Choose the **Accounts** option from the UniAdmin main window to create a new UniVerse account. The **UniVerse Accounts Admin** window appears with a list of all UniVerse accounts currently defined in the UV.ACCOUNT file (see The UV.ACCOUNT File for information about this file). The following example illustrates the UniVerse Accounts Admin window:



The tasks you can perform from this window include:

- Creating a new account
- Viewing or modifying account details
- Deleting an account

How you create an account depends on whether you are administering a UNIX server or a Windows server.

Note: Users can also create UniVerse accounts by entering **uv** at an operating system prompt. If an account is created this way, the UV.ACCOUNT file is not updated and the account cannot be administered using the **Accounts** option.

When you add a new account, UniVerse does the following:



- Assigns an account compatibility flavor
- Updates the UV.ACCOUNT file
- On UNIX systems, edits the *.profile* file in the account directory
- Edits the LOGIN entry in the UniVerse account

Creating a New Account on a UNIX System

To create a new UniVerse account:

1. From the **UniVerse Accounts Admin** window, click **New**. The **Create account** dialog box appears, as shown in the following example:

The screenshot shows a dialog box titled "Create Account - colt ():UV". It contains the following fields and controls:

- Account Name:** An empty text input field.
- Account Flavor:** A dropdown menu currently showing "IDEAL".
- Owner:** A dropdown menu currently showing "root".
- Group:** A dropdown menu currently showing "system".
- Pathname:** A text input field containing "alpha/uv102_060727_6092".
- File Permissions:** A section with a checked checkbox "Use Defaults" and a text field showing "rwxrwxrwx". There is also an unchecked checkbox "Use Default Login".
- Buttons:** "OK", "Cancel", "Help", and "Browse..." (located next to the Pathname field).

2. Enter the name of the account in the **Account Name** field.
3. Select one of the following flavors from the Account Flavor list:
 - **IDEAL**. Choose this flavor if you are new to UniVerse. It contains the best features of all the flavors.
 - **INFORMATION**. Choose this flavor for compatibility with Prime INFORMATION.
 - **PIOPEN**. Choose this flavor for compatibility with PI/open.
 - **PICK**. Choose this flavor for compatibility with Pick or Advanced Pick.
 - **REALITY**. Choose this flavor for compatibility with Microdata REALITY.
 - **IN2**. Choose this flavor for compatibility with IN2.



4. Enter a destination for the new account in the **Pathname** field. Do one of the following:

- Click **Browse** to search the system for an appropriate directory.
- Enter the path of a directory directly. If you enter the name of a directory that does not exist, it is created when you click **OK**. For example:

/usr/users/newuser

The parent directory (*/usr/users*) must exist.

***Note:** You can choose a directory path of an existing UniVerse account. In this case, the new account is added to the UV.ACCOUNT file, but no changes are made to the existing account files.*

5. Select the owner of the account from the **Owner** list.
6. Select a group to assign the new account to from the **Group** list.
7. Select the **Use Default LOGIN** check box if you do not want to use the default LOGIN entry for the new account.
8. Set the file permissions by doing one of the following:
 - Select the **Use Defaults** check box to accept the default permissions *rwx-rwxrwx*.
 - Enter suitable settings in the **Permissions** field. This field is active when the **Use Defaults** check box is cleared.
9. Click **OK**. The UniVerse account is created in the chosen directory, and the UV.ACCOUNT file and the **UniVerse Accounts Admin** window are updated.

Creating a New Account on Windows Platforms

To create a new UniVerse account, complete the following steps:

1. From the **UniVerse Accounts Admin** window, click **New**. The **Create account** dialog box appears, as shown in the following example:

2. Enter the name of the account in the **Account Name** field.
3. Select one of the following flavors from the **Account Flavor** list:
 - **IDEAL**. Choose this flavor if you are a new user. It contains the best features of all the flavors.
 - **INFORMATION**. Choose this flavor for compatibility with Prime INFORMATION.
 - **PIOPEN**. Choose this flavor for compatibility with PI/open.
 - **PICK**. Choose this flavor for compatibility with Pick or Advanced Pick.
 - **REALITY**. Choose this flavor for compatibility with Microdata REALITY.
 - **IN2**. Choose this flavor for compatibility with IN2.
4. Enter a destination for the new account in the **Pathname** field. Click **Browse** to search the system for the appropriate directory. You can also enter the path of a directory directly. If you enter the name of a directory that does not exist, it is created when you click **OK**. For example:
D:\uv\accounts\newuser
The parent directory (*D:\uv\accounts*) must exist.
5. Clear the **Use Default LOGIN** check box if you do not want to use the default LOGIN entry for the new account.
6. Click **OK**. The UniVerse account is created in the chosen directory, with the ownership and security of the parent directory. The UV.ACCOUNT file and the **UniVerse Accounts Admin** window are updated.



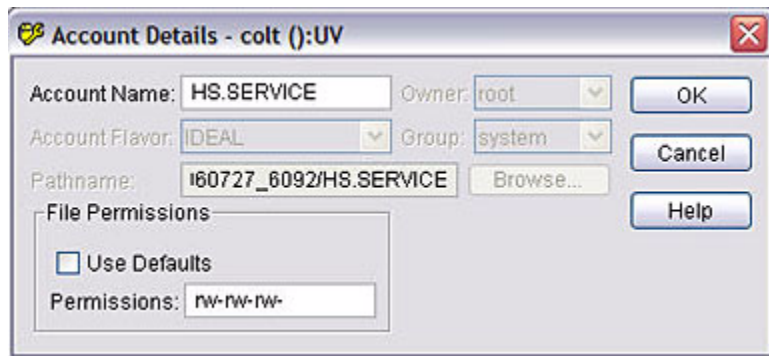
Note: You can choose a directory path of an existing UniVerse account. In this case, the new account is added to the UV.ACCOUNT file, but no changes are made to the existing account files in the directory.

Viewing or Modifying Account Details

To view the details of an account, execute one of the following steps:

- From the **UniVerse Accounts Admin** window, double-click the account in the **UniVerse Accounts Admin** list.
- Choose an account and click **Detail**.

The **UniVerse Account Details** dialog box appears, as shown in the following example:



You can modify the account settings, except for the **Account Flavor** setting and account path. Changes are saved when you click **OK**.

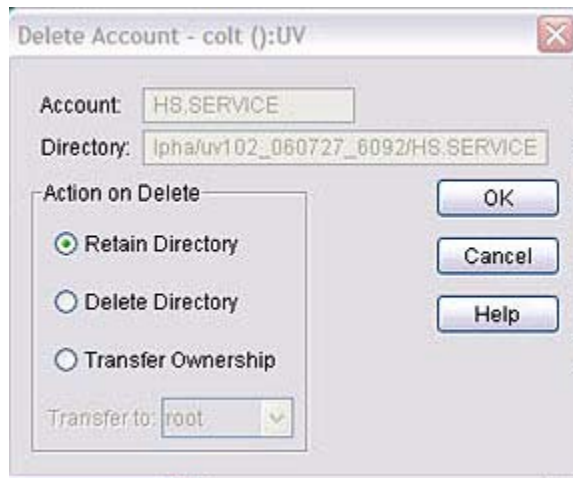
Deleting an Account

To delete a UniVerse account, choose the **Accounts** option from the **UniAdmin** main window. How you delete the account depends on whether you are connected to a UNIX or a Windows server.

Deleting a UniVerse Account on a UNIX System

Complete the following steps to delete a UniVerse account:

1. Select the account you want to delete from the **UniVerse Accounts** list in the **UniVerse Accounts Admin** window.
2. Click **Delete**. The **Delete Account** dialog box appears with the name and location of the chosen account, as shown in the following example:





3. Choose how you want to delete the account by clicking on of the following options:

- **Retain Directory.** UniVerse removes the account from the UV.ACCOUNT file.
- **Delete Directory.** The account is removed from the UV.ACCOUNT file and the directory, along with all of its contents, is deleted. If other accounts in the UV.ACCOUNT file use the files in this directory, you will also be prompted whether you want to delete these accounts.
- **Transfer Ownership.** The account is removed from the UV.ACCOUNT file and the account directory ownership is transferred to a specified user. Select a user from the Transfer to list.

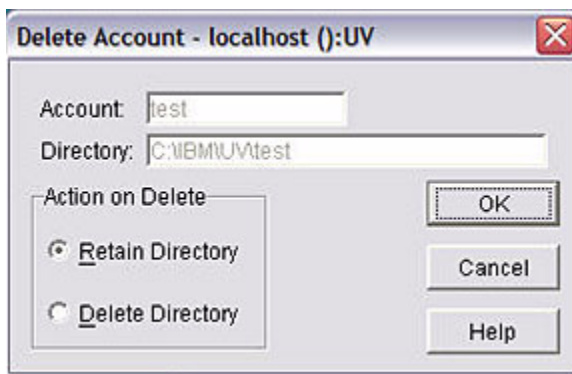
***Note:** Options available in the dialog box change dynamically according to your choice to retain or delete a directory.*

4. Click **OK**. A message box appears.
5. Click **Yes** to remove the account as specified. The **Account Admin** window is updated.

Deleting a UniVerse Account on a Windows System

To delete a UniVerse account:

1. Select the account you want to delete from the **UniVerse Accounts** list in the **UniVerse Accounts Admin** window.
2. Click **Delete**. The **Delete Account** dialog box appears with the name and location of the chosen account, as shown in the following example:



3. Choose how the account is deleted by clicking the appropriate option:
 - **Retain Directory.** The account is removed from the UV.ACCOUNT file.
 - **Delete Directory.** The account is removed from the UV.ACCOUNT file and the directory, along with all of its contents, is deleted. If other accounts in the UV.ACCOUNT file share this directory, you are asked whether you want to delete these accounts.
4. Click **OK**. A message box appears.
5. Click **Yes** to remove the account as specified. The **UniVerse Accounts Admin** window is updated.

Customizing UniVerse Accounts

You can customize UniVerse accounts. For example, you can prevent certain users from creating or modifying accounts from their own UniVerse accounts.

You can also set up alternative account flavors and their associated VOC files. To customize your system in this way, you must modify the NEWACC file and the UV.FLAVOR file.

Using UniAdmin, you can specify the account flavor to use, which in turn affects the VOC file and the user's access to UniVerse. There are six standard flavors: IDEAL, INFORMATION, PICK, REALITY, PIOPEN, and IN2. These are listed when you create accounts using the **Accounts** option from UniAdmin.

On Windows platforms, you can specify the account directory or UniVerse account to which each user initially logs on from a telnet session. Use the Network Services option of the UniAdmin to do this.

UniVerse Account Control Files

UniAdmin uses the account control files to check the validity of responses to some of the data entry screens. These files are updated only when you create or modify an account using UniAdmin, so it is important to create accounts using UniAdmin.

Although these files are used primarily by UniAdmin, you can use LIST and SORT them to create reports. You can update them with ReVise or the UniVerse Editor.

The UV.ACCOUNT File

The UV.ACCOUNT file is in the UV account. It contains a list of UniVerse accounts and their path names. This file is automatically updated when you create or delete an account using UniAdmin. Each UniVerse account has a record in the UV.ACCOUNT file. The record ID is the account name. Each record also contains the following fields, but only the PATH field is updated by UniVerse.

Field Name	Description
@ID	Account name
PASSWORD	Account's password (<i>/etc/passwd</i>)
MAX	
MIN	
AGE	
UID	User ID number (<i>/etc/passwd</i>)
GID	Group ID number (<i>/etc/passwd</i>)
NAME	Account owner's name (<i>/etc/passwd</i>)
OFFICE	
EXT	
PHONE	
PATH	Account directory's pathname (<i>/etc/passwd</i>)
SHELL	UNIX shell (<i>/etc/passwd</i>)
PERMISSIONS	

UV.ACCOUNT Fields

The UV.FLAVOR File

The UV.FLAVOR file is in the UV account. The UV.FLAVOR file dictionary contains X-descriptors that define each flavor. The UV.FLAVOR data file contains records that specify restrictions on creating or updating accounts. This file lets you choose the account flavor for an individual or for a group of users.

Each record in UV.FLAVOR can have one of the three types of record ID:

Record ID	Description	Example
USER. <i>name</i>	<i>name</i> is the login name of a user.	USER.alice
GROUP. <i>name</i>	<i>name</i> is the name of a user group.	GROUP.users
OTHER	A specially defined account.	OTHER

UV.FLAVOR Record IDs

Each record has two fields. The second field specifies one of the six different flavors: IDEAL (UniVerse), INFORMATION, PICK, REALITY, PIOPEN, or IN2. The first field specifies one of the following codes:

Code	Description
C	The user is prompted to choose an account flavor when creating or updating an account.
F	The account is automatically assigned the flavor designated in field 2.
N	The user cannot create or update an account.

UV.FLAVOR Record Attributes

The following example is of a sample UV.FLAVOR file:

```

UV.FLAVOR..... Access
Code.. Flavor.....

GROUP.users      F      PICK
OTHER            C
GROUP.demo       F      NEWACC

```

To prevent users creating or modifying an account, create an entry for them in the UV.FLAVOR file and set the access code to N.

To use a custom flavor, create an entry in the UV.FLAVOR file for users who will use the flavor, and set the access code to F. Enter the custom flavor name in field 2 of the UV.FLAVOR file.



The UV.LOGINS File on Windows Platforms

The UV.LOGINS file is in the UV account only on Windows platforms. It contains a list of users and the UniVerse accounts they log on to when they first connect to UniVerse through a telnet session. Use the Network Services option of UniAdmin to maintain this file.

Note: *There is no UV.LOGINS file on UNIX systems.*

Essential UniVerse Files

For users to work in the UniVerse environment, their current working directories must contain a number of UniVerse files, including the VOC file and its associated file dictionary. In addition, each UniVerse account is set up in a specified flavor of compatibility, such as IDEAL (UniVerse), PICK, or INFORMATION.

The system administrator does not set up the VOC file and its associated file dictionary. They are created when the user logs on to the new UniVerse account for the first time.

If the directory has not been set up as a UniVerse account, the system notifies the user that the account has not yet been set up. The user must answer the system prompts to create or update the VOC file.

The VOC File

The VOC file is created according to the restrictions specified in the UV.FLAVOR file. The master files used as templates for creating VOC files are in the file NEWACC.

If your VOC is being updated rather than created, replaced records are moved to the file&TEMP& to prevent them from being destroyed. The names of any records that are moved to &TEMP& are listed on your screen.



Note: *The VOC file defines the UniVerse account. The contents of the VOC file limits access to commands and files in a UniVerse account. Users cannot access any files or commands not defined in the VOC file of the account in which they are working.*

The UV.LOGIN and LOGIN Entries

If the UniVerse command processor is specified as the account's command interpreter, UniVerse executes the UV.LOGIN entry in the VOC file of the UV account when the user logs on to the account. The UV.LOGIN entry can be a paragraph, a proc, a UniVerse BASIC program, or a menu. It is typically a paragraph containing commands that establish systemwide defaults. After executing UV.LOGIN, UniVerse executes the LOGIN entry in the VOC file of the user's account.

The standard VOC file on UNIX systems contains a LOGIN entry that is analogous to the UNIX *.profile* file. The default LOGIN entry in the *sample* directory of the UV account is a paragraph that looks like this:

```
LOGIN
001 PA
002 PTERM ERASE ON KILL ON WERASE ON RPRNT ON FLUSH ON LNEXT ON
SUSP ON_
003 INTR ON QUIT ON STOP ON START ON EOF ON BRK OFF_
004 ECHO ON ECHO CTRL ON TABS ON CRMODE ON TYPE FAST LFDELAY 0
FFDELAY 2
005 UMASK 077
```

This LOGIN entry uses PTERM (UNIX) to set terminal characteristics, and it uses the UniVerse UMASK command to set the default file permission mask. These commands have the same function as the UNIX commands *stty* and *umask*.

Many of the functions performed by the LOGIN entry are identical to those performed in the UNIX *.profile* file. For example, the *stty* command in *.profile* determines which keys perform erase, kill, interrupt and quit operation on the user's terminal:

```
stty erase '^H' kill '^U' intr '^?' quit '^_' -tabs ff0 cr0 nl0
```

On Windows platforms, the default LOGIN entry in the sample directory is a paragraph that looks like this:

```
LOGIN
001 PA
002 PTERM ERASE ON KILL ON WERASE ON RPRNT ON INTR ON_
003 ECHO ON ECHO CTRL
004 CLR
```

Controlling Access to UniVerse on UNIX Systems

You can make UniVerse the user’s default working environment by entering */usr/ibm/uv/bin/uv* as the user’s default shell in the */etc/passwd* file. When users log on and out, they log directly on to and out of UniVerse.

If, on the other hand, the */etc/passwd* file specifies a UNIX shell (for example, */bin/sh*), the user logs on to a UNIX shell. The user can then invoke the UniVerse environment with the *uv* command. Even if the */etc/passwd* file specifies a UNIX shell, the user’s *.profile* or *.login* file can log the user directly on to the UniVerse environment. To do that, add the following line to the user’s *.profile*:

```
exec uv
```

The *exec* command replaces the current shell with the shell specified, in this case *uv*. On exiting UniVerse, the user also exits the system.

Login Shell Specified in <i>/etc/passwd</i>	Initialization Files	User logs in...	User logs out...
<i>/usr/ibm/uv/bin/uv</i>	LOGIN	Directly to UniVerse.	To a UNIX login shell.
<i>/bin/sh</i>	<i>.profile</i>	To a UNIX Bourne shell.	To a UNIX login shell.
<i>/bin/sh</i>	<i>.profile</i> containing <i>exec uv</i>	To UniVerse. The UNIX login is transparent to the user.	To a UNIX login shell.
<i>/bin/csh</i>	<i>.cshrc</i> <i>.login</i>	To a UNIX C shell.	To a UNIX login shell.
<i>/bin/csh</i>	<i>.cshrc</i> <i>.login</i> containing <i>uv</i> command	To UniVerse. The UNIX login is transparent to the user.	To a UNIX C shell.

Login Shell and Environment on Exiting UniVerse

Note: If users interrupt execution of the *.profile* or *.login* file (for example, by pressing the **Break** key) before the *uv* command is executed, they are left in a UNIX shell.



Controlling Access to UniVerse on Windows Platforms

The UV.LOGINS file is used on Windows platforms to define how users connect to UniVerse via telnet sessions. Use the Network Services option of the UniAdmin to specify how users should connect to UniVerse.

Customizing a UniVerse Account

A valid UniVerse account always includes a VOC file and its associated file dictionary. The VOC file defines all the commands and keywords that can be used, and all the files that can be accessed from that account. Master files in the UV account directory are used to create the VOC files in all new accounts.

Choosing a UniVerse Flavor

Any UniVerse account can be one of several standard flavors: IDEAL (UniVerse), IN2, INFORMATION, PICK, PIOPEN, or REALITY.

- The PIOPEN flavor is used for compatibility with PI/open.
- The INFORMATION flavor is used to maintain an environment compatible with Prime INFORMATION products.
- The IN2, PICK, and REALITY flavors are used for compatibility with the different versions of the Pick system. These flavors can be chosen by users who are more comfortable with a Pick system and want UniVerse to behave in the same way.
- The IDEAL flavor contains the best of both the Pick and Prime worlds.

New users are encouraged to choose the IDEAL UniVerse flavor.

NEWACC Files

The NEWACC file in the UV account contains the different VOC file templates for each flavor of UniVerse. These templates are stored as multiple data files of the NEWACC file. Each data file is a fully configured VOC template whose name corresponds to the flavor. To list the contents of the data file containing the template for IDEAL flavor VOC files, enter either of the following commands from the UV account:

```
>LIST NEWACC  
>LIST NEWACC,NEWACC
```

To list the contents of the NEWACC template for INFORMATION flavor VOC files, enter:

```
>LIST NEWACC, INFORMATION
```

The VOC file can reference a particular VOC template as a single data file by using its full path name in field 2 of the File Definition record. See the File Definition record for NEWACC in the VOC file in any UniVerse account other than the UV account. For example, this VOC entry points to the NEWACC template for PICK flavor VOC files:

```
NEWACC
001 F File
002 /usr/ibm/uv/NEWACC/PICK
003 /usr/ibm/uv/D_NEWACC
```

Customizing NEWACC Files

You can modify the standard NEWACC files to ensure that the VOC files of new accounts contain only the records you want. For example, you can remove records for commands that you do not want users to access, or you can add records for files that are needed for an application.

You can also create up to 27 additional customized NEWACC files (see *UniVerse System Description* for information about adding data files to a UniVerse file). Each NEWACC file is a template for a new flavor of UniVerse. For each new flavor, you must add an X-descriptor to the dictionary of the UV.FLAVOR file. The record ID of the X-descriptor is the name of the new NEWACC file, and field 2 contains the description of the new flavor. This description appears in the list of UniVerse flavors when you create new accounts. The following steps describe the easiest way to create a customized flavor:

1. Change to the UV account directory and invoke UniVerse:

```
# cd /usr/ibm/uv
# bin/uv
```

2. Make a copy of one of the standard NEWACC files. Do this by creating a new data file in NEWACC and copying the contents of the standard NEWACC file to the new data file:

```
>CREATE.FILE DATA NEWACC,MY.FLAVOR 3 23 4
Creating file "/usr/ibm/uv/NEWACC/MY.FLAVOR" as Type 3, Modulo 23,
Separation 4.
>COPY FROM NEWACC,INFORMATION TO NEWACC,MY.FLAVOR ALL
355 records copied.
```

3. Use the UniVerse Editor or ReVise to add, delete, or change standard VOC entries in your new flavor.
4. Use the UniVerse Editor to add an X-descriptor to the DICT of UV.FLAVOR:

```
>ED DICT UV.FLAVOR
Record name = MY.FLAVOR
New record.

----: I
0001=X
0002=My own custom UniVerse flavor
0003=
Bottom at line 2
----: FI
"MY.FLAVOR" filed in File "DICT UV.FLAVOR"
```

Transferring Accounts

Transferring Non-UniVerse Accounts	7-3
Manually Restoring Accounts from Tape	7-6
Restoring Accounts to UNIX Systems.	7-6
Restoring Accounts to Windows Systems	7-15
Transferring UniVerse Accounts from UNIX to Windows Platforms . . .	7-22
Creating the Backup Image	7-22
Transferring the Backup Image	7-23
Restoring the Backup Image.	7-23
File Naming Conventions	7-24

This chapter describes:

- How to transfer non-UniVerse accounts to UniVerse
- How to transfer UniVerse accounts from UNIX to Windows platforms

Transferring Non-UniVerse Accounts

UniVerse provides two commands for transferring non-UniVerse accounts to UniVerse:

- The *acct.restore* command transfers an account from a Pick system ACCOUNT-SAVE tape to a UniVerse PICK flavor account.
- The *magrst* command transfers an account from a Prime INFORMATION MAGSAV tape to a UniVerse INFORMATION flavor account.

In both cases the transfer and conversion of data has five steps:

1. Make an ACCOUNT-SAVE or MAGSAV tape.
2. Load the account onto the UniVerse system.
3. Convert the account from its original format to UniVerse format.
4. Compile the converted UniVerse dictionaries.
5. Convert and compile the UniVerse BASIC programs, recataloging them if necessary.

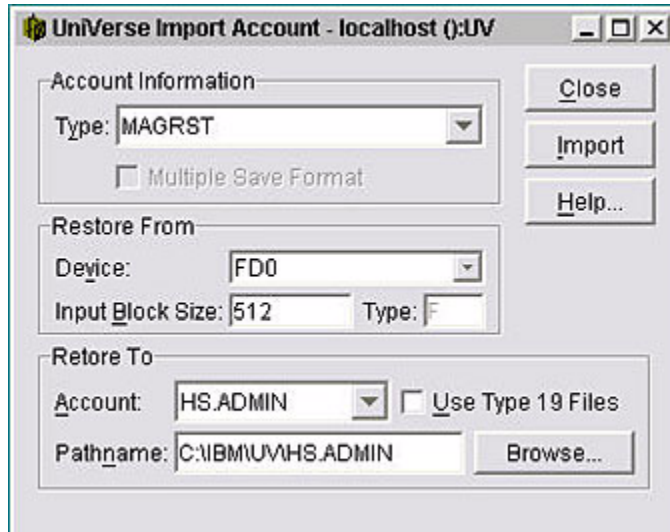
You can use one of two methods to transfer accounts:

- Use UniAdmin.
- Enter the commands from a UNIX shell or an MS-DOS window.

First we describe how to use UniAdmin to transfer Pick and Prime INFORMATION accounts from tape. Later sections describe how to use operating system commands to transfer Pick and Prime INFORMATION accounts.

You can also use the Account Conversion menu, displayed by the CONVERT.ACCOUNT command, to convert the transferred accounts.

To transfer non-UniVerse accounts to UniVerse, choose **Import** from the UniAdmin main window. The **Import Account** window appears, listing all the information required to transfer an account to UniVerse, as shown in the following example:



To transfer a non-UniVerse account to UniVerse:

1. Select one of the following account types from the Type list:

- MAGRST
- PICK
- REALITY (Microdata)
- IN8000
- IN5000

The option you choose determines which import program is used and what tape positioning is required.

2. Select the **Multiple Save Format** check box, if required. This option is available only if you chose **REALITY (Microdata)** as the account type.
3. Select the tape device to use from the **Device** list. When you select a tape device, the **Input Block Size** and **Type** fields are populated with the settings stored in the &DEVICE& file.
4. Enter an alternative block size in the **Input Block Size** field, if required.

5. Choose one of these destinations for the transferred account:
 - Select an account from the **Account** list. The account path is entered in the **Pathname** field.
 - Enter a path in the **Pathname** field, or click **Browse** to search the system for a suitable directory.
6. Select the **Use Type 19 Files** check box, if required. This setting determines whether the account is restored using type 19 files or type 1 files. The default setting is unchecked, for example, type 1 files are used.
7. Click **Import**. The import begins and the results of the transfer appear in the **UniVerse Command Output** window.
8. Click **Close** to close this window.

Manually Restoring Accounts from Tape

You can also restore non-UniVerse accounts directly from tape. You may need to use this method if you require additional import options that are not available in the UniVerse Admin **Import** option.

How you restore accounts from tape depends on whether you are restoring to a UNIX or a Windows server.

Restoring Accounts to UNIX Systems

You can manually restore Pick or Prime INFORMATION accounts from tape to a UNIX server. If you are restoring a single-reel ACCOUNT.SAVE or MAGSAV tape, execute the following steps:

1. Log on as a UniVerse Administrator.
2. Create a new directory to be the parent directory for the account. The directory name can be anything, so you might enter:
mkdir /u1/demo
3. Move to that directory. For example:
cd /u1/demo
4. Use *acct.restore* to load a Pick account, or use *magrst* to load a Prime INFORMATION account. These commands create a UniVerse account in the current directory as well as subdirectories that contain the accounts loaded from the tape.

If you are restoring a multireel ACCOUNT-SAVE or MAGSAV tape, execute a UniVerse program called *tapein*.

Using acct.restore

The syntax for *acct.restore* is as follows:

acct.restore [*options*]

The following table describes the *acct.restore options*:

Option	Description
–	Reads input from standard input.
–a <i>pathname</i>	Reads input from the specified UNIX path. <i>pathname</i> is the full UNIX path.
–d	Reads input from a device other than a tape.
–D <i>device</i>	The path of a terminal (tty) device used when multiple instances of <i>acct.restore</i> are run.
–i	Restores data to an IN2 account.
–m	Restores data from a Microdata REALITY tape.
–n	Creates files with a default separation of 4 (each group buffer holds 2048 bytes). Use this option when you are transferring accounts from systems such as ADDS Mentor or PICK 370, where a separation of 1 means 2K bytes instead of 512 bytes.
–s	Used with the <i>tapein</i> filter. Sends the appropriate signals back to the <i>tapein</i> filter so <i>tapein</i> can prompt for the next volume of input and exit cleanly after <i>acct.restore</i> finishes.
–t <i>device</i>	Reads input from a tape device defined in the &DEVICE& file. <i>device</i> is the ID of the entry in the &DEVICE& file. See The &DEVICE& File in Chapter 10, “ Configuring Peripheral Devices ,” for information about the &DEVICE& file.
–u	Restores data using Ultimate cartridge format.
–19	Restores type 1 files as type 19 files.

acct.restore Options

Using magrst

The syntax for *magrst* is as follows:

magrst [*options*]

The following table describes the options for *magrst*.

Option	Description
–	Reads input from standard input.
–a <i>pathname</i>	Reads input from the specified UNIX path. <i>pathname</i> is the full UNIX path.
–d	Reads input from a device other than a tape.
–n	Creates files with a default separation of 4 (each group buffer holds 2048 bytes). Use this option when you are transferring accounts from systems such as ADDS Mentor or PICK 370, where a separation of 1 means 2K bytes instead of 512 bytes.
–s	Used with the <i>tapein</i> filter. Sends the appropriate signals back to the <i>tapein</i> filter so <i>tapein</i> can prompt for the next volume of input and exit cleanly after <i>magrst</i> finishes.
–t <i>device</i>	Reads input from a tape device defined in the &DEVICE& file. <i>device</i> is the ID of the entry in the &DEVICE& file. See “The &DEVICE& File,” for information about the &DEVICE& file.
–18	Restores dynamic files as type 18 files.
–19	Restores type 1 files as type 19 files.
–634	Restores tape using old SAM file code.

magrst Options

If you specify the *acct.restore* or *magrst* command without any arguments, the command uses MT0 as the default tape device.

When using standard input, you can use normal UNIX redirection or pipe syntax to select the input. For example:

```
# magrst - < device
```

When you use standard input, the first record is assumed to be the start of the data. You must use UNIX tape positioning commands or dummy *acct.restore* or *magrst* calls to position the tape over label records or file marks. For example:

```
# magrst - < device> /dev/null
```

If *acct.restore* is executed by *tapein*, or if it is reading input from standard input, it does not prompt the user when an error condition exists, such as unrecognized data or a file cannot be opened. It ignores the data or error and continues with the restoration.

If the MAGSAV tape is constructed on a PRIMOS environment of Release 20 or later, you must specify the `–NO –ACL` option of the MAGSAV command. This makes tapes that are compatible with Release 18. At Release 21, you must also specify the `–REV19` option to create tape formats compatible with Release 19. UniVerse does not support tape formats compatible with Release 20 or later formats.

Using pqic

The *pqic* utility is a filter that decomposes Prime T.DUMP images. The *pqic* utility uses the 4-byte header to determine how many bytes of data are present. Then it writes the data to the standard output channel. After you run the *pqic* utility, you store the results in a file. Then you run the UniVerse T.LOAD utility to read from the file and store the data in a UniVerse data file.

Cartridge tapes made on a Prime 50 Series machine using the T.DUMP utility are written using the Prime 2350/2450 QIC stream tape format. The UniVerse T.LOAD utility does not recognize this format and cannot read these images.

To run *pqic*, you must know the tape device name. Do the following to run *pqic*:

1. Log on to the UniVerse machine.
2. Be sure that you have the UniVerse *bin* directory in your path.
3. Put the tape in the tape drive.
4. Check that you have write permissions to the directory where *pqic* results will be written.
5. From a UNIX shell prompt, enter the following command:

```
dd device.name | pqic > filename
```

In the following example, the UNIX utility *dd* dumps the contents of the tape from the specified tape drive and pipes it to the *pqic* utility. Then *pqic* writes to standard output, which is redirected to the *pqic.results* file.

```
# dd /dev/rmt/0h | pqic > pqic.results
```

See the *UNIX System Administrator's Manual* for information about the correct device names for your system.

Running T.LOAD from a File

Before running T.LOAD, you must set up an &DEVICE& entry to access the *pqic.results* file. Specify a device type of O (for Other) in field 4, and specify the full path of the *pqic.results* file for the device paths (rewind and no rewind (fields 6 and 7)).

Once the device entry has been defined, use the ASSIGN command to assign control of the newly defined device for use with the T.LOAD command.

Shortening Names for acct.restore on UNIX Systems

On System V UNIX systems, type 1 record names cannot be longer than 41 characters. When *acct.restore* encounters a record name longer than 41 characters, it truncates the name to 41. If the new record name already exists, the last two characters are also removed and a two-digit number from 1 through 99 is appended. For example, the record:

```
A . REALLY . REALLY . REALLY . REALLY . REALLY . LONG . NAME
```

becomes:

```
A . REALLY . REALLY . REALLY . REALLY . REALLY . LONG
```

If this name already exists, it gets changed to:

```
A . REALLY . REALLY . REALLY . REALLY . REALLY . LO01
```

A list of the record name changes is written to a file called &TRUNCATED& in the restored account. (The file dictionary D_&TRUNCATED& is located in */u1/uv*.) The *acct.restore* program logs all shortened filenames and record names in this file. Therefore, if you use *acct.restore* to restore over an existing account, you should retain the &TRUNCATED& file in that account to ensure that long names are correctly mapped to their truncated versions.



Support for Long Filenames on UNIX Systems

On some System V computers (such as Pyramid, Encore, and Hewlett-Packard) UniVerse supports a LONGNAMES mode. The file name length allowed is machine-dependent. This mode is controlled by the LONGNAMES command. To turn on long filename support, enter LONGNAMES ON. To turn it off, enter LONGNAMES OFF. LONGNAMES ON modifies the commands CREATE.FILE, CNAME, and DELETE.FILE, by adding a V to field 4 and adding the word LONGNAMES to field 5 of the VOC entry for each of these verbs.

The *acct.restore* and *magrst* commands look at the VOC entry for CREATE.FILE to determine whether to create long file names or to truncate the file names.

Note: If you use the LONGNAMES ON mode to create long filenames, it may result in files that are not portable from one UNIX system to another.

Removing Labels from ACCOUNT-SAVE Tapes

Because of the many variations of tape labelling on ACCOUNT-SAVE tapes, you may need to remove labels before *acct.restore* can process the ACCOUNT-SAVE tape. Use the *rmv.lbl* command to remove tape labels. The source for *rmv.lbl* is included in the file *sample/rmv.lbl.c* in the UniVerse account directory.

The syntax for *rmv.lbl* is as follows:

rmv.lbl [*-ssize*] [*-input.file*] [*-output.file*]

The following table describes each parameter of the syntax.

Parameter	Description
<i>-ssize</i>	The size of the label to remove in bytes. The default size is 80. The maximum label size is 16384.
<i>-input.file</i>	The input file path. The default is standard input.
<i>-output.file</i>	The output file path. The default is standard output.

rmv.lbl Parameters

A typical use of *rmv.lbl* would be in a script that creates UNIX disk files containing just the data area of the ACCOUNT-SAVE tape, using *dd* to strip off the leading EOF marks, and *rmv.lbl* to remove the labels, as follows:

```
dd  if=/dev/rstp/0nn  ibs=8192  >/dev/null  Remove leading EOF
dd  if=/dev/rstp/0nn  ibs=8192  >/dev/null  Remove the head file
dd  if=/dev/rstp/0nn  ibs=8192  |  rmv.lbl  -s80 >diskfile1
dd  if=/dev/rstp/0nn  ibs=8192  |  rmv.lbl  -s80 >diskfile2
.
.
.
[etc. for each successive tape]
acct.restore  -a diskfile1
```

When *acct.restore* finishes processing the first file, it prompts for the remaining files. Make sure that you specify an absolute path for the file, because *acct.restore* will have changed directories during processing.

You must use the no rewind tape device path. The input block size specified to *dd* must be larger than the largest physical block on the tape, otherwise data will be lost.

Using tapein to Restore Multireel Tapes

For restoring multireel MAGSAV and ACCOUNT-SAVE tapes, the system administration routines execute a UniVerse program named *tapein*. *tapein* can also be executed from a UNIX shell directly. The *tapein* program uses the standard UNIX filter *dd* to take data from an input device or file and pipe the data to *magrst* or *acct.restore*.

The *dd* filter properly handles end-of-media on most devices, eliminating many of the problems of multivolume tape handling. The *tapein* filter provides the necessary handshaking and signalling between *dd* and the *magrst* and *acct.restore* procedures. *tapein* prompts for the next volume of input and performs the necessary cleanup when the restoration procedures finish.

It is not necessary to use the *tapein* filter for diskettes.

The syntax for *tapein* is as follows:

```
tapein -prestore.filter -ffilename [options]
```

The following table describes each parameter of the syntax.

Parameter	Description
-prestore,filter	The full UNIX path for either <i>acct.restore</i> or <i>magrst</i> .
-filename	The input file name for the <i>if</i> option of the <i>dd</i> command. The input file name is the UNIX path of the input device or file.

tapein Parameters

Do not separate variables from the option codes by a space.

The following table describes the tapein options.

Option	Description
-apathname	Reads input from the specified UNIX path. <i>pathname</i> is the full UNIX path.
-bblocksize	The input block size for the <i>ib</i> option of the <i>dd</i> command. For 1/2-inch tape input, <i>blocksize</i> should be larger than the largest physical block on tape. For other devices the most likely value would be 512 bytes. The default is 8192, which should be correct for most 1/2-inch tapes.
-c	Specifies that <i>dd</i> should use the <i>conv=swab</i> option to swap data bytes on input.
-Ddevice	The path of a terminal (tty) device used when multiple instances of <i>acct.restore</i> are run.
-gpathname	The path of the rewind tape device.
-hfrhdr	The number of header files to skip over at the beginning of the first reel only. The default is 0. The value of this parameter should be 0 for <i>magrst</i> and REALITY ACCOUNT-SAVE tapes, and 2 for most Pick ACCOUNT-SAVE tapes.
-ice	Generates the input prompt character for use with the ic_execute function.

tapein Options

Option	Description
<i>-l</i> <i>labels</i>	The number of label records to skip over at the beginning of each reel. The default value is 0. <i>labels</i> is used as the value of the <i>skip</i> option of the <i>dd</i> command. For MAGSAV tapes, the value of this parameter should be 0. For the ACCOUNT-SAVE tapes, it depends on the type of media and the type of ACCOUNT-SAVE tape. For 1/2-inch ACCOUNT-SAVE tapes the value of this parameter should be 1. For most cartridge tapes, if the input block size is 512 and the ACCOUNT-SAVE tape has a tape label blocked at 512 characters, this parameter should be 1. If the ACCOUNT-SAVE tape is on a cartridge and the tape label is blocked at greater than 512 characters, then the input block size times the number of label records should equal the total number of characters in the tape label block. (For example, if the cartridge tape has a label blocked at 8192, an input block size of 512 skipping 16 label records, will work correctly.) This same formula also works with UNIX files containing a tape label.
<i>-m</i>	Restores data from a Microdata REALITY tape.
<i>-mas</i>	Restores data from a Microdata 7.0 M-A-S tape.
<i>-n</i>	Specifies that the separation is a factor of 2048.
<i>-rsrhdr</i>	The number of header files to skip over at the beginning of each subsequent reel for multireel input tapes. The default value is 0. The value of this parameter should be 1 for <i>magrst</i> tapes and should probably be 0 for Pick ACCOUNT-SAVE tapes.
<i>-t</i>	Specifies that <i>magrst</i> and <i>acct.restore</i> should create type 19 files instead of type 1 files. (See <i>UniVerse System Description</i> for information about type 1 and type 19 files.)
<i>-u</i>	Restores data using Ultimate cartridge format.

tapein Options (Continued)

The source for the *tapein* filter is in the file *sample/tapein.c* in the UV account directory. It can be modified to your specific needs. To compile a new version of *tapein*, save the original, make your changes, and compile as follows:

```
# cc tapein.c -o tapein
```

Because *tapein* is a user-modifiable program, it is not linked with the UniVerse library and does not trap the **Break** key in the same way as UniVerse. If you press the **Break** key, *tapein* tries to clean up all its subprocesses before exiting.

Restoring Accounts to Windows Systems

You can manually restore Pick or Prime INFORMATION accounts from tape to a Windows server.

If you are restoring a single-reel ACCOUNT-SAVE or MAGSAV tape, follow these steps:

1. Log in as a UniVerse Administrator.
2. Create a new directory to be the parent directory for the account.
3. Move to that directory.
4. Use the *acct.restore* and *magrst* executables. *acct.restore.exe* loads a Pick account and *magrst.exe* loads a Prime INFORMATION account. These executables create a UniVerse account in the current directory as well as subdirectories that contain the accounts loaded from the tape.

If you are restoring a multireel ACCOUNT-SAVE and MAGSAV tape, you need to use *tapein.exe*.

If you are restoring an account from an IN2 system, you need to use *uvmt.exe*.

Using acct.restore.exe and magrst.exe

The *acct.restore* executable transfers a Pick system ACCOUNT-SAVE tape to a UniVerse PICK flavor account on a Windows server.

The *magrst* executable transfers a Prime INFORMATION MAGSAV tape to a UniVerse INFORMATION flavor account on a Windows server.

These executables are in the *bin* directory of the UV account directory. You must run these executables from an MS-DOS window. They have the following syntax:

```
acct.restore.exe [options]  
magrst.exe [options]
```

The following table describes the options for the *acct.restore.exe* and *magrst.exe* executables.

Option	Description
–	Reads input from standard input.
–a <i>pathname</i>	Reads input from the disk file specified in the path.
–m	This option is available for <i>acct.restore.exe</i> only. Restores REALITY Microdata tapes.
–n	This option is available for <i>acct.restore.exe</i> only. Creates files with a default separation of 4 (each group buffer holds 2048 bytes). Use this option when you are transferring accounts from systems such as ADDS Mentor or PICK 370, where a separation of 1 means 2K bytes instead of 512 bytes.
–s	This option is available for <i>acct.restore.exe</i> only. Used with the <i>tapein</i> executable. Send the appropriate signals back to the <i>tapein</i> filter so <i>tapein</i> can prompt for the next volume of input and exit cleanly after <i>acct.restore.exe</i> or <i>magrst.exe</i> finishes.
–t <i>uv.device</i>	Reads input from a tape device defined in the &DEVICE& file. <i>uv.device</i> is the ID of the entry in the &DEVICE& file.
–19	Creates type 19 files instead of type 1 files.

acct.restore.exe and magrst.exe Options

For example:

```
d:\> magrst.exe -t MT0
```

If you use *acct.restore.exe* or *magrst.exe* without any arguments, the executable uses MT0 as the default tape device.

If *acct.restore.exe* is run by *tapein.exe*, it does not prompt the user when an error condition exists, such as unrecognized data or a file cannot be opened. It ignores the data or error and continues with the restoration.

If the MAGSAV tape is constructed on a PRIMOS environment of Release 20 or beyond, you must specify the –NO –ACL option of the MAGSAV command. This makes tapes that are compatible with Release 18. At Release 21, you must also specify the –REV19 option to create tape formats compatible with Release 19. UniVerse does not support tape formats compatible with Release 20 or later formats.

Using tapein.exe

The *tapein* executable is in the *bin* directory of the UV account directory. Use *tapein.exe* to restore multireel ACCOUNT-SAVE and MAGSAV tapes to a Windows platform.

tapein.exe takes data from a tape device and pipes the data to *acct.restore.exe* or *magrst.exe*. This program handles end-of-media on most devices, eliminating many of the problems of multivolume tape handling. *tapein.exe* prompts for the next volume of input and performs the necessary cleanup when the restoration procedures finish.

It is not necessary to use *tapein.exe* for diskettes.

This executable must be run from an MS-DOS window. It has the following syntax:

```
tapein.exe -prestorefilter -ffilename [-bblocksize] [-t] [-c] [-l] [-m]
[-n][-u] [-i] [-a] [-r] [-norew]
```

Do not separate variables from the option codes by a space.

The following table describes each parameter of the syntax.

Parameter	Description
<i>-prestore.filter</i>	The full Windows path for either <i>acct.restore.exe</i> or <i>magrst.exe</i> .
<i>-ffilename</i>	The input file name. This is the path of the tape device. For example: \\.\ tape0.
<i>-bblocksize</i>	The input block size. For 1/2-inch tape input, <i>blocksize</i> should be larger than the largest physical block on tape. For other devices the most likely value would be 512 bytes. The default is 8192, which should be correct for most 1/2-inch tapes.
<i>-t</i>	Specifies that <i>magrst.exe</i> and <i>acct.restore.exe</i> should create type 19 files instead of type 1 files. (See <i>UniVerse System Description</i> for information about type 1 and type 19 files.)
<i>-c</i>	Specifies the use of cartridge tape format.
<i>-l</i>	Activates logging to the <i>tapein.log</i> file.
<i>-m</i>	Specifies Microdata REALITY format.

tapein.exe Parameters

Parameter	Description
-n	Sets the separation as a factor of 2048.
-u	Specifies ultimate cartridge format.
-i	Specifies an IN2 type tape.
-a	Specifies that the input is an ASCII file (not a tape).
-r	Verifies the reel number.
-norew	Specifies not to rewind the tape after the restoration is complete.

tapein.exe Parameters (Continued)

For example:

```
d:\> tapein.exe -pd:\uv\uv\bin\magrst.exe -f\\.\tape0 -1
```

Using uvmt.exe

The *uvmt* executable is in the *bin* directory of the UV account directory. Use *uvmt.exe* to restore accounts from a tape created on an IN2 system to a Windows system.

You must run this executable from an MS-DOS window. It has the following syntax:

```
uvmt.exe -ddevicename [-bblocksize] [-c] [-t] {command}
```

Do not separate variables from the option codes by a space.

The following table describes each parameter of the syntax.

Parameter	Description
-ddevicename	Reads input from the tape device. For example: <code>\\.\tape0</code> .
-bblocksize	The input block size. For 1/2-inch tape input, <i>blocksize</i> should be larger than the largest physical block on tape. For other devices the most likely value is 512 bytes. The default is 512, which should be correct for most 1/2-inch tapes.
-c	Specifies that the device is a cartridge tape device. This is the default setting.
-t	Specifies that the device is a magnetic tape device.

uvmt.ext Parameters

The following table describes the *command* options.

Command	Description
rew	Rewinds the tape device.
bskip	Skips a tape block.
fskip	Skips a tape file (EOF skip).
bread	Reads a block from tape and writes it to standard out.
fread	Reads a file from tape and writes it to standard out.

uvmt.ext Commands

Restoring IN8000 Tapes.

To restore tapes created on an IN8000 machine (in SMA format):

1. Rewind the tape using the following command:

```
uvmt -d\\.\tape0 -b512 -c rew
```
2. Skip the first two files on tape using these commands:

```
uvmt -d\\.\tape0 -b512 -c fskip
uvmt -d\\.\tape0 -b512 -c fskip
```
3. Run the restoration process using *tapein.exe*:

```
tapein.exe -p$UVHOME\bin\acct.restore.exe -f\\.\tape0 -b512 -i
-c -r
```



***Note:** \$UVHOME represents the UV account directory. The device name shown is an example.*

If the tape contains multiple accounts, the *-norew* option should be included in the *tapein.exe* command line. This stops the tape from rewinding at the end of the first account restoration.

For example, for a tape with two accounts:

1. Restore the first account using these commands:

```
uvmt -d\\.\tape0 -b512 -c rew
uvmt -d\\.\tape0 -b512 -c fskip
uvmt -d\\.\tape0 -b512 -c fskip
tapein.exe -p$UVHOME\bin\acct.restore.exe -f\\.\tape0 -b512 -i
-c -r -norew
```


2. Restore the second account using these commands:

```
uvmt -d\\.\tape0 -b512 -c fskip  
tapein.exe -p$UVHOME\bin\acct.restore.exe -f\\.\tape0 -b512 -i  
-c -r
```

Restoring IN5000 Tapes

To restore tapes created on an IN5000 machine:

1. Rewind the tape using the following command:

```
uvmt -d\\.\tape0 -b512 -c rew
```

2. Skip the header files by using these commands:

```
uvmt -d\\.\tape0 -b512 -c fskip  
uvmt -d\\.\tape0 -b512 -c fskip
```

3. Run the restoration process using *INfilter.exe*. This executable is in the *bin* directory of the UV account. It filters data from an IN5000 tape by removing the header from each block before it is passed to *acct.restore.exe*. The pipe symbol, |, pipes the data between each program.

```
uvmt -d\\.\tape0 -b512 -c fread | INfilter.exe | acct.restore.exe  
-i -
```

For example, for a tape with two accounts:

1. Restore the first account using these commands:

```
uvmt -d\\.\tape0 -b512 -c rew  
uvmt -d\\.\tape0 -b512 -c fskip  
uvmt -d\\.\tape0 -b512 -c fskip  
uvmt -d\\.\tape0 -b512 -c fread | INfilter.exe | acct.restore.exe  
-i -
```

2. Restore the second account using these commands:

```
uvmt -d\\.\tape0 -b512 -c fskip  
uvmt -d\\.\tape0 -b512 -c fread | INfilter.exe | acct.restore.exe  
-i -
```

Transferring UniVerse Accounts from UNIX to Windows Platforms

This section describes how to transfer UniVerse accounts from a UNIX system to a Windows platform. You can transfer accounts using the *uvbackup* and *uvrestore* commands. This can be done using a tape device or with the FTP utility. The process for both is similar. It involves:

1. Creating the backup image using *uvbackup* (UNIX)
2. Transferring the backup image to the target (Windows) machine through tape or FTP
3. Restoring the backup image using *uvrestore* (Windows Platforms)

Note: The *uvbackup* and *uvrestore* commands must use the relative path in the syntax for the restoration of the accounts to work when restoring them onto a Windows system.



Creating the Backup Image

Before you create the backup image you should change to the directory containing the file or directory that you want to transfer. For example, if you have the account, */usr/account/my_account*, change to the directory */usr/account*.

You can use the *uvbackup* utility to create the backup image in a file or directly onto a tape. In both cases, use the UNIX *find* utility in the command line.

To back up the account *my_account* to a file, use the following command:

```
# find my_account -print | uvbackup -f -v - >backupfile
```

In this example the output is sent to a file called *backupfile*.

To back up the account *my_account* to tape, use the following command:

```
# find my_account -print | uvbackup -f -v - -t MT0
```

In this example the output is sent to a valid UniVerse tape device defined in the *&DEVICE&* file.

Transferring the Backup Image

If you save the backup image to a disk file, use FTP to transfer the file to the target machine. Since the disk file contains binary data, it is important to specify *binary* in your FTP utility before the transfer to ensure the data is not corrupted.

If you save the backup image to tape, put the tape in the tape drive of the target machine.

Restoring the Backup Image

Change to the directory where the account will be restored.

Use the *uvrestore* command to restore the backup. To restore an account backed up to a disk file, use the following command:

```
uvrestore -v backupfile
```

backupfile is the name of the disk file.

To restore a backup made to a tape device, use the following command:

```
uvrestore -v -t device
```

device is the name of a UniVerse tape device in the &DEVICE& file on the target system.

After the account is restored, it is a good idea to log in to the account and run the UPDATE.ACCOUNT command. This ensures that all VOC entries are updated. You should also recompile, and, if necessary, recatalog all UniVerse BASIC programs.

If the account contains applications that use absolute paths, they may not run correctly on Windows platforms, as these applications may be in different places. You may need to modify and recompile all such programs.

If the account relies on cataloged routines that are not part of the account, they may not be found on the target machine. You must transfer these separately, recompiling, and installing them in the correct location.

If the account relies on any database files outside the directory where the account resides, they are not found on the target system. These must be transferred separately and installed in the correct location.

File Naming Conventions

UniVerse reserves certain characters for its own use at the operating system level to allow users to type file names or record IDs in type 1 or type 19 files that would otherwise be rejected by the operating system. Normally it maps these characters to a replacement sequence. However, the list of reserved characters is different on UNIX and Windows platforms.

On UNIX systems:

This Character...	Maps to...
/	?\
?	??
empty filename	?0
. (leading period)	?.

Reserved Characters on UNIX

On Windows Platforms:

This Character...	Maps to...
/	%S
?	%Q
empty filename	%
"	%D
%	%%
*	%A
:	%C
<	%L
(vertical bar)	%V

Reserved Characters on Windows NT

This Character...	Maps to...
>	%G
\	%B
↑ (up-arrow)	↑↑ (up-arrow)
ASCII 1 through ASCII 26	↑A through ↑Z
ASCII 27 through ASCII 31	↑1 through ↑5

Reserved Characters on Windows NT (Continued)

This can create problems when you transfer files from UNIX to a Windows platform if any of the file names or record IDs in type 1 or type 19 files contain mapped characters. For example, the UNIX filename PERCENT% is PERCENT%% on Windows platforms.

UNIX System Security

Security Overview	8-4
User Permissions and File Permissions	8-4
File Permission Modes	8-6
Using the umask Command	8-6
Protecting User Accounts with Passwords	8-7
Assigning a Password	8-7
Making a Nonlogin Account.	8-8
Using Groups Effectively	8-9
Defining Groups	8-9
VOC File Security	8-11
Security Subroutines	8-12

UNIX System Security

On Windows platforms, you maintain system security using the Windows User Manager. On UNIX systems, you maintain system security using the **Accounts** option of UniAdmin and setting UNIX file permissions with the *umask* and *chmod* commands.

This chapter describes how to implement system security only on UNIX systems. For information about system security on Windows platforms, see your Microsoft Windows documentation.

Security Overview

Security on the UNIX operating system is configurable by installation and by user. Basic file protection is provided by mode information associated with each file when it is created. This mode information specifies permission to read, write, or execute the file. Permission is specified independently for the owner of the file, for members of the owner's group, and for all other users.

The user password and the file permission mask in the user's *.profile* file, *.login* file, or the UniVerse account's LOGIN entry are the chief mechanisms by which security is implemented.

In addition to the standard UNIX security mechanisms, security can be added to a UniVerse account. This includes editing the VOC file, restricting access to it, and controlling users' access to specific commands.

UniVerse SQL tables have their own security mechanism. UniVerse SQL security is described in *UniVerse SQL Administration for DBAs*.

User Permissions and File Permissions

It is important to distinguish between permissions that are set for a user and permissions that are set for a UniVerse account. When you use UniAdmin to change permissions on files in a UniVerse account, you actually change the permissions on the directory containing the UniVerse account and the permissions on all the files and subdirectories located in that directory.

File permissions on a user's files and directories are set when you add the user to the system. They are also determined when a UniVerse account is created. Default file permissions are set by the *umask* specification in a user's *.profile* file or in a UniVerse account's LOGIN entry. The *umask* specification sets permissions for all files and directories subsequently created by that user.

Users can set or change their own *umask* specification by editing the *.profile* file in their home directory. They can use the UMASK command in UniVerse. You can set or change file permissions on existing files and directories from a UNIX shell using the *chmod*(1) command.

Users can set or change the passwords for their own login accounts with either the UNIX command *passwd*(1) or the UniVerse command PASSWD.



Note: The root and uvadm accounts should be assigned passwords. The root password is important because root has essentially unlimited access to all system resources. An untrained user logged in as root can do a great deal of damage.

File Permission Modes

UniVerse file permissions are controlled by UNIX file permissions. Refer to your UNIX documentation for descriptions of UNIX file permissions.

Specifying file permissions for UniVerse account files does two things:

- It sets the permissions for all files in the directory containing the UniVerse account. It also sets the same permissions for all subdirectories (and their dependencies) in the account's directory.
- It sets the *umask* for the owner of the UniVerse account so that all files subsequently created in that account are given the permissions assigned to the owner of the account.

You usually set file permissions for a UniVerse account when you create it. You can also use UniVerse Admin to change file permissions on an existing account.

Do not change permissions on the files in the UV account directory.

Using the *umask* Command

Permission to access files created by a user is set by the file creation mask *umask*, specified in the user's *.profile* file or in a UniVerse account's LOGIN entry.

The UniVerse UMASK command performs the same function as the UNIX *umask* command.

The default *umask* is set to 022 (octal) so that only the owner can write to the file but all users can read it.

Protecting User Accounts with Passwords

The file permission modes can easily be circumvented if there is no mechanism to prevent one user from logging on under another user's name. UNIX uses a password mechanism to prevent unauthorized users from logging on to the system or to prevent one user from logging on to another user's account and gaining access to protected files. You can create or modify a password with either the UNIX *passwd* command or the UniVerse *PASSWD* command, which are described in the next sections.

Assigning a Password

You can assign a new password when you create the account, or you can let the user select a password. In either case the following description outlines the password assignment procedure.

Add a User	
File	Action
Logname	? █
User ID (UID)	:
Real Name	:
GID/Group name	:
Login Shell	:
Home Directory	:
Add Password (Y/N)	: No

Help Region
Press <F10> to activate the Menu Bar, <ESC> to Abort, Up Arrow to go up a field, Down arrow to go down a field, and <F1> for longer help about each prompt.
Enter the login name for the user you are creating.

You can assign passwords from either a UNIX shell or a UniVerse account.

Making a Nonlogin Account

To make an account that no one can log on to, you must be logged on as *root*. Use an editor to replace the current password encryption in the password file with a string to which no password encrypts. The most common string is an asterisk (`*`) or a string containing blanks such as `**NO LOGIN**`.

Using Groups Effectively

Passwords are a basic mechanism for restricting access to the system itself. File permission modes let you to define restrictions on who can see and use certain files. They also let you restrict access to UniVerse accounts. However, unless you define groups of users, file permission modes are of limited use. They allow a single user to have private files, but they do not allow certain files to be shared by a specified group of users. Groups allow you to take full advantage of the security system.

Group IDs let the members of one group protect their files on the basis of group membership—only members of the group can read and execute the files. You can give users of a UniVerse account different permissions for files in a group account and for their own files.

Defining Groups

Defining groups is a two-step process:

1. Define the group name and its associated group ID number.
2. Assign the group ID number to each user.

You can use the System Administration menus to define group names and ID numbers and to assign group ID numbers to users (see [Maintaining Users and User Groups](#) in Appendix A, “UniVerse System Administration Menus”).

Group permission is an important part of file system security in the UniVerse environment. Unlike standard UNIX accounts, which normally belong to individual users, UniVerse accounts are often used by a group of people performing the same function. It therefore may make sense for the users of the same UniVerse account to be defined as members of the same UNIX group.

One useful option is to create a set of related UNIX login accounts. For example, assume that you have a sales department whose members want to share the same UniVerse account. However, they would also like to have private files. You might make the following entries in */etc/passwd*:

```
sales::100:24:Sales dept:/u1/sales:/usr/ibm/uv/bin/uv
jim::101:24:Jim Hunter:/u1/sales:/usr/ibm/uv/bin/uv
markj::102:24:Mark Green:/u1/sales:/usr/ibm/uv/bin/uv
```

The following entry is added to */etc/group*:

```
sales::24:sales,jim,mark
```

Because Jim and Mark have the same home directory as *sales*, they have the same VOC file and hence the same UniVerse account. The default *umask* for the *sales* account should be set to 007 to allow access to the account's files to all members of the sales group.

Individual users could set the mode for their own private files to 600 by using a *umask* of 077, and restrict access to these files to themselves.



Note: *If you use the System Administration menus to change file permissions, ownership, or group ownership on a UniVerse account, the changes are applied to all files and subdirectories in the account directory.*

VOC File Security

In addition to the standard UNIX security mechanisms, you can exercise a degree of control over the actions of users who are logged on directly to the UniVerse environment by editing the contents of the VOC file for that account, and then assigning UNIX file permissions that prevent a user from writing the VOC file. Because the VOC file contains all the verbs and commands that a user can execute, you can keep a user from executing unwanted commands simply by removing them from the VOC file.



***Note:** If you want to create a secure installation in which only privileged users have access to the underlying UNIX operating system, you must define `/usr/ibm/uv/bin/uv` as the account's login shell. If `uv` is invoked from a `.profile` file, interrupting the execution of the `.profile` (for example, by pressing the **Break** key) leaves the user in a UNIX shell.*

Security Subroutines

VOC entries that point to remote items provide a mechanism for controlling access to certain commands. You can specify a user-supplied subroutine in field 4 of remote-type VOC entries. Such a subroutine sets a flag that permits or restricts access to the remote item. The UniVerse command processor checks the flag returned by the subroutine accessing the remote item.

A security subroutine must be set up for seven arguments.

```
SUBROUTINE security (remote, sentence, level, port, acct, log, flag)
```

security is the name of the subroutine. The first six arguments are passed to the subroutine by the command processor, and the last argument is a return argument. These arguments are briefly described in the following table.

Argument	Description
<i>remote</i>	The contents of the VOC entry for the remote item being executed.
<i>sentence</i>	The value of @SENTENCE (the command which invoked the remote item).
<i>level</i>	The following values are set: 0 Command processor 1 Execute 2 Execute of execute
<i>port</i>	User's port number
<i>acct</i>	Current account name
<i>log</i>	Login name of user
<i>flag</i>	The return flag: 1 Permit access 0 Restrict access

security Arguments

Upon return from the subroutine, the command processor checks the return flag. If the flag is set to 1, the command pointed to by the R-type VOC record is executed. If the flag is set to 0, access to the command is denied.

An effective way to use security subroutines is to put commands to which you want to restrict access in the UV account's VOCLIB file and put remote pointers to them in users' VOC files. Then add the appropriate security subroutines to the new remote pointers.

Managing Locks

Record Locks and File Locks	9-3
Shared Record Lock	9-4
Update Record Lock	9-5
Shared File Lock	9-6
Intent File Lock	9-6
Exclusive File Lock	9-7
Transactions and Locks	9-8
Managing Locks with UniVerse Admin.	9-9
File and Record Locks	9-10
Group Locks	9-11
Clearing Locks	9-12
Managing Deadlocks	9-13
Starting and Stopping the Deadlock Manager	9-15
Configuring Deadlock Management	9-17
Using the uvdlockd Command	9-18
Resolving Deadlocks Automatically	9-19

Certain UniVerse BASIC statements and UniVerse commands set locks on UniVerse files. The type of lock determines what a user or process can access while other users and processes have locks on records or files. You can monitor and clear UniVerse record and file locks with the UniAdmin **Locks** option, and you can resolve lock conflicts with the **Deadlocks** option.

Record Locks and File Locks

UniVerse record and file locks control access to records and files among concurrent user processes. To control access to records and files, UniVerse supports two levels of lock granularity:

- Fine granularity of record locks
- Coarse granularity of file locks

Granularity refers to the level at which a process or program acquires a lock. Record locks affect a smaller element, the record, and provide a fine level of granularity, whereas file locks affect a larger element, the file, and produce a coarse level of granularity.

Lock compatibility determines what a user's process can access while other processes have locks on records or files. Record locks allow more compatibility because they coexist with other record locks, thus allowing more transactions to take place concurrently. However these “finer-grained” locks provide a lower isolation level. File locks enforce a high isolation level, more concurrency control, but less compatibility. For information about transaction processing and isolation levels, see *UniVerse BASIC*.

Lock compatibility decreases and isolation level increases as strength and granularity increase. This may increase the possibility of deadlocks at high isolation levels. Within each granularity level, the strength of the lock can vary. UniVerse supports the following locks (in order of increasing strength):

- Shared record lock
- Update record lock
- Shared file lock
- Intent file lock
- Exclusive file lock

The locks become less compatible as the granularity, strength, and number of locks increase. Therefore the number of lock conflicts increase, and fewer users can access records and files concurrently. Weaker locks can always be *promoted* to stronger locks or *escalated* to a coarser level of granularity if needed.

Shared Record Lock

This lock is also called a READL lock, and is displayed as RL in the LIST.READU output. The shared record lock affects other users as follows:

Allows other users to acquire:	Prevents other users from acquiring:	Is ignored if the current user already owns:
Shared record lock	Update record lock	Shared record lock
Shared file lock	Exclusive file lock	Update record lock
Intent file lock		Shared file lock
		Intent file lock
		Exclusive file lock

How Shared Record Locks Affect Other Users

The shared record lock can be promoted or escalated as follows:

Promoted to...	If...
Update record lock	No shared record locks are owned by another user
	No shared file locks are owned by another user
	No intent file locks are owned by another user

How Shared Locks are Promoted

Escalated to...	If...
Shared file lock	No intent file locks are owned by another user
	No update record locks are owned by another user
Intent file lock	No intent file locks are owned by another user
	All update record locks are owned by the current user
Exclusive file lock	No intent file locks are owned by another user
	All shared and update record locks are owned by the current user

How Shared Locks are Escalated

Update Record Lock

This lock is also called a READU lock, and is displayed as RU in the LIST.READU output. The update record lock affects other users as follows:

Allows other users to acquire:	Prevents other users from acquiring:	Is ignored if the current user already owns:
No locks	Shared record lock	Update record lock
	Update record lock	Exclusive file lock
	Shared file lock	
	Intent file lock	
	Exclusive file lock	

How Updata Record Locks Affect Other Users

An update record lock is incompatible with a shared file lock owned by the same user.

The update record lock can be escalated as follows:

Escalated to...	If...
Intent file lock	All update record locks are owned by the current user
Exclusive file lock	All shared and update record locks are owned by the current user

How Update Record Locks are Escalated

Shared File Lock

This lock is displayed as FS in the LIST.READU output. The shared file lock affects other users as follows:

Allows other users to acquire:	Prevents other users from acquiring:	Is ignored if the current user already owns:
Shared record lock	Update record lock	Shared file lock
Shared file lock	Intent file lock	Intent file lock
	Exclusive file lock	Exclusive file lock

How Shared File Locks Affect Other Users

A shared file lock is incompatible with an update record lock owned by the same user. The shared file lock can be promoted as follows:

Promoted to...	If...
Intent file lock	No shared file locks are owned by another user
Exclusive file lock	No shared file locks or shared record locks are owned by another user

How Shared File Locks are Promoted

Intent File Lock

This lock is displayed as IX in the LIST.READU output. The intent file lock affects other processes as follows:

Allows other users to acquire:	Prevents other users from acquiring:	Is ignored if the current user already owns:
Shared record lock	Update record lock	Intent file lock
	Shared file lock	Exclusive file lock
	Intent file lock	
	Exclusive file lock	

How Intent File Locks Affect Other Processes

The intent file lock can be promoted as follows:

Promoted to...	If...
Exclusive file lock	No shared record locks are owned by another user

How Intent Files Locks Are Promoted

Exclusive File Lock

This lock is displayed as FX in the LIST.READU output. The exclusive file lock affects other users as follows:

Allows other users to acquire:	Prevents other users from acquiring:	Is ignored if the current user already owns:
No locks	Shared record lock Update record lock Shared file lock Intent file lock Exclusive file lock	Exclusive file lock

How Exclusive File Locks Affect Other Users

Transactions and Locks

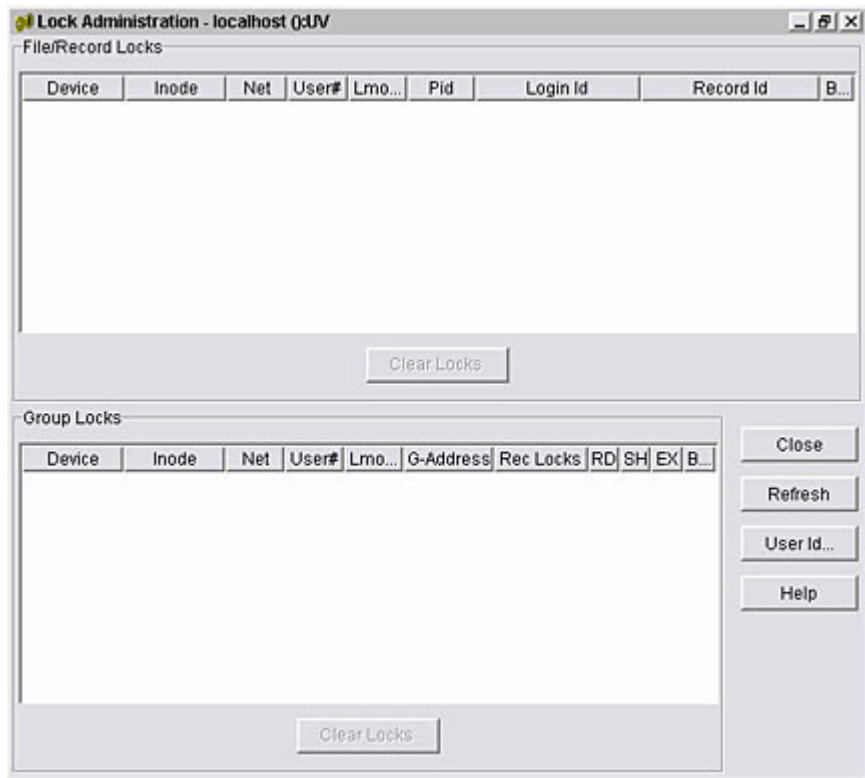
Locks acquired before a transaction exists, or outside an active transaction, are inherited by the active transaction. Locks acquired or promoted within a transaction are not released. Instead they adhere to the following behavior:

- Locks acquired or promoted within a nested transaction are adopted by the parent transaction when the nested transaction commits.
- Locks acquired within a nested transaction are released when the nested transaction rolls back.
- Locks promoted within a nested transaction are demoted to the level they were before the start of that transaction when the nested transaction rolls back.
- All locks acquired, promoted, or adopted from nested transactions are released when the top-level transaction commits or rolls back.

Managing Locks with UniVerse Admin

Locks are set on UniVerse files by certain BASIC statements and UniVerse commands. The type of lock determines what a process can access while other processes have locks on records or files. You can monitor and clear locks with the **Locks** option, and control the deadlock manager with the **Dead Locks** option.

To monitor and clear locks, choose **Locks** from the **UniAdmin** main window. The **Lock Administration** dialog box appears with a list of all the file, record, and group locks on the system:



The information displayed in the **Lock Administration** window is a snapshot of the record, file, and group locks when the **Locks** option was activated. To view the current state of locks, click **Refresh**.

The Lock Administration window is divided into two main areas:

- File/Record Locks
- Group Locks

File and Record Locks

The following information is in the File/Record Locks list:

Parameter	Description
Device	A number that identifies the logical partition of the disk where the file system is located.
Inode	A number that identifies the file that is being accessed.
Net	A number that identifies the host from which the lock originated. Zero (0) indicates a lock on the local machine.
User#	The user ID.
Lmode	<p>The lock semaphore number and the type of lock.</p> <p>For record locks, there are two settings:</p> <p>RU for an update lock</p> <p>RL for a shared lock</p> <p>For file locks, there are six settings:</p> <p>FS for a shared lock</p> <p>IX for a shared lock with intent to acquire an exclusive file lock</p> <p>FX for an exclusive file lock</p> <p>XU for an exclusive lock set by CLEAR.FILE</p> <p>CR for a shared file lock set by RESIZE</p> <p>XR for an exclusive file lock set by RESIZE or UVFIXFILE</p>
Pid	The process ID number.

File/Record Locks Parameters

Parameter	Description
Login Id	The login ID.
Record Id	The name of the record that is locked.
Bad	The bad lock indicator. If this column is empty, the lock is good. If this column contains an *, this indicates that the lock is bad.

File/Record Locks Parameters (Continued)

Group Locks

The following information is in the Group Locks list:

Parameter	Description
Device	A number that identifies the logical partition of the disk where the file system is located.
Inode	A number that identifies the file that is being accessed.
Net	A number that identifies the host from which the lock originated. Zero (0) indicates a lock on the local machine.
User#	The user ID.
Lmode	The lock semaphore number and the type of lock. There are five settings: EX for an exclusive update lock SH for a shared lock RD for a read lock WR for a write lock IN for an information lock
G-Address	The logical disk address of the group. This value is 1 for a type 1 or type 19 file. Any other value is represented in hexadecimal format.
Rec Locks	The number of locked records in the group.
RD	The number of readers in the group.

Group Lock Parameters

Parameter	Description
SH	The number of shared group locks.
EX	The number of exclusive update locks.
Bad	The bad lock indicator. If this column is empty, the lock is good. If this column contains an *, this indicates that the lock is bad.

Group Lock Parameters (Continued)

Clearing Locks

You can clear a single file, record, or group lock, or all the locks for a specified user using the **Lock Administration** window.

To clear a file or record lock:

1. Select the lock from the File/Record Locks list.
2. Click **Clear Lock**. The **Lock Administration** window is updated.

To clear a group lock:

1. Select the lock from the **Group Locks** list.
2. Click **Clear Group Lock**. The **Lock Administration** window is updated.

To clear all the locks for a specified user:

1. Click **User ID**. The **Clear User Locks** window appears.
2. Enter the user ID in the **User Id** field.
3. Click **OK**. The **Lock Administration** window is updated.

Managing Deadlocks

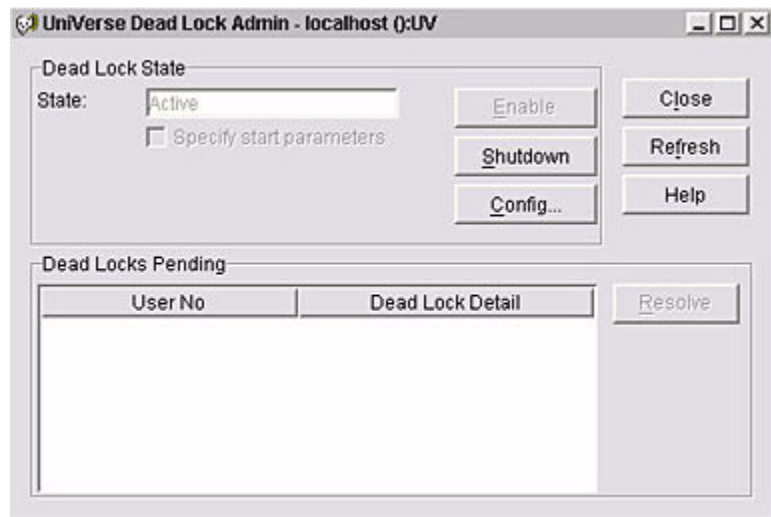
Deadlocks occur when one of several processes acquiring locks incrementally tries to acquire a lock that another process owns, and the existing lock is incompatible with the requested lock. Conditions such as the following can lead to deadlocks:

- Lock promotion from a shared record or shared file lock to a stronger lock
- Lock escalation to file locks when two processes try to escalate at the same time

You can configure UniVerse to automatically identify and resolve deadlocks as they occur, or you can manually fix a deadlock by selecting and aborting one of the deadlocked user processes. The deadlock daemon *uvdlockd* identifies and resolves deadlocks.

To start, stop, or configure the deadlock manager on the server, or to manually resolve file locking conflicts, choose **Dead Locks** from the **UniAdmin** main window. When the deadlock manager is running on the server, deadlocks are automatically resolved. The deadlock manager keeps a log file that records all deadlocks that it automatically resolved.

When you choose **Dead Locks** from the **UniAdmin** menu, the **UniVerse Dead Lock Admin** window appears, as shown in the following example:



The following information appears in the **UniVerse Dead Lock Administration** window:

Field	Description	Action
Dead Lock State	Indicates whether deadlocking is enabled or disabled, and whether the deadlock manager process is running.	Click Enable to turn on deadlock management with the default settings. Select Specify start parameters , then click Enable to configure deadlock resolution parameters. Click Shutdown to disable deadlock management. Click Config... to set deadlock resolution parameters. Click Refresh after changing the current management state.
Dead Locks Pending	Displays the server's current deadlock processes.	Select a process and click Resolve to terminate the process.

Dead Lock Administration Information

Starting and Stopping the Deadlock Manager

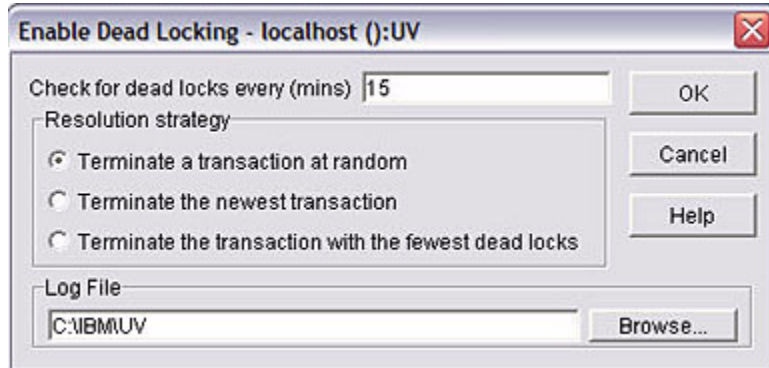
To start the deadlock manager on the server using system default settings, click **Enable**. Clicking **Shutdown** disables the deadlock manager.

***Note:** When the deadlock manager process is running, you cannot manually resolve deadlocks, and the **Resolve** button is grayed out. If you shut down the deadlock manager, click **Refresh** to select and resolve deadlocks displayed in the *Dead Locks Pending* box.*

To specify dead lock resolution parameters at startup time, select the **Specify start parameters** check box and click **Enable**.



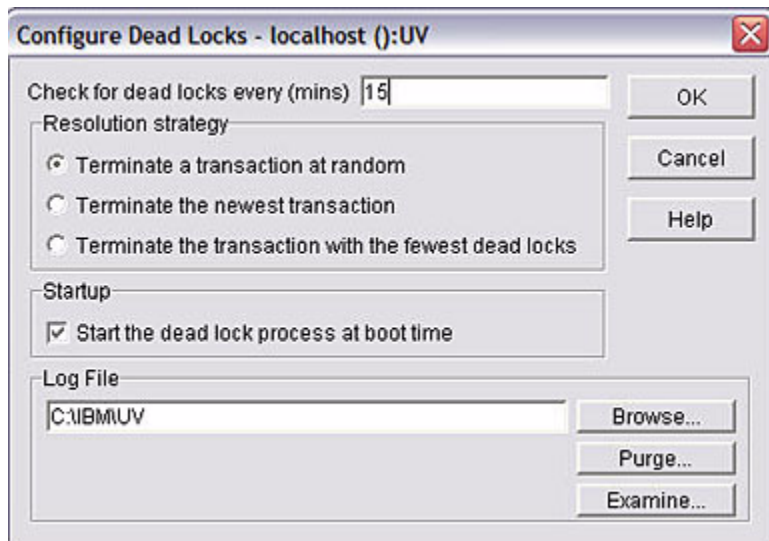
The **Enable Dead Locking** dialog box appears, as shown in the following example:



Configure dead lock resolution parameters as described in [Configuring Deadlock Management](#).

Configuring Deadlock Management

To configure deadlock management, click **Config** on the UniVerse **Dead Lock Admin** window. The **Configure Dead Locks** dialog box appears, as shown in the following example:



The following information appears in the **Configure Dead Locks** dialog box:

Field	Description	Action
Check for dead locks every (mins)	The number of minutes that the deadlock process waits before checking deadlock conditions.	Enter the number of minutes.
Resolution strategy	Defines how the deadlock process automatically resolves conflicts.	Click the option corresponding to the deadlock resolution strategy you want to use.
Startup	Indicates whether the deadlock manager starts when the server boots.	Select the check box to start the deadlock manager at server startup time.
Log Directory	Displays the full pathname of the log file. The log file displays information about deadlocks that have been automatically resolved and the way in which they were resolved.	Click Browse to select the directory for an existing log file. Click Purge to clear an existing log file to save disk space. Click Examine to view an existing log file.

Configure Dead Locks Fields

Options available in the dialog box change dynamically according to your choice to back up to disk or tape.

When you have completed your changes, click one of the following:

- **OK** implements your changes. The deadlock management process starts, and the **Dead Lock Administration** window appears.
- **Cancel** discards any changes. The **Dead Lock Administration** window appears.

Using the *uvdlockd* Command

You can also use the *uvdlockd* command from a UNIX shell or an MS-DOS window to administer the deadlock daemon. The syntax is as follows:

```
uvdlockd { [-t time] [-r resolution] [-l location] } | [-query] |  
[-stop] | [-v victim]
```

time is the time interval (in seconds) between the deadlock daemon's successive checks of the lock-waiter tables. The default is 60 seconds.

resolution is the resolution strategy the deadlock daemon uses. *resolution* is one of the following:

Value	Description
0	Selects a transaction at random. This is the default.
1	Selects the newest transaction.
2	Selects the transaction with the fewest number of locks held.

resolution Values

location is the location of the deadlock log file (the default is *uvhome/uvdlockd.log*).

-query generates a report based on a one-shot analysis of the lock-waiter tables and any detected deadlocks.

-stop shuts down the deadlock daemon.

victim specifies which user number to select as the process to abort.

If the deadlock daemon is not running, the *uvdlockd* command starts it.

Resolving Deadlocks Automatically

The deadlock daemon automatically resolves deadlocks by creating and updating a set of lock-waiter tables, which represent the state of the locking and transactional system. These tables are continually examined for evidence of a deadlock. Once the daemon detects a deadlock, it selects one of the currently active transactions to abort, removing the deadlock.

The deadlock daemon notifies the selected transaction that a deadlock has occurred and aborts the current execution layer. This rolls back any active transactional statements and cleans up any remaining locks.

UniVerse provides three automatic resolution strategies for removing deadlocks:

- Selecting a transaction at random
- Selecting the transaction with the fewest number of locks held
- Selecting the newest transaction

Selecting a random transaction works well in most situations. Selecting the transaction with the fewest locks or selecting the newest transaction work well when transactions are long. When UniVerse starts up, the system administrator determines which of these methods the deadlock daemon should use to remove deadlocks.

Configuring Peripheral Devices

The &DEVICE& File	10-3
Administering the &DEVICE& File	10-4
Configuring Tape Drives	10-5
Defining a New Tape Drive on a UNIX System.	10-5
Defining a New Tape Drive on a Windows Platform	10-9
Viewing and Modifying a Tape Drive Definition	10-12
Using the Test Tape Button	10-12
Deleting a Tape Drive Definition	10-12
Configuring Other Devices	10-13
Defining a New Device	10-13
Viewing and Modifying a Device Definition.	10-14
Deleting a Device Definition	10-14
Configuring Terminals on UNIX Systems	10-15
Terminal Line Naming Conventions	10-15
Setting Default Terminal Characteristics	10-16
The terminfo Facility	10-19
Customizing Terminal Capabilities While Logged On.	10-28
Mapping Terminals and Auxiliary Printers.	10-29

This chapter describes how to configure peripheral devices such as tape drives and terminals. It also helps with solving some of the problems that can occur with these peripherals.

Chapter 11, [“Administering Printers and the UniVerse Spooler”](#) describes how to configure printers.

The &DEVICE& File

You must define printers and tape drives in the &DEVICE& file for UniVerse applications to access them. The &DEVICE& file is in the UV account. It contains definitions for all devices defined in UniVerse. Each record in this file corresponds to a defined device.

Each device listed in the &DEVICE& file has a name, a brief description, a path, and options, such as lock files. The type of device (printer, tape drive, or other) is also indicated.

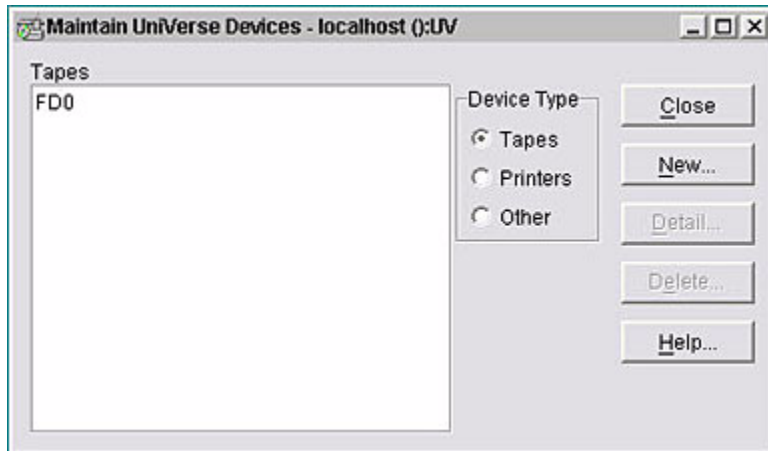
Here is a sample &DEVICE& entry that defines a tape device. Check the *UniVerse Installation Guide* to see the &DEVICE& file entry definition for your system.

```
DEVICE..... MT0
Description.... Magnetic tape drive number 0 (1600bpi)
UNIX pathname... /dev/rmt/2n
Block size..... 8192
Device type..... DT
Lock files.....
Rewind device... /dev/rmt/2
NoRewind device. /dev/rmt/2n
cpio-backup.... find $file -print | cpio -ovcB > $tape
cpio-restore.... cpio -ivcdumB $file < $tape
Skip file..... mt -f $tape fsf
Rewind tape..... mt -f $tape rewind
Tape offline.... mt -f $tape offline
Field 13..... A
Field 14..... y
Field 15..... y
Input Blksize... 16384
Rotate Flag.....
NLS Map Name....
```

Administering the &DEVICE& File

To administer the &DEVICE& file, choose **Devices** from the UniAdmin main menu. The **Maintain UniVerse Devices** dialog box appears, as shown in the following example. This dialog box displays (by default) the tape drives configured within UniVerse. The tasks you can perform from this window include:

- Configuring tape drives
- Configuring printers
- Configuring other devices



Configuring Tape Drives

Choose the **Devices** option from the UniAdmin main menu to add, modify, and delete tape drive definitions. The &DEVICE& file is automatically updated with your changes.

To define a tape drive, click **Tapes** in the **Device Type** area of the **Maintain UniVerse Devices** dialog box.

The **Tapes** list in the **Maintain UniVerse Devices** dialog box displays all the tape drives defined in the &DEVICE& file.

***Note:** How you define a new tape drive depends on whether you are connected to a UNIX or a Windows server.*



Defining a New Tape Drive on a UNIX System

To define a new tape drive:

1. Click **Tapes** in the **Maintain UniVerse Device** dialog box.

2. Click **New**. The Tape Details dialog box appears, as shown in the following example:

Tape Details - colt ():UV

Tape Name: Tape Type: DT (Dflt 9-track) ▼

Description:

Pathnames

No Rewind: Browse...

Rewind: Browse...

Lock File: Browse...

NLS Map: ▼

Read/Write Position: L (Load point) ▼

Flags

☒ Close On Read

☒ Multiple Read at EOF

☐ Add To Rotating File Pool

Block Size

Default:

Account Transfer: 8192

Shell Commands

Backup:

Restore:

Skip:

Rewind:

Offline:

OK Cancel Help

3. Enter the name of the new tape drive in the **Tape Name** field. This name is used in various UniVerse commands, such as ASSIGN, to refer to the device.
4. Select one of the following types from the Tape Type list:
 - DT (Dflt 9-track)
 - DC (Dflt cartridge)
 - T (9-track)
 - C (Cartridge)
 - F (Floppy)



Note: Multireel tape handling for the UniVerse *T.DUMP* and *T.LOAD* commands is supported only for device types DC, DT, and F.



5. (Optional) Enter a brief description of the tape drive in the **Description** field.
6. Enter a file path in the **No Rewind** field (for example, `/dev/rmt0n`). You can use **Browse** to search the system for a suitable file. A no-rewind tape drive does not rewind when closed.
7. Enter a file path in the **Rewind** field (for example, `/dev/rmt0`). You can use **Browse** to search the system for a suitable file. You cannot use the same path as the one for **No Rewind** option. A rewind tape drive rewinds when closed.

***Note:** Be sure to assign the correct access permissions to the device. You can set permissions for a device with the `UNIX chmod(1)` command.*

8. (Optional) Enter the name of a lock file in the **Lock File** field. You can use **Browse** to search the system for a suitable directory.

When a device is shared by UniVerse and UNIX system processes, it needs a special lock file that coordinates access to the device when more than one process tries to access it. This field contains the UNIX paths used to implement the locking protocol used by the Universe spooler and UNIX facilities such as the spooler and *uucp*. This field is usually empty for tape devices but can be used to display ownership information. For information about the form of the lock file name for a system, see the UNIX reference manual for the process that is sharing the device.

9. (Optional) If your system runs with NLS enabled, enter the name of a character set map for the device in the **NLS Map** field. For information about maps, see *UniVerse NLS Guide*.
10. (Optional) Click **Test Tape** if you want to run the tape device testing program to determine the following:
 - Where to allow the tape mode to change from read to write mode
 - What action to take when a tape file that is opened for read is closed
 - If a second read call at the end-of-file should return the end-of-file condition again

If you run the tests, their results are automatically filled in for you. If you do not run the tests, you can fill in your own values for these fields.



11. Select the setting for the read/write position from the Read/Write Position list. This specifies where on a tape a change from read to write mode is allowed.

- **L (Load point)** (This is the default.)
- **E** (Load point or EOF)
- **A (Anywhere)** (This usually works only on 1/2-inch tapes.)

This field is automatically updated with a suitable setting if you use the **Test Tape** button.

***Note:** Most Berkeley device drivers work with Read/Write Position set to A or E. Most System V device drivers work with Read/Write Position set to L or E.*

12. Set any of these flags by selecting the appropriate check boxes:

- **Close On Read.** This flag determines the action taken when a tape opened for reading is closed. If selected (the default setting), the tape moves forward to the beginning of the next file. If clear, the tape does not move forward. Most 1/4-inch tape devices use Close On Read. This field is automatically updated with a suitable setting if you use the **Test Tape** button.



***Note:** Most Berkeley device drivers work with this flag turned off. Most System V device drivers work with it turned on.*

- **Multiple Read at EOF.** This flag specifies the behavior of the tape when end-of-file (EOF) is reached. If selected (the default setting), the second read also returns EOF. If clear, the second read reads the next block/record after EOF. Most 1/4-inch tape devices use Multiple Read at EOF. This field is automatically updated with a suitable setting if you use the **Test Tape** button.



***Note:** Most Berkeley device drivers work with this flag turned off. Most System V device drivers work with it turned on.*

- **Add To Rotating File Pool.** If this flag is selected, the device is included in the rotating file pool. The default setting is clear.

13. Enter a block size in the **Default** field. This is the block size used for normal tape operations. It is needed only if the device is for cartridge tape (types DC and DT) or diskette (type F).

For diskettes the default block size is 500. Do not change this setting; any other block size can cause problems.

If the tape drive is a cartridge (C), this value must be a multiple of 512.

For nine-track tape (types T and DT) there is no default block size for IDEAL and INFORMATION flavor accounts: tape records are read or written with variable length. If this field is empty, the default block size for PICK or REALITY flavor accounts is 8192.

If the device is assigned using the ASSIGN command, the default block size is taken from this field in the &DEVICE& file. If the device is assigned using the T.ATT command, the default block size is taken from this field unless it is empty. In that case, the default block size is taken from the VOC entry for the T.ATT command.
14. Enter a value for the block size in the **Account Transfer** field. This value is used when importing accounts. You can accept the default of 8192 or enter a new value.
15. Enter UNIX shell commands in the **Backup, Restore, Skip, Rewind,** and **Offline** fields, if required. The **Backup** and **Restore** fields are automatically updated with a suitable setting if you use the **Test Tape** button.
16. Click **OK**. The new tape device is written to the &DEVICE& file. The **Maintain UniVerse Devices** dialog box is updated.

Defining a New Tape Drive on a Windows Platform

To define a new tape drive:

1. Click **Tapes** in the **Maintain UniVerse Devices** dialog box.

2. Click **New**. The **Tape Details** dialog box appears, as shown in the following example:

Tape Details - localhost (:):UV

Tape Name: Tape Type:

Description:

Device Pathname

Device:

NLS Map:

Read/Write Position:

Flags

☒ Close On Read

☒ Multiple Read at EOF

Block Size

Default:

Account Transfer:

DOS Commands

Skip:

Rewind:

3. Enter the name of the new tape drive in the **Tape Name** field.
4. Select one of the following types from the Tape Type list:
 - DT (Dflt 9-track)
 - DC (Dflt cartridge)
 - T (9-track)
 - C (Cartridge)
 - F (Floppy)
5. Enter a description of the tape drive in the **Description** field.
6. Enter the device pathname in the **Device** field, for example:
\\.\tape0.

7. Select the setting for the read/write position from the Read/Write Position list. This specifies where on a tape a change from read to write mode is allowed.

- **L (Load point)** (This is the default.)
- **E** (Load point or EOF)
- **A** (Anywhere)

This field is automatically updated with a suitable setting if you use the **Test Tape** button.

8. Select the check boxes to set any of these flags:

- **Close On Read.** This flag determines the action taken when a tape opened for reading is closed. If selected (the default setting) the tape moves forward to the beginning of the next file. If clear, the tape does not move forward. This field is automatically updated with a suitable setting if you use the **Test Tape** button.
- **Multiple Read at EOF.** This flag specifies the behavior of the tape when EOF is reached. If selected (the default setting), the second read also returns EOF. If clear, the second read reads the next block/record after EOF. This field is automatically updated with a suitable setting if you use the **Test Tape** button.

9. Enter a block size in the **Default** field. This is the block size used for normal tape operations. If the tape drive is a cartridge (C), this value must be a multiple of 512. A block size of 0 (variable) is normally used for DT and T type drives.

10. Enter a value for the block size in the **Account Transfer** field. This value is used when importing accounts. You can accept the default of 8192 or enter a new value.

11. Enter commands in the **Skip** and **Rewind** fields, if required. For example, you might enter the following in the **Skip** field:

uvmt -d\\.\tape0 -b512 fskip

(See [Using uvmt.exe](#) in Chapter 7, [Transferring Accounts](#) for information about the *uvmt* command.)

12. Click **OK**. The new tape drive is written to the &DEVICE& file. The **Maintain UniVerse Devices** dialog box is updated.

Viewing and Modifying a Tape Drive Definition

You can view and modify the details of any tape drive defined in the &DEVICE& file. To view a tape drive definition, do one of the following:

- Double-click the tape drive in the Tapes list.
- Choose the tape drive from the Tapes list and click **Detail**.

The **Tape Details** dialog box appears. You can modify any of the definition settings. Click **OK** to save any changes to the &DEVICE& file.

Using the Test Tape Button

If a tape drive has a tape type of DT or DC, the **Test Tape** button on the **Tape Details** dialog box is enabled. You can then use this button to fill in some of the tape drive definition details.

To start the tape tests, mount the tape and click the **Test Tape** button. Suitable settings are found for the following fields on the **Tape Details** dialog box:

- Read/Write Position
- Close On Read
- Multiple Read at EOF

***Note:** On a UNIX system, suitable entries are also determined for the **Backup** and **Restore** UNIX shell command fields.*

Settings found for these fields are automatically updated after the tests have been completed, overwriting any previous settings. Click **OK** to save the new definition.

Deleting a Tape Drive Definition

To delete a tape drive definition:

1. Select the tape drive from the **Tapes** list in the **Maintain UniVerse Devices** dialog box.
2. Click **Delete**. A message box appears.
3. Click **Yes**. The tape drive definition is removed from the &DEVICE& file. The **Maintain UniVerse Devices** dialog box is updated.



Configuring Other Devices

Choose the **Devices** option from the **UniAdmin** main window to add, modify, and delete devices other than tape drives and printers. The definitions for these devices are stored in the &DEVICE& file, with a device type of O.

To configure other devices, click **Other** in the **Maintain UniVerse Devices** dialog box.

The Others list in the **Maintain UniVerse Devices** dialog box is updated to display all the other devices defined in the &DEVICE& file.

Defining a New Device

To define a device that is not a tape drive or a printer:

1. Click **Other** in the **Maintain UniVerse Devices** dialog box.
2. Click **New**. The **Device Details** dialog box appears, as shown in the following example:

Device Details - localhost ():UV

Device Name:

Description:

Field #2:

Field #3:

Field #4:

Field #5:

Field #6:

Field #7:

Field #8:

Field #9:

Field #10:

Field #11:

Field #12:

Field #13:

Field #14:

Field #15:

Field #16:

Field #17:

Field #18:

Field #19:

Field #20:

OK Cancel Help

3. Enter the name of the device in the **Device Name** field.

4. Enter a description of the device in the **Description** field.
5. Enter an operating system path for the device in **Field #2**. Field 4 (the device type) is automatically set to O and cannot be edited. You need not put information in the other fields (Fields 3 and 5 through 20), as most of these fields apply only to tape or printer devices. These fields correspond to the field numbers in the &DEVICE& file. See Updating the &DEVICE& File for a description of the information these fields contain.
6. Click **OK**. The new device is written to the &DEVICE& file. The Maintain &DEVICE& File window is updated.

Viewing and Modifying a Device Definition

You can view and modify the details of any devices defined in the &DEVICE& file. To view a device definition, do one of the following:

- Double-click the device in the **Others** list.
- Select the device from the **Others** list and click **Detail**.

The **Device Details** dialog box appears. You can modify any of the definition settings. Click **OK** to save any changes to the &DEVICE& file.

Deleting a Device Definition

To delete a device:

1. Select the device from the Others list in the **Maintain UniVerse Devices** dialog box.
2. Click **Delete**. A message box appears.
3. Click **Yes**. The device definition is removed from the &DEVICE& file. The **Maintain UniVerse Devices** dialog box is updated.

Configuring Terminals on UNIX Systems

You must administer and maintain terminals entirely from the UNIX environment. The UNIX operating system supports many different terminals, ranging from teleprinter terminals to CRTs with sophisticated graphics capabilities. Many programs use different terminal characteristics to do their jobs. Some programs use only the most basic terminal features. Other programs, such as *vi*, require a detailed knowledge of the terminal's capabilities. A knowledge of the kind of terminal being used is important.

UNIX maintains a database of many different terminal types. Each type has an entry in the database that is identified by the manufacturer's name and the product number. The entry lists the control codes used to access specific features of the terminal's capabilities. The system usually specifies the terminal type automatically during the login process by setting an environment variable called `TERM` in the user's *.profile* file. `TERM` identifies the name of a particular entry in the terminal database.

In addition to the terminal type, you must define a number of serial line characteristics (such as baud rate and parity) in order for a terminal to work properly with the system.

The type of terminal attached to each serial line and the line characteristics are determined automatically when someone logs in, assuming you have properly identified the terminals associated with each line in system startup files.

Terminal Line Naming Conventions

UNIX treats a terminal (and any other device) as a special kind of file, with a file name and a location in the UNIX file system hierarchy. These files are located in the */dev* directory. The files associated with terminal lines have names of the form */dev/ttyxxx*. For example, the special file associated with terminal line 1 might have the special file name */dev/tty001*.

In the UNIX environment the *tty* command tells you to what file the current terminal is attached to, as shown in the following example:

```
# tty  
/dev/tty001
```

In UniVerse you can use the `STATUS` command with the `USERS` option to get this information.

Your system is shipped with a number of terminals already defined. You should have a special file in */dev* for each of the terminals. The procedures for adding a special file for a terminal line are outlined in the UNIX system administrator's manual supplied with your system.

Setting Default Terminal Characteristics

On most UNIX systems the *init-getty-login* sequence sets up a terminal for logging in and defines certain default characteristics of the terminal line (such as the speed) and the TERM variable in the environment. This sequence has four parts:

1. The *init* process reads a file called */etc/inittab* that specifies which terminal lines should have a *getty* created for them. Each entry in */etc/inittab* contains a flag that points to a definition in another file, */etc/gettydefs*.
2. The *getty* process reads */etc/gettydefs*, which specifies the speed of the terminal line and a minimal set of terminal characteristics. *getty* displays a login prompt on the terminal screen.
3. When a user responds to the login prompt by logging in, the *login* process begins, and the terminal type associated with the line is determined. Finally the directory */usr/lib/terminfo* is searched. This is the master database of terminal capability definitions.
4. You may need to modify */etc/inittab* if you add terminals to the system or if you want to disable unused terminal lines so that the system does not create *getty* processes for them. You also need to change this file if you want to use a terminal that operates at a speed other than 9600 baud.

If you want to use a terminal not contained in the *terminfo* database you need to add a definition to the database.

The following sections look at the format of each of these files and describe the changes you may need to make to them.

Note: The procedures for defining new terminals may differ from system to system. See your UNIX system administration manuals for detailed instructions.



The */etc/inittab* File

On most UNIX systems the file */etc/inittab* contains an entry for each terminal line on the system. Each entry has the following format:

```
ID:level:type:process
```

The following table describes the */etc/inittab* format.

Parameter	Description
<i>ID</i>	A one- to four-character identifier used by <i>init</i> internally to label entries in its process table. This identifier generally corresponds to the name of a terminal line.
<i>level</i>	Specifies at what levels <i>init</i> should be concerned with this entry. Any time <i>init</i> 's internal level matches <i>level</i> , this entry is <i>active</i> . If <i>init</i> 's internal level does not match any of the levels specified, then <i>init</i> makes certain that the process is not running. The importance of different levels for <i>init</i> is described in the system administrator's guide for your system.
<i>type</i>	Specifies some further condition required for or by the execution of an entry. This field should be one of the following: <i>off</i> , <i>once</i> , <i>wait</i> , <i>respawn</i> , <i>boot</i> , <i>bootwait</i> , <i>power</i> , <i>powerwait</i> , or <i>initdefault</i> . See the system administrator's guide supplied with your system for a detailed description of these options.
<i>process</i>	The action that <i>init</i> asks a shell (<i>sh</i>) to perform whenever the entry is activated.

/etc/inittab Format

For example, the entry in */etc/inittab* for */dev/tty001* might read:

```
001:2:respawn:/etc/getty tty001 9600
```

There are three possible changes that you may want to make to this file:

- You may want to disable terminal lines that are not being used. This keeps the system from creating a *getty* for that process and saves memory. For example, if your system has four terminal lines, but you have only three terminals installed, you might disable one of the terminal lines by changing the entry in the *type* field of */etc/inittab* to *off*. You may also want to disable *gettys* on lines that do not require a login (for example, lines associated with serial printers, or modems that are used only to dial out).

- You may want to change the speed associated with a line. For example, you might make an entry for a 1200 baud terminal by changing the *9600* to *1200* in the *process* field of the */etc/inittab* file.
- You may want to change the *level* field. This field determines whether the current entry is active based on the level at which *init* is running. *level* is a string of one or more characters. Any number from 0 through 6, and the letters a, b, c, and r can be used in the *level* field. If the *level* field is empty, it is equivalent to the string 0123456. See the system administrator's guide supplied with your system for more details about the *level* field.

The /etc/gettydefs File

When *getty* is invoked, it references */etc/gettydefs* to determine how to set up the line. Each entry in *gettydefs* has the following format:

```
label#initial flags#final flags#login message#next label
```

For a hypothetical terminal attached to */dev/tty001*, the following entry would be read:

```
9600# B9600 CLOCAL BRKINT IGNPAR ISTRIP IXON IXOFF ECHO OPOST\
ONLCR #BRKINT ISTRIP ICRNL IXON OPOST ONLCR B9600 CS8 CREAD \ ISIG
ICANON ECHO ECHOE ECHOK IXANY TAB3 #IBM Corporation.\ UNIX version
V #1200
```

Fields in each entry are separated by the hash sign (#).

The *label* field usually contains the speed setting of the terminal. When *getty* reads */etc/gettydefs*, it checks the speed and stops at an entry whose *speed* field matches what it determines to be the speed of the device attempting to log on.

The *initial flags* field allows the *getty* to accept the login. It should be a list of basic terminal characteristics.

The *final flags* field entries restore the terminal to basic terminal characteristics after the logon session.

The *login message* field gives the message that *getty* prints while it is waiting for a user to enter a logon name. You may want to customize this field for your particular system.

The *next label* field provides a link to the next entry for *getty* to search if *getty* finds that the current speed being tried is wrong.

In general, you should not have to modify the entries in */etc/gettydefs*, with the exception of the *login message* field. For further details about the */etc/gettydefs* file, see the system administrator's guide supplied with your system.

Setting the TERM Variable

Once *getty* has opened the line with the characteristics specified in */etc/gettydefs*, the *TERM* variable must be set to establish the kind of terminal attached to the terminal line. This is done in the file *.profile*. The following example shows a Wyse 50 terminal attached to the terminal line:

```
TERM=wy50
```



Note: On some UNIX systems *login* reads a file called */etc/ttytype* to determine what kind of terminal is attached to a given terminal line and sets up the *TERM* variable accordingly. The file contains two columns, the first lists the terminal type and the second lists the terminal line name. On other systems, the default terminal type may be specified in the */etc/inittab* file.

The *tty* command prints the line to which the terminal is attached.

The terminfo Facility

The UNIX *terminfo* facility lets you build efficient, portable, and hardware-independent routines in your programs. *terminfo* comprises a database of terminal capabilities and the library routines that provide access to this database. Each supported terminal's capabilities are defined in a separate file that is compiled into binary format for efficiency.

terminfo files enable programs to send appropriate escape sequences to a terminal by referencing functional descriptions of the terminal's capabilities. At execution time the escape sequence needed for a particular capability is taken from the *terminfo* definition for the terminal. On input, a program can determine what function key or control key has been entered by comparing the input sequence against the *terminfo* definition.

UniVerse provides extensions to the BASIC language that let you use standard UNIX *terminfo* capabilities and terminal capabilities available only to UniVerse applications. UniVerse also provides tools that simplify the creation and modification of terminal definitions.

The terminfo Database

Unlike many proprietary systems that support only one brand of terminal, UNIX supports many terminal types. Most UNIX *terminfo* databases contain descriptions of over 400 terminals. Each file in the database contains one terminal's capabilities, such as the number of columns and rows supported, the escape sequences used to manipulate the terminal, and the character sequences generated by pressing function keys and cursor control keys. To use the *terminfo* database, you must program your application to get the terminal capabilities from the *terminfo* file named by an environment variable. This way the program is bound to the terminal type at execution time.

UniVerse maintains the UniVerse enhancements to *terminfo* in a local *terminfo* database in the UV account directory (such as */usr/ibm/uv*). Standard UNIX capabilities are maintained in the UNIX *terminfo* database, usually in */usr/lib/terminfo*. During UniVerse installation, the *terminfo* source file *terminfo.src*, located in the sample directory, can be compiled to generate the UniVerse *terminfo* database in the UV account directory. Most of the entries in the *terminfo.src* file are similar to the UNIX entries, but they also include terminal capabilities specific to UniVerse.

Each *terminfo* source file contains a group of names the terminal is known by, separated by vertical bars (|). Following the list of names is a list of comma-separated fields describing the terminal's actions, capabilities, and definitions. These include the number of columns, special escape sequences for cursor positioning, highlighting, intercharacter delays, and other terminal-specific features. For a complete discussion of these terminal capabilities, see your UNIX documentation (look at *terminfo*(4)), or the Nutshell Handbook, *termcap & terminfo*, available from O'Reilly & Associates, Inc. Here is an example of the first line of an entry containing the terminal definition names:

```
vt100|vt100-am|dec vt100,
```

All but the last terminal type name (dec vt100) are names of the *terminfo* database files containing the description. The naming convention for a terminal definition that is a variation of a base definition is a suffix appended to the base name with a hyphen (such as, vt100-132 for a 132-column vt100). The definition of a terminal emulation has the name of the original terminal followed by a colon and the name of the emulation (such as wy50:vp for a Wyse Technology 50 emulating Adds Viewpoint).

UNIX and UniVerse *terminfo* directories contain subdirectories for each number (1 through 9) and for the letters of the alphabet. Each subdirectory contains the *terminfo* definition files that start with its letter or number. When you compile the *terminfo* source file for the vt100 terminal definition mentioned in the previous example, four files are created (if the *-a* option to *uvtic* is used):

```
/usr/lib/terminfo/v/vt100  
/usr/lib/terminfo/v/vt100-am  
/usr/ibm/uv/terminfo/v/vt100  
/usr/ibm/uv/terminfo/v/vt100-am
```

In both directories, the vt100 and vt100-am files are linked so there are not two copies of the data.

Creating or Modifying a terminfo Entry

To create or modify a *terminfo* entry:

1. Create a source file containing the definition of a terminal type close to the one that you are creating.
2. Change the name of the terminal definition in the source file.
3. Use *vi* or another UNIX editor to make changes to the capabilities.
4. Keep in mind that some UNIX systems support *termcap*. You need to modify */etc/termcap* file to reflect the changes in *terminfo*.
5. Use the UniVerse *terminfo* compiler to generate the new binary definition.
6. Set your terminal type to the new value and test it.
7. Make your new source file part of the standard file you recompile after any system upgrade.

Defining and Enhancing terminfo Capabilities

The first step in defining a new *terminfo* entry or enhancing an existing one is to read the owner’s manual for your terminal. The manual should describe all the character strings that make up transmitted sequences from your keyboard and all the escape sequences for manipulating the display. Here is an example of a standard UNIX vt100 *terminfo* source definition:

```
vt100|vt100-am|dec vt100,
am, xenl, xon, cols#80, it#8, lines#24, vt#3,
bel=^G, blink=\E[5m, bold=\E[1m, clear=\E[H\E[J,
cr=\r, csnm=\E[5m, csr=\E[%i%p1%d;%p2%dr,
cub=\E[%p1%dD, cub1=•, cud=\E[%p1%dB, cud1=\n,
cuf=\E[%p1%DC, cuf1=\E[C, cup=\E[%i%p1%d;%p2%dH,
cuu=\E[%p1%dA, cuu1=\E[A, docr=\E[J, ed=\E[J, el=\E[K,
home=\E[H, ht=, hts=\EH, ind=\n, initc=31, kbs=\b,
kcub1=\EOD, kcud1=\EOB, kcufl1=\EOC, kcuu1=\EOA,
kfl1=\EOP, kf2=\EOQ, kf3=\EOR, kf4=\EOS, oc=10;13,
rc=\E8, rev=\E[7m, ri=\EM, rmkx=\E[?11\E>, rmso=\E[m,
rmul=\E[m, rs2=\E>\E[?31\E[?41\E[?51\E[?7h\E[?8h,
sc=\E7,
sgr=\E[%?%p1%t;7%;%?%p2%t;4%;%?%p3%t;7%;%?%p4%t;5%;%?%p6%t;1%;m,
sgr0=\E[m, smkx=\E[?1h\E=, smso=\E[7m, smul=\E[4m,
subcs=\E[H\E[J, supcs=\E[H, tbc=\E[3g, zerom=\E[K,
```

Capabilities defined in a *terminfo* source file are divided into three classes:

- Numeric
- Boolean
- String capabilities (strings and parameterized strings)

For more information about *terminfo* capabilities, with examples, see Appendix C, [“terminfo Terminal Capabilities,”](#)

Modifying a terminfo Entry

Whether you are creating a new *terminfo* definition or enhancing an existing one, start by copying an existing *terminfo* definition that is close to the one you are trying to create, then test each capability as you modify the definition.

You can use the UniVerse *terminfo* decompiler *uvtidc* to produce a source file as a basis for your new *terminfo* definition. For example, to create a variation of the vt100 *terminfo* definition, first capture the source output of *uvtidc* in a file. In this example, the file is put in the UniVerse *sample* directory, but it can go anywhere:

```
# cd /usr/ibm/uv/sample
# /usr/ibm/uv/bin/uvtidc vt100 >my.vt100
```

After this command finishes, the file */usr/ibm/uv/sample/my.vt100* contains:

```
vt100|vt100-am|dec vt100,
am, xenl, mir, msgr, xon, cols#80, it#8,
lines#24, vt#3, bel=^G, cr=/r, csr=\E[%i%p1%d;%p2%dr,
tbc=\E[3g, clear=\E[H\E[J$<50>, el=\E[K$<3>, ed=\E[J$<50>,
cup=\E[%i%p1%d;%p2%dH$<5>, cudl=\n, home=\E[H, cub1=.,
cuf1=\E[C$<2>, cuu1=\E[A$<2>, smacs=^N, blink=\E[5m$<2>,
bold=\E[1m$<2>, rev=\E[7m$<2>, smso=\E[1;7m$<2>, smul=\E[4m$<2>,
rmacs=^O, sgr0=\E[m^O$<2>, rmso=\E[m$<2>, rmul=\E[m$<2>,
kbs=., kcu1=\EOB, kf0=\EOy, kf1=\EOP, kf10=\EOx, kf2=\EOQ,
kf3=\EOR, kf4=\EOS, kf5=\EOt, kf6=\EOu, kf7=\EOv, kf8=\EOl,
kf9=\EOw, kcub1=\EOD, kcuf1=\EOC, kcuu1=\EOA, cud=\E[%p1%dB,
cub=\E[%p1%dD, cuf=\E[%p1%DC, cuu=\E[%p1%DA,
rs2=\E>\E[?31\E[?41\E[?51\E[?7h\E[?8h, rc=\E8, sc=\E7, ind=\n,
ri=\EM$<5>,
sgr=\E[0%?%p1%p6%|t;1%;%?%p2%t;4%;%?%p1%p3%|t;7%;%?%p4%t;5%;m%?%p9%t^
N%e^O%;,
hts=\EH, ht=, kal=\EOq, ka3=\EOs, kb2=\EOr, kc1=\EOp, kc3=\EOn,
acsc='`aaffggjjkkllmmnnnooppqrrssttuuvvwxxyyzz{|}|}~~,
enacs=\E(B\E)0, kent=\EOM, ell=\E[1K$<3>,
#! uv kexit=10;13, kebs=8, at1=\E[H\E[J, at2=\E[H, at3=\E[J,
#! uv at4=\E[K, at5=\E[5m, at6=\E[m, at8=\E[m^O$<2>, at9=.,
#! uv at10=\E[A, at12=\E[m^O$<2>, at13=\E[7m, at14=\E[m,
at15=\E[4m,
#! uv at16=\E[m, at17=\E[1m, at18=\E[m, at20=\E[m^O$<2>
```

This is the *terminfo* definition for a vt100 terminal with two associated names, vt100 and vt100-am. You should first change the name line so that any modifications you make do not overwrite the original definitions. To change the name, modify the first line in the definition so that it has a unique name, and eliminate any aliases:

```
myvt100|Testing new terminfo for vt100,
```

If you are creating a *terminfo* definition that does not match your base definition, make sure all the capabilities you are not going to change match the capabilities of the terminal you are defining. Do this by checking the terminal's manual against the definitions, or by testing the capabilities themselves.

Compiling terminfo Definitions

Use compilers *tic* (UNIX) and *uvtic* (UniVerse) to compile *terminfo* database entries from source into compiled format. UNIX *tic* compiles only the UNIX *terminfo* database, whereas UniVerse *uvtic* compiles UNIX and UniVerse. When you install UniVerse, you can specify that the *terminfo* entries shipped with UniVerse should be compiled. If you choose this option, only the UniVerse capabilities are compiled. To update the standard UNIX capabilities with those defined in the *terminfo* entries shipped with UniVerse, you must explicitly compile the *terminfo.src* file in the sample directory, using the *-a* option to *uvtic*.

uvtic without the *-a* option compiles only the UniVerse capabilities into the UniVerse *terminfo* directory located in the UV account directory. If you specify the *-a* option, *uvtic* compiles the UNIX capabilities into the */usr/lib/terminfo* and compiles the UniVerse capabilities into the UniVerse *terminfo* directory. The source files are compiled into standard binary file format.

Updating the Whole terminfo Directory

To update your *terminfo* database:

1. Save all current *terminfo* entries for terminals not supported by UniVerse. Only terminals that are not supported by UniVerse should be saved.

```
# cd /usr/ibm/uv/sample
# uvtidc terminal.name >> terminfo.src
```
2. Remove all remaining terminal definitions and set permissions for the *terminfo* directory to 755:

```
# rm -rf /usr/ibm/uv/terminfo/*
# chmod 755 /usr/ibm/uv/terminfo
```
3. Recompile the UniVerse terminal definitions. You should be logged on as a UniVerse Administrator to execute this command:

```
# uvtic -v /usr/ibm/uv/sample/terminfo.src
```
4. Use the *tiindex* command to rebuild the *terminfo* index file:

```
# tiindex
```

The *tiindex* command lists all terminal names available to the SET.TERM.TYPE UniVerse command. You must rebuild this index file when you add or delete a terminal. You should be logged in as a UniVerse Administrator to use *tiindex*.

Keep in mind that some UNIX systems support *termcap*. If your system is one of these, you need to modify the */etc/termcap* file to reflect the changes in your new *terminfo*.

Adding or Changing a Single Entry

To compile and test the new vt100 *terminfo* definition, you need to compile the UNIX and the UniVerse capabilities. The *-v* option to *uvtic* allows error messages to be displayed to standard error. If you do not specify *-v*, unsupported capabilities are ignored. You should be logged in as a UniVerse Administrator to execute this command:

```
# /usr/ibm/uv/bin/uvtic -v -a myvt100
```

Once you successfully compile the *terminfo* entry, test it in UniVerse and non-UniVerse environments. The best test for the non-UniVerse environment is usually the *vi* editor. A good test in the UniVerse environment is the system administration routines.

To test the entry in the example, do the following:

```
# TERM=myvt100;export TERM
# vi myvt100
# cd /usr/ibm/uv
# bin/uv
```

In the UniVerse environment you can set the terminal type to *myvt100* by using the SET.TERM.TYPE or TERM command.

Decompiling terminfo Entries

One reason for decompiling a *terminfo* entry is to add UniVerse enhancements to an existing UNIX *terminfo* entry. If UniVerse BASIC programs or the system administration routines do not clear the screen or present a meaningful display, you may have a *terminfo* entry that lacks the necessary UniVerse enhancements. The UniVerse *terminfo* decompiler can add these enhancements for subsequent compiling into the UniVerse *terminfo* database. For example, if you have a UNIX *terminfo* entry named svt100, you can add UniVerse capabilities with the following sequence, which uses the *-i* option to generate source for UniVerse capabilities:

```
# cd /usr/ibm/uv/sample
# /usr/ibm/uv/bin/uvtidc -i vt100 > svt100.src
# /usr/ibm/uv/bin/uvtic -v svt100.src
```

If you have several terminal definitions, or if you have enhancements to several definitions, you may want to consolidate the modified definitions with the definitions shipped with UniVerse (in the *terminfo.src* file in the sample directory), so you can install everything with one compile. The following example shows how to combine several source terminal definitions (*my.vt100* and *svt100.src*) with the default UniVerse *terminfo* source file (*/usr/ibm/uv/sample/terminfo.src*), update all the UNIX and UniVerse binaries, and rebuild the index of available terminal types for the SET.TERM.TYPE command:

```
# cd /usr/ibm/uv/sample
# cat terminfo.src my.vt100 svt100.src terminfo.nsrc
# /usr/ibm/uv/bin/uvtic -v -a terminfo.nsrc
# /usr/ibm/uv/bin/tiindex
```

Making Non-UniVerse terminfo Entries Compatible with UniVerse

Decompile the entry with the *-u* option, then recompile it.

```
# uvtidc terminal.name | uvtic -u
```

Making All terminfo Entries Work Properly with UniVerse

Decompile the whole *terminfo* directory, then recompile it.

```
# uvtidc `cut -f1 /usr/lib/terminfo/index` | uvtic -v
```

The terminfo Utilities

On System V systems, in addition to the *terminfo* compiler *tic*, you can use the *infocmp* utility to compare and print *terminfo* definitions. *infocmp* does not display the UniVerse enhancements to *terminfo*, but it is very useful in comparing *terminfo* and termcap entries. Here are some of the options used by *infocmp*:

Option	Description
-d	Lists all capabilities that are different in both entries.
-c	Lists all capabilities that are the same in both entries. Unset capabilities are ignored.
-n	Lists capabilities that are in neither entry. Use this as a quick check to see if anything was left out of the description.

infocmp Options

The following command compares a vt100 to a wy50:

```
$ infocmp -d vt100 wy50
```

This command produces the following output, which lists only the capabilities that are different:

```
comparing vt100 to wy50.
comparing booleans.
bw: F:T. (where bw is false (F) for vt100 and TRUE (T) for wy50)
hs: F:T.
mir: F:T.
enl: T:F.
.
.
.
csr: '\E[%i%p1%d;%p2%dr', 'NULL'.
cub: '\E[%p1%dD', 'NULL'.
.
.
.
kcud1: '\E[B', '\n'.
kcuf1: '\E[C', '\f'.
kcuu1: '\E[A', '^K'.
.
.
.
smacs: 'NULL', '\EH^B'.
smir: 'NULL', '\Eq'.
smkx: '\E[?1h\E=', 'NULL'.
smso: '\E[7m', '\EGt'.
smul: '\E[4m', '\EG8'.
tbc: '\E[3g', '\E0'.
tsl: 'NULL', '\Ez('.
```

On Berkeley systems, use *uvtdc* to get similar output.

UniVerse terminfo Requirements

The UniVerse System Administration menu requires the following *terminfo* characteristics in addition to the standard UNIX *terminfo* characteristics. These required capabilities also apply to other IBM products. Appendix C, “[terminfo Terminal Capabilities](#),” lists terminal capabilities.

Customizing Terminal Capabilities While Logged On

The UNIX *stty* command and the UniVerse PTERM (UNIX) command lets you set terminal characteristics while you are logged on. Chapter 6, “[UniVerse Accounts](#),” discusses how to use these commands in the LOGIN entry. However, you can also use them directly at the terminal.

See the *UNIX Programmer’s Manual* and *UniVerse System Description* for details about the options available with these commands. There are two points worth mentioning here, though.

First, on System V systems there is a special form of the *stty* command that is sometimes useful when a program dies and leaves a terminal in an inconsistent state:

```
$ stty sane
```

sane resets all modes to reasonable values.

Second, on Berkeley systems you can use the *reset* command to reset the terminal. Type a linefeed (**Ctrl-J**) before and after you type **reset**, because pressing **Return** does not always work.

```
% <Linefeed>reset<Linefeed>
```

Mapping Terminals and Auxiliary Printers

If your system is running UniVerse with NLS mode enabled, you can assign a map to a terminal or auxiliary printer to convert UniVerse data from its internal storage format to an external character set. You can assign a terminal map by:

- Setting the system default in the *uvconfig* file
- Setting an entry in the *terminfo* file
- Using the SET.TERM.TYPE command

To set the default map for any terminal having no map assigned to it, set the NLSDEFTERMMAP configurable parameter. This parameter also sets a default map for any auxiliary printer attached to a terminal. For example:

```
NLSDEFTERMMAP ISO8859-1
```

To assign a map for a terminal, change the *at80* setting in a *terminfo* entry. To assign a map for an auxiliary printer, change the *at81* setting. For example:

```
at80=ISO8895-1  
at81=ISO8895-1
```

To override the default map on the current terminal or its auxiliary printer, use the SET.TERM.TYPE command.

For more information about terminal and auxiliary printer maps, see *UniVerse NLS Guide*.

Administering Printers and the UniVerse Spooler

Configuring Printers.	11-4
Defining a New Printer	11-4
Viewing and Modifying a Printer Definition.	11-7
Deleting a Printer Definition.	11-7
Defining and Administering Printer Groups	11-8
Defining a Printer Group	11-8
Adding Users or Printers to a Printer Group	11-11
Removing Users or Printers from a Printer Group	11-11
Deleting a Printer Group	11-12
Managing Printers	11-13
Mounting Forms on a Printer	11-13
Setting Printer Queuing Options	11-14
Starting and Stopping Printers	11-14
Configuring the UniVerse Spooler	11-15
Menu Bar.	11-16
Printer Information.	11-17
Jobs List	11-17
Task Buttons.	11-18
Changing the Spooler Configuration	11-19
Managing Print Jobs.	11-22
Changing Print Job Characteristics.	11-22
Controlling Print Jobs	11-25
Logging Spooler Activity	11-28
Displaying Spooler Log Files	11-28
Determining When a Job Was Printed	11-29
Starting, Stopping, and Resetting the Spooler	11-31
Starting the Spooler	11-31

Stopping the Spooler	11-31
Resetting the Spooler	11-31
About the UniVerse Spooler	11-33
What Happens When the Spooler Is Installed	11-33
Spooler Directories and Files.	11-33
Spooler Processes and Commands	11-35
How the Spooler Works	11-36
Using UniVerse Spooler Printer Drivers.	11-38
Using a UNIX Executable as a Driver	11-38
The Bourne Shell as a Driver.	11-39
Using a Driver for Remote Printing.	11-40
Complex Shell Script Drivers	11-40
Setting Interface Characteristics in a Driver	11-41
Capturing Spool Output	11-42
Using Command Line Arguments in Driver Scripts	11-42
Using the UNIX Spooler with the UniVerse Spooler.	11-44
Changing the UNIX <i>lp</i> Interface File	11-44
Adding a DRIVER Option to the sp.config Entry	11-45
Troubleshooting the Spooler	11-46
Printing Problems	11-46
Getting Incorrect Printout.	11-53
Frequently Asked Questions	11-57

On Windows platforms, you perform all printer administration using the Windows Print Manager. On UNIX systems you configure printers from the **Devices** dialog of UniAdmin. You administer the UniVerse spooler and manage print jobs from the **Spooler** dialog box of UniAdmin.

This chapter describes how to configure printers and administer the UniVerse spooler only on UNIX servers. For information about administering printers and print jobs on Windows platforms, see your Windows documentation.



Configuring Printers

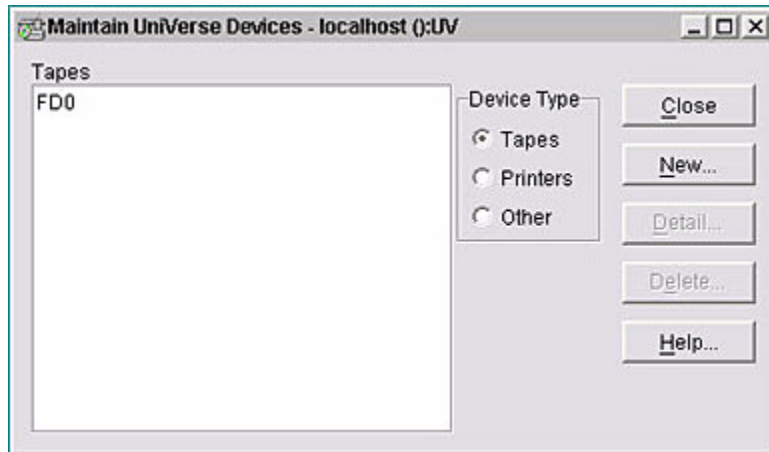
Choose the **Devices** option from the UniAdmin menu to add, modify, and delete printer definitions on a UNIX system. The `&DEVICE&` file and `sp.config` file are automatically updated with your changes.

***Note:** On a Windows system, you must configure all printers using the Windows Print Manager. You cannot use the **Devices** option of UniAdmin to configure printers if you are connected to a Windows system.*

Defining a New Printer

Complete the following steps to define a new printer:

1. From the UniAdmin Choose **Devices** from the UniAdmin main menu. The **Maintain UniVerse Devices** dialog box appears, as shown in the following example:



2. Click **Printers**. The Printers list displays all the printers defined in the `&DEVICE&` file and the `sp.config` file.

- Click **New**. The Printer Details dialog box appears, as shown in the next example:

- Enter the name of the new printer in the **Printer Name** field.
- (Optional) Enter a description of the printer in the **Description** field.
- Select a suitable baud rate from the **Baud Rate** list. The default is 9600.
- To mount a form on the new printer, enter a name in the **Form** field.
Note: To mount more than one form on the printer, use the usa command.
- Select a suitable word length from the **Word Length** list. This is the number of data bits that make up a word, not including the parity bit. Use the arrows to scroll between the minimum and maximum values (5 and 8, respectively).
- Enter the path of the printer device file in the **Device** field, or click **Browse** to search the system for a suitable file.



10. Enter the path of the driver file in the **Driver** field (if used), or click **Browse** to search the system for a suitable file.
11. Set any of these flags by selecting the appropriate check boxes:
 - **Enable Printing.** This flag determines the initial behavior of the spooler daemon for this print device. The default setting is selected, for example, printing is enabled.
 - **Enable Queuing.** This flag determines the initial behavior of the spooler daemon for this print device. The default setting is selected, for example, queuing is enabled.
 - **Tab Expansion.** This flag determines whether tabs are expanded to spaces on output. The default setting is selected, for example, tabs are expanded.
12. Determine how carriage returns and linefeeds are handled by clicking the appropriate CR Mode option. The default setting is **No Conversion**.
13. Click the appropriate **Flow Control** option. This setting determines the communication flow control for the printer. The default setting is **XON/XOFF starts**.
14. Enter the paths of lock files to use in the **Lock Files** fields. These are optional fields where you can specify the paths of two different lock files. The specified files must not exist on the system. You can also use **Browse** to construct a suitable file path.
15. Select the linefeed setting from the **Line Feed** list. The default setting is **None**.
16. Select the formfeed setting from the **Form Feed** list. The default setting is **No FF**.
17. Click the appropriate **Parity** option. The default setting is **None**.
18. Edit the **Other Options** field, if required. You can use this field to specify any of the UniVerse PTERM settings. See *UniVerse User Reference* for a description of the PTERM (UNIX) command.
19. Click **OK**. The new print device definition is written to the &DEVICE& file and the *sp.config* file. The **Maintain UniVerse Devices** window is updated.

Viewing and Modifying a Printer Definition

You can view and modify the details of any printer defined in the &DEVICE& file. To view a printer definition, double-click the printer in the **Printers** list, or select the printer from the **Printers** list and click **Detail**.

The **Printer Details** dialog box appears. You can modify any of the definition settings. Click **OK** to save any changes to the &DEVICE& and *sp.config* files.

Deleting a Printer Definition

Complete the following steps to delete a printer:

1. Select the printer from the **Printers** list in the **Maintain UniVerse Devices** dialog box.
2. Click **Delete**. A message box appears.
3. Click **Yes**. The printer definition is removed from the &DEVICE& and *sp.config* files. The **Maintain UniVerse Devices** dialog box is updated.

Defining and Administering Printer Groups

System printers are usually controlled by the system administrator. However, you can define a subset of printers as a printer group that can be controlled by users who do not have administrative privileges.

Printer groups and the users allowed to access them are defined in the *print_group* file, which is in the UniVerse spooler directory. The *print_group* file has a format similar to that of the */etc/group* file. Each printer group is defined by a line in the *print_group* file. This line has the following format:

```
ptr.group: user1, user2, user3, ...: printer1, printer2, printer3, ...
```

ptr.group (field 1) is the name of the printer group. Field 2 contains the names of users who have access to the printers in the printer group. Field 3 contains the names of the printers included in the printer group.

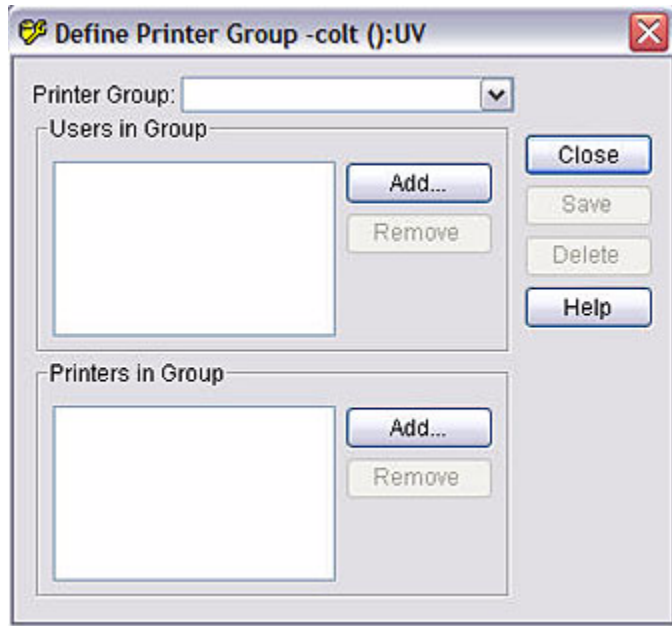
To define or modify a printer group, from the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears. From this dialog box you can add, change, or delete a printer group definition. The *print_group* file is updated when you click **Save**.

Defining a Printer Group

When you define a printer group, you enter a unique name for the group and specify the users and printers that belong to the group. Each printer group must contain at least one printer and one user.

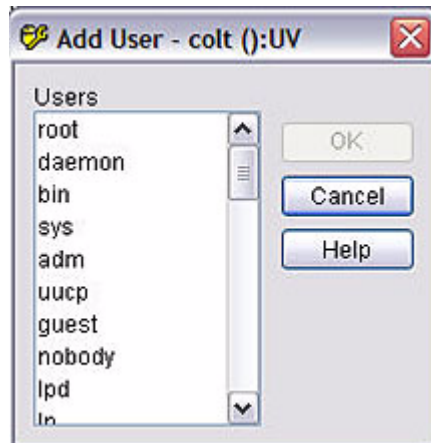
Complete the following steps to define a printer group:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears, as shown in the following example:

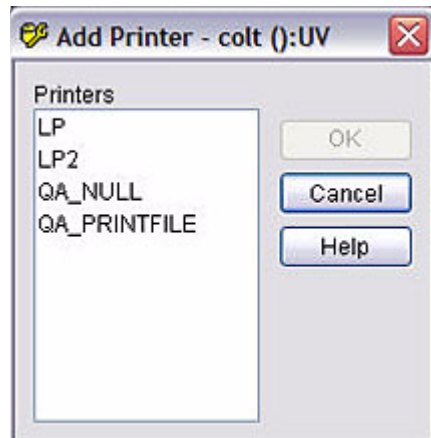


2. Enter the name of the new group in the **Printer Group** field.

3. In the **Users in Group** area, click **Add**. The **Add User** dialog box appears, as shown in the following example:



4. Select the users to add from the **Users** list.
5. Click **OK**.
6. In the **Printers in Group** area, click **Add**. The **Add Printer** dialog box appears, as shown in the following example:



7. Select the printers to add from the **Printers** list.
8. Click **OK**. The **Define Printer Group** dialog box is updated.
9. Click **Save**, then **Close** to exit the **Define Printer Group** dialog box.

Adding Users or Printers to a Printer Group

You can add users or printers, or both, to a printer group. Complete the following steps to add users to a printer group:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears.
2. Select the printer group you want to modify from the **Printer Group** list.
3. In the **Users in Group** area, click **Add**. The **Add User** dialog box appears.
4. Select the users to add from the **Users** list.
5. Click **OK**. The **Define Printer Group** dialog box is updated.
6. Click **Save** followed by **Close** to exit the **Define Printer Group** dialog box.

Complete the following steps to add printers to a printer group:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears.
2. Select the printer group you want to modify from the **Printer Group** list.
3. In the **Printers in Group** area, click **Add**. The **Add Printer** dialog box appears.
4. Select the printers to add from the **Printers** list.
5. Click **OK**. The **Define Printer Group** dialog box is updated.
6. Click **Save**, then **Close** to exit the **Define Printer Group** dialog box.

Removing Users or Printers from a Printer Group

You can remove users or printers, or both, from a printer group. To remove users from a printer group:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears.
2. Select the printer group you want to modify from the **Printer Group** list.
3. Select the users you want to remove from the list.
4. Click **Remove**. The **Define Printer Group** dialog box is updated.

5. Click **Save** followed by **Close** to exit the **Define Printer Group** dialog box.

To remove printers from a printer group:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears.
2. Select the printer group you want to modify from the **Printer Group** list.
3. Select the printers you want to remove from the list.
4. Click **Remove**. The **Define Printer Group** dialog box is updated.
5. Click **Save**, then **Close** to exit the **Define Printer Group** dialog box.

Deleting a Printer Group

Complete the following steps to delete a printer group:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Printer Groups**. The **Define Printer Group** dialog box appears.
2. Select the group you want to delete from the **Printer Group** list.
3. Click **Delete**. A message box appears.
4. Click **Yes**. The printer group is removed.
5. Click **Save**, then **Close** to exit the **Define Printer Group** dialog box.

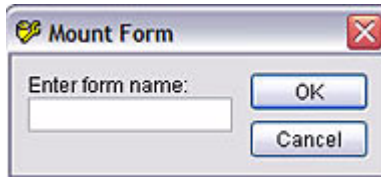
Managing Printers

Use the Printer menu options from the **Spooler** dialog box to start and stop a printer, specify whether jobs are queued, or mount a form. These options apply to the current printer displayed in the **Printer** field.

Mounting Forms on a Printer

Complete the following steps to mount one form on a printer:

1. From the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Mount Form**. The **Mount Form** dialog box appears, as shown in the following example:



2. Enter the name of the form in the **Enter form name** field.
3. Click **OK**. The Spooler window reappears and the **Form** field is updated with the name of the mounted form.

***Note:** Using UniAdmin, you can mount only one form on a printer. Use the `usa` command to mount more than one form on a printer.*

To remove all currently mounted forms from a printer, follow the steps to mount a form, but do not enter anything in the **Enter form name** field.

The UNIX shell command for mounting and aligning forms on a printer is as follows:

```
usa -p printer -F[a|d] formlist -a { #lines | filename }
```

formlist is a list of form names separated by commas (no spaces before or after the comma). The *-F* option replaces the list of currently mounted forms with the new form list. *-Fa* adds the forms in *formlist* to the current list. *-Fd* deletes the forms in *formlist* from the current list.

#lines is the number of lines of the next queued job to print. *filename* is the name of the UNIX file to print.



Setting Printer Queuing Options

To allow or disallow queuing on a printer, from the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Allow Queuing**.

This option is used to both allow and disallow queuing on a printer. When queuing is enabled, a check mark appears next to the option.

Starting and Stopping Printers

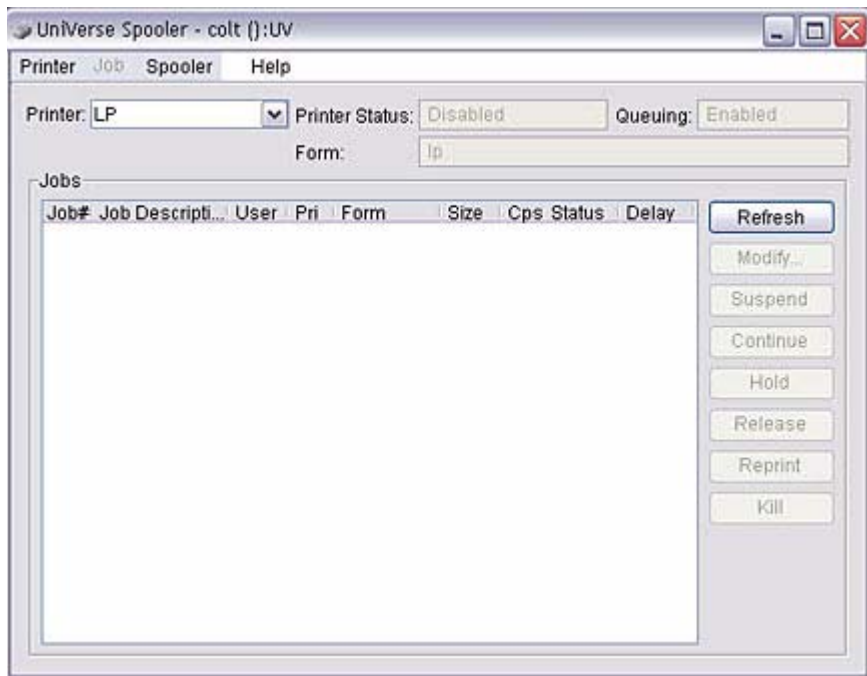
To start or stop a printer, from the UniAdmin main menu, click **Spooler**, then select the **Printer** menu, then click **Allow Printing**.

This option is used to both start and stop a printer. When a printer is started, a check mark appears next to the option.

Configuring the UniVerse Spooler

The UniVerse spooler lets users share system printers. The spooler queues print files that are waiting to be printed to a specific printer and handle requests to print multiple copies of a file, to use special forms, and to format print jobs for specific devices.

To display the Spooler window, choose **Spooler** from the UniAdmin main menu. The **UniVerse Spooler** dialog box appears with a list of the currently spooled print jobs, as shown in the following example:



Note: The **Spooler** option is not available when you are connected to a Windows server.

The tasks you can perform from this window include:

- Changing the spooler configuration
- Managing print jobs
- Logging spooler activity

- Starting, stopping, and resetting the spooler
- Defining and administering printer groups
- Managing printers

The Spooler dialog box has the following components:

- Menu bar, with four pull-down menus
- Printer information
- Jobs list
- Task buttons

Menu Bar

The menu bar has four pull-down menus:

Menu	Description
Printer	Manages printer groups, mounts forms, defines printer characteristics, and exits the Spooler window.
Job	Modifies the characteristics of a chosen print job. This menu is available only when you select a print job.
Spooler	Configures, resets, starts, and stops the spooler, and also views log files.
Help	Invokes the Help system.

Spooler Menus

Printer Information

This part of the Spooler window contains the following fields:

Field	Description
Printer	Shows the name of a chosen printer. You can select any printer from the list, which includes all the printers defined in the &DEVICE& file.
Printer Status	Displays the current state of the chosen printer (whether printing is enabled or disabled).
Queuing	Displays whether queuing is enabled or disabled for the chosen printer.
Form	Shows the name of a mounted form.

Spooler Window Fields

Jobs List

The Jobs list contains all the queued print jobs for the chosen printer. The following information appears for each entry:

Column	Description
Job#	The job ID number, assigned when the print job is created.
Job Description	A description of the print job.
User	The name of the user who issued the print command.
Pri	The priority number of the print job. The lower the number, the higher the priority.
Form	The name of an attached form on which the job must be printed. The same form must be mounted on the printer.
Size	The size (in bytes) of the print job.

Jobs List Fields

Column	Description
Cps	The number of copies to be printed.
Status	The current status of the print job. Hold indicates that the print job is being held. Wait indicates that the job is queued. Active indicates that the print job is currently being printed. If the status is followed by &, a copy of this file has already been printed. If the status is followed by *, the print file will be held after being spooled.
Delay	This field is blank if no delay has been set or if the delay period has been exceeded. This field counts down the amount of delay time.

Jobs List Fields (Continued)

Task Buttons

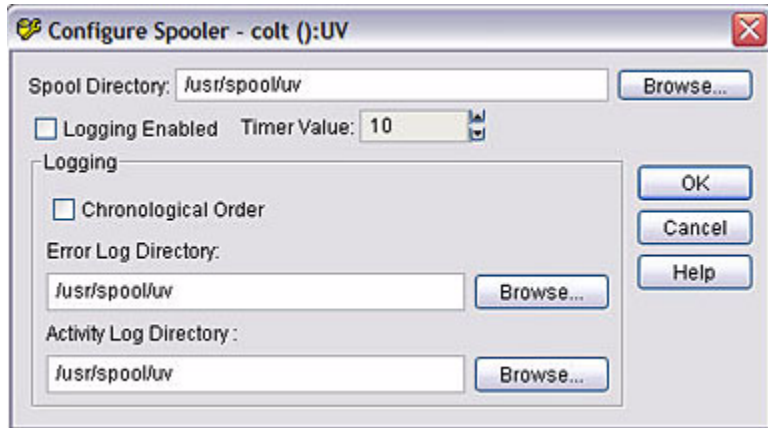
You can control print jobs by choosing options from the Job pull-down menu or by using the buttons. These functions are enabled (active) only when you choose a print job from the Jobs list. The following buttons are available:

Button	Description
Refresh	Refreshes the Jobs list with current printer information.
Modify	Displays the Modify Job Characteristics dialog box, which allows you to modify the characteristics of the chosen print job.
Suspend	Interrupts printing by suspending the chosen print job. All other print jobs are queued until the suspended print job is continued.
Continue	Continues printing a previously suspended print job.
Hold	Puts the chosen print job in the Hold state. The print job remains in the queue until it is released, reprinted, or killed.
Release	Releases and prints a job in the Hold state.
Reprint	Sets the Retain After Printing flag and releases a job for printing.
Kill	Kills the chosen print job, removing it from the queue.

Task Buttons

Changing the Spooler Configuration

To configure the UniVerse spooler, from the UniAdmin main menu, choose the **Spooler** menu, then click **Configure Spooler** from. The **Configure Spooler** dialog box appears, as shown in the following example:



You can change the following information that the spooler uses to communicate with the system:

- Location of the spooler directory
- Order of print jobs
- Spooler response time
- Paths of log files
- Enabling spooler log files

After you make the changes you want, click **OK**. The new settings are saved in the *uv.rc* file.

You can also make these changes using the UNIX shell command *usd*. For more information about *usd*, see *UniVerse User Reference*.

Note: To use the new settings, you must stop and restart the spooler.



Moving the UniVerse Spooler Directory

The spooler directory path defines where spooled files are created. The system printer configuration file *sp.config*, and the spooler queue log file *usplog*, are in this directory. To move the spooler directory, enter a new path in the **Spool Directory** field. You can also use **Browse** to search the system for a suitable path.

The UNIX shell command to change the location of the spooler directory is **usd directory**, where *directory* is the new path for the spooler directory.

Changing the Order in Which Jobs Are Printed

The **Chronological Order** check box specifies the order in which print jobs are printed. When this check box is selected, print jobs are printed in first-in, first-out order. The default setting is cleared (off).

The UNIX shell command to change the print job order is **usd -t**.

Changing the Response Time

The response time is the number of seconds the spooler waits for the system to respond to commands. Set the time in the **Timer Value** field. Enter a number or use the arrows to increase or decrease the value. The default is 10 seconds.

The UNIX shell command to change the response time is **usd -w seconds**.

Changing Log File Paths

The **Error Log File** field displays the name of the file the spooler uses to report errors. If this entry is the name of a directory, the default *err.log* file is used. To change the name or destination of the error log file, enter a new path in this field. You can also use **Browse** to search the system for a suitable path.

The **Activity Log File** field displays the name of the file that stores activity messages. If this entry is the name of a directory, the default *act.log* file is used. To change the name or destination of the activity log file, enter a new path in this field, or click **Browse** to search the system for a suitable path.

The UNIX shell command **usd -L** creates the default error and activity log files in the spooler directory. To create the error log file in another directory, use **usd -e path**; to create the activity log file in another directory, use **usd -a pathname**.

Enabling Logging

The **Logging Enabled** check box determines whether logging is active. Select this check box to enable logging to the error log file and the activity log file. The default setting for this option is cleared (logging disabled).

Managing Print Jobs

You can manage most print jobs from the Job menu or the buttons on the Spooler dialog box. The **Modify** button lets you change various print job characteristics such as the number of pages to print, which pages to print, when to print, and so on. The buttons in the **Spooler** dialog box let you make changes to the print jobs themselves—for example, cancelling or temporarily suspending print jobs, or reprinting them.

You can also perform many spooler administration functions from a UNIX shell, using the commands `usa` and `usm`. For more information about these commands, see *UniVerse User Reference*.

Changing Print Job Characteristics

To view or change the characteristics of a queued print job, choose the print job from the Jobs list on the Spooler dialog box and click **Modify**, or from the **Job** menu, click **Modify**.

The Modify Job Characteristics dialog box appears:

The screenshot shows a dialog box titled "origin - Modify Job Characteristics". It contains the following fields and controls:

- Job ID:** A text field containing "1".
- Copies:** A spin box set to "1".
- Retain After Printing:** An unchecked checkbox.
- Destination:**
 - Printer Name:** A dropdown menu showing "LP".
 - Form Name:** An empty text field.
 - Priority:** A spin box set to "30".
- Defer Printing:**
 - Delay By:** A selected radio button.
 - Defer Until:** An unselected radio button.
 - Hours:** A text field.
 - Mins:** A text field.
- Pages to Print:**
 - All:** A selected radio button.
 - Pages:** An unselected radio button.
 - From:** A text field.
 - To:** A text field.
- Lines to Print:**
 - All:** A selected radio button.
 - Lines:** An unselected radio button.
 - From:** A text field.
 - To:** A text field.
- Buttons:** "OK", "Cancel", and "Help" buttons on the right side.

Each queued print job has a job ID, which cannot be edited. The following sections describe how to change job characteristics. Changes made to these fields are saved when you click **OK**.

Specifying the Number of Copies

The number of copies to print defaults to 1. Use the arrows to change the number of copies or enter a number in the **Copies** field. The UNIX shell command is **usm -n copies print.job**.

Specifying a Printer

You can specify a printer to output to by selecting one from the Printer Name list. The printers defined in the &DEVICE& file are listed here. The UNIX shell command is **usm -p printer print.job**.

Attaching a Form

The **Form Name** field contains the name of a form if one was specified when the print command was issued. You can attach a form to a queued print job by entering the form name in the **Form Name** field.

***Note:** The name entered must match the name of a form attached to a printer. Otherwise the print job is not printed.*

The UNIX shell command is **usm -F formname print.job**.

Setting the Job Priority

The highest job priority is 1 and the lowest is 255. Use the arrows to change the priority or enter a number in the **Priority** field. The UNIX shell command is **usm -P priority print.job**.

Specifying When to Print

You can define the time at which you want a print job to print. This can be done in two ways:

- **Delay By.** This option specifies the relative time, for example, delay printing until 4 hours from now. This is the default setting.
- **Defer Until.** This option specifies the absolute time you want the spooler to print the job, in hours and minutes.

The delay time must be entered in the **Hours** and **Mins** fields. These fields are blank by default, that is, there is no delay period.



The UNIX shell command is **usm -t** *delay print.job*.

Specifying the Pages to Print

You can print all pages in the print job or only those in a selected range. The default setting for the pages to print option is **All**.

To print a range of pages:

1. Click the **Pages** option.
2. Enter the page number where you want printing to start in the **From** field
3. Enter the page number where you want printing to end in the **To** field.

The UNIX shell command is **usm -x** *start.page* [*-end.page*] *print.job*.

Specifying the Lines to Print

You can print all lines in a print job or only those in a specified range. The default setting for the lines to print option is **All**.

To print a range of line numbers:

1. Click the **Lines** option.
2. Enter the line number where you want printing to start in the **From** field.
3. Enter the line number where you want printing to end in the **To** field.

The UNIX shell command is **usm -y** *start.line* [*-end.line*] *print.job*.

Retaining a Job After Printing

You can retain a print file after it has been printed by selecting the **Retain After Printing** check box. The print file is retained in the Hold state, which you can release or reprint later.

This flag is set automatically when you choose to reprint a job.



Note: *Once this flag has been set, it is permanently active and the **Retain After Printing** option is dimmed on the **Modify Job Characteristics** dialog box. The print job is then held in the queue in a permanent Hold state, and you must kill it to remove it from the queue.*

The UNIX shell command is **usm -q** *print.job*.

Controlling Print Jobs

You can control a print job that is currently printing or in the queue in the following ways:

- Killing a print job
- Holding a print job
- Releasing a print job
- Reprinting a print job
- Suspending a print job
- Continuing a suspended print job

These tasks can be performed from the Job pull-down menu or by using the buttons in the Spooler dialog box. You can also use UNIX shell commands.

Killing a Print Job

You can remove (kill) a print job from the queue at any time. If the print job is actively printing, the print job ends prematurely, and the contents of the printer buffer are printed out. Complete the following steps to kill a print job:

1. Select the print job from the Jobs list.
2. Click **Kill**, or from the **Job** menu, select **Kill Job**.

The UNIX shell command is **usm -k print.job**.

Holding a Print Job

You can hold any print job that is not actively printing. The print job is then held with a status of Hold, and is printed when it is released or reprinted. Complete the following steps to hold a print job:

1. Select the print job from the **Jobs** list.
2. Click **Hold**, or from the **Job** menu, select **Hold Job**.

You can also retain a print file in a Hold state after printing it by selecting the **Retain After Printing** check box on the **Modify Job Characteristics** dialog box.

A print file is also retained in a Hold state if you choose to reprint a job.

The UNIX shell command is **usm -h print.job**.



Releasing a Print Job

When a print job is held in the queue (with a Hold status), you can release it for printing as follows:

1. Select the hold file from the Jobs list.
2. Click **Release**, or from the **Job** menu, click **Release Job**.

The released print job is printed (when a printer becomes available) and removed from the queue.

*Note: If the **Retain After Printing** flag has been set for the print job, clicking the **Release** button results in the file being printed, but the job is retained in a Hold state.*

The UNIX shell command is **usm -r print.job**.

Reprinting a Print Job

When you click **Reprint**, the **Retain After Printing** flag is set for the chosen job and the file is released for printing. The print file is retained in the queue in a Hold state, and you can reprint the job again or kill it to remove it from the queue. Complete the following steps to reprint a file:

1. Select the file from the Jobs list.
2. Click **Reprint**, or from the **Job** menu, select **Reprint Job**.
 - Click the **Reprint** button.
 - Choose **Job -> Reprint Job**.

Suspending a Print Job

If a print job is actively printing, you can suspend it until you are ready to continue printing. Complete the following steps to suspend an active print job:

1. Select the active print job from the Jobs list.
2. Click **Suspend**, or from the **Job** menu, click **Suspend Job**.

The print job remains in a suspended (Wait) state until you continue printing. All other print jobs to the printer are also queued until the suspended print job is continued.

The UNIX shell command is **usa -p printer -b**.

Continuing a Suspended Print Job

Complete the following steps to continue a suspended print job:

1. Select the suspended print job from the Jobs list.
2. Click **Continue**, or from the **Job** menu, select **Continue Job**.
The print job prints when a printer becomes available.

The UNIX shell command is **usa -p printer -c**.

Logging Spooler Activity

When the UniVerse spooler is first installed, the spooler queue log file *usplog* is created. This file contains information about the printers that exist on the system and print jobs queued for each printer.

You can create two additional spooler log files: a file that logs all spooler errors generated by the spooler process (*err.log*), and a file that logs all spooler and printer activity (*act.log*). By default, these two files are created in the UniVerse spooler directory, but you can specify any UNIX path for these log files.

See “[Changing the Spooler Configuration](#)” on page 19 for details about how to enable logging and how to specify an alternative path for the log files.

The options to view log files and to find out details of a print job are in the Spooler pull-down menu.

Displaying Spooler Log Files

You can display the *err.log* file and the *act.log* file from the Spooler pull-down menu.

To display the error log file, from the **Spooler** menu, select **Read Error Log**. The **Error Log File** window appears. Close this window by clicking **Close**.

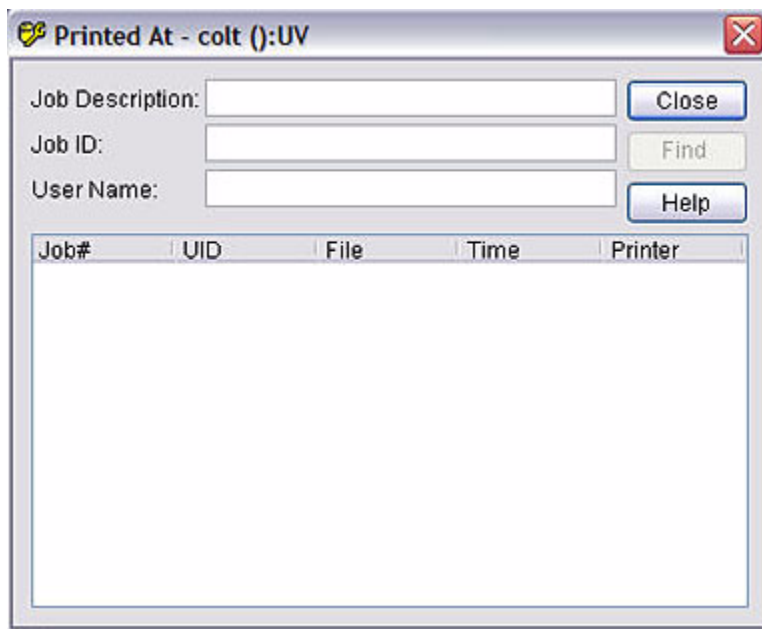
To display the activity log file, from the **Spooler** menu, select **Read Activity Log**. In this case, the **Activity Log File** window appears. Close this window by clicking **Close**.

***Note:** In both cases, only the last 16K bytes of the log file are displayed.*



Determining When a Job Was Printed

If logging is enabled, you can determine when a job was printed by choosing **File Printed At** from the **Spooler** menu. The **Printed At** dialog box appears, as shown in the following example:



To find details for a print job:

1. Fill in the **Job Description**, **Job ID**, or **User Name** fields as appropriate.
2. Click **Find**. The list is updated with each print job that meets the entered criteria.

The following information appears for each entry in the list:

Field	Description
Job#	The job ID of the printed file.
UID	The UniVerse user number of the user who issued the print command.

Job List Information

Field	Description
File	The description of the print job.
Time	The time the file was printed.
Printer	The printer used to output the file.

Job List Information (Continued)

3. Click **Close** to close this dialog box.

Starting, Stopping, and Resetting the Spooler

You can start, stop, and reset the spooler from the **Spooler** menu.

Starting the Spooler

If the spooler daemon was stopped, you can restart it. When the spooler daemon starts, the *usplug* file is read and the queues are restored.

Complete the following steps to start the spooler daemon:

1. From the **Spooler** menu, select **Start Spooler**. (This option is available only if the spooler daemon is stopped.) The Spool message box appears.
2. Click **Yes**.

Stopping the Spooler

If the spooler daemon is running, you can stop it. When you stop the spooler daemon, the spooler daemon process is terminated.

Complete the following steps to stop the spooler daemon:

1. From the **Spooler** menu, select **Stop Spooler**. (This option is available only if the spooler is running.) The Spool message box appears.
2. Click **Yes**. A message box appears.
3. Click **OK** to acknowledge the message.

Resetting the Spooler

If the spooler daemon was started but appears to hang, you can reset it. When you reset the spooler, the spooler daemon is restarted, the *sp.config* file and *usplug* file are reread, and the queues are reinstated to the last active state.

Complete the following steps to reset the spooler daemon:

1. From the **Spooler** menu, select **Reset Spooler**. (This option is active only when the spooler is started.) The Spool message box appears.
2. Click **Yes**. The spooler daemon is reset.

About the UniVerse Spooler

Installation and a general understanding of how the spooler works require a basic understanding of UNIX.

What Happens When the Spooler Is Installed

The UniVerse spooler is installed as part of the initial UniVerse installation procedure. The following things occur when the spooler is installed:

- The UniVerse spooler directory (by default, */usr/spool/uv*) is created.
- The UniVerse spooler daemon (*usd*) is installed.
- The UniVerse spooler queuing process command (*usp*) is installed.
- The UniVerse spooler administration command (*usa*) is installed.
- The UniVerse spooler queue modification command (*usm*) is installed.
- A spooler queue log file (*usplog*) is created.
- A system printer configuration file (*sp.config*) is created.

When the initialization script (*/etc/uv.rc*) is installed and run, any device lock files in the UniVerse spooler directory are cleared, and the UniVerse spooler daemon starts.

Spooler Directories and Files

Several directories and files are associated with the UniVerse spooler.

The Spooler Directory

The default spooler directory, unless you change it by reconfiguring UniVerse, is */usr/spool/uv*. This directory stores copies of all files queued for printing. It also contains files that control the spooler. The spooler directory must have *rw-rw-rw-* protection, which means that it can be read, written, and executed by the owner, group, and all others.

The system startup script */etc/uv.rc* gets the spooler directory location from the configurable parameter *UVSPOOL* in the *uvconfig* file. You can move the spooler directory to a different partition.

Moving the Spooler Directory

You may find that the `/usr` partition is too small to handle many large print files. You can solve this problem by moving the UniVerse spooler directory to a partition with more space.

Note: Before moving the spooler directory, make sure no users are using UniVerse or the spooler.

To move the spooler directory:

1. Change the UVSPPOOL configurable parameter in the `uvconfig` file to the path of the new spooler directory.

Note: This directory must already exist and have permissions `rwrxwrxwx`. Changing this parameter directs print files generated in the UniVerse environment and by the `usp` command to the specified directory.

2. Move all files in the `/usr/spool/uv` directory to the new directory.
3. When no users are on the system, run `uvregen` and `DBsetup`.
4. Shut down and start up the spooler using the Spooler pull-down menu options.

Spooler Queue Log File

A spooler queue log file called `usplog` is created in the spooler directory. `usplog` preserves information about the printers and the queued print jobs. As print files are created, their details are written to this file, overwriting the previous contents. The `usplog` file ensures that print files are not lost if the spooler fails or the system crashes.

Warning: Do not edit the `usplog` file. It contains information the spooler daemon uses when it restarts after a normal or abnormal system shutdown. The information in `usplog` ensures that the spooler restarts in more or less the same state it was in when the system shut down.

You can create additional files to log spooler activity. See [“Logging Spooler Activity”](#) on page 28.



System Printer Configuration File

A system printer configuration file called *sp.config* is created in the spooler directory. This file describes the device path and the characteristics of printers controlled by the UniVerse spooler.

When changes are made to the spooler configuration, the new settings are written back to this file.

The uv.rc script (in the /etc directory) contains the spooler configuration settings and is used to invoke the spooler daemon at system startup.

On some machines it may be necessary to rebuild the kernel to allow the spooler to function properly. Check the *UniVerse Installation Guide* to see if this applies to your system.

Spooler Processes and Commands

A single spooler process (daemon) automatically starts when the system starts.

You can use several UniVerse commands to perform spooler administration without using the **Spooler** option from UniAdmin. The spooler administration commands are *usp*, *usa*, and *usm*. You use these commands from a UNIX shell; you do not use them from the UniVerse environment. See the *UniVerse User Reference* for more information about syntax and use.

Spooler Daemon (usd)

The UniVerse spooler daemon is installed in the *bin* directory of the UV account. This daemon is generally initiated at system startup and runs at all times in the background. *usd* monitors the spooler queue for files to be printed, manages the mounting of special forms on the system printers, and spawns copies of itself to print the files that are queued for active printers with the correct characteristics.

Spooler Queuing Process Command (usp)

The UniVerse spooler queuing process command resides in the *bin* directory of the UV account. Use this command to queue files to be printed.

Spooler Administration Command (usa)

The UniVerse spooler administration command resides in the *bin* directory of the UV account. Use this command to enable or disable printing and queuing, to define printer groups, and to manage the spooler daemon.

Spooler Queue Modification Command (usm)

The UniVerse spooler queue modification command is installed in the *bin* directory of the UV account. Use this command to alter the characteristics of a job in a printer queue.

How the Spooler Works

Once the UniVerse spooler is installed and configured, it should be up and running. At system initialization, the UniVerse spooler daemon *usd* starts. *usd* checks for a message queue entry with a KEY of the form 0xaceaxxxx. If such a message queue entry exists, *usd* exits with the following message:

```
usd: daemon: spooler already active!
```

If the message queue entry does not exist, *usd* creates it and then reads the printer configuration file (*sp.config*) and the *usplog* file, both from the UniVerse spooler directory.

When a UniVerse user sends a printing request to the spooler, a file is created in the UniVerse spooler directory with a name in one of two forms. If the request was generated by a BASIC PRINT ON statement or by using the LPTR keyword, the form is as follows:

```
uvnnnnnnxxxxxxxx
```

If the request was generated by the UNIX shell command *usp*, the form is as follows:

```
usnnnnnnxxxxxxxx
```

xxxxxx is a control sequence generated by the spooler and nnnnn is the spool job number. When the print file is closed, UniVerse sends a message via the message queue. The message states that the file is available for printing, and indicates the number of copies, the form name, whether to delay printing, and so on.

usd uses the information in the message to build an entry in the *usplog* file and then tries to find an available printer that matches the print file characteristics. If it does not find one, *usd* queues the file to the first printer defined in the *sp.config* file and puts it in the wait state.

If *usd* finds a printer, it creates a new process (a copy of itself) to print the file.

The copy of *usd* locks the printer by opening the device with the exclusive use bit set. If a lock file is defined in field 5 of the printer definition in the *&DEVICE&* file, this lock file is used by default. If a lock file is not specified, a zero-length file is created in the spooler directory. This new lock file has the name *lock.xxx*, where *xxx* is the base name of the device path. This file is used by the spooler daemon to coordinate different logical printers that are defined for the same physical printer.

The print file is printed. On completion, the copy of *usd* sends a message to the spooler daemon that printing is complete. It then removes the lock file, closes the device, erases the print file, and exits.

The spooler daemon removes the entry for that print job from the in-memory queue and from the *usplog* file, and attempts to schedule another print job to that printer.

Both the *usm* and the *usa* commands work by sending a message to the spooler daemon via the message queue. The spooler daemon acts on the request and, if appropriate, sends an answering message.

Using UniVerse Spooler Printer Drivers

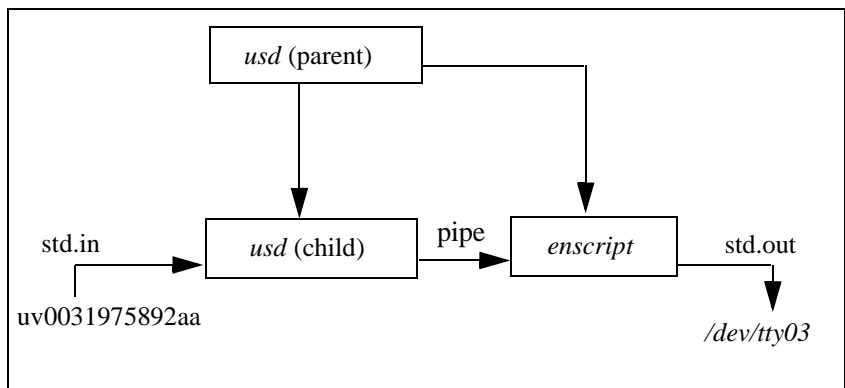
The UniVerse spooler can use printer device drivers. Using a printer driver, you can use UNIX executables (such as driver scripts for PostScript) to print to devices connected to remote systems on a network, specify interface characteristics, and capture spool output.

Once you create a printer driver, you can use the UniAdmin **Devices** option to define it. Under **Pathnames** on the **Printer Details** dialog box, enter the path of the driver in the **Driver** field.

Using a UNIX Executable as a Driver

A common activity is to output to a PostScript printer. To do this, ASCII print files must be translated into PostScript directives. This is easily accomplished by defining a driver. In the following illustration, the PostScript filter is a UNIX executable named `/usr/bin/enscript`, which processes the ASCII file presented on standard input to its PostScript equivalent on standard output.

In the following example, the *sp.config* interface characteristic settings are not shown:



Flow of Data Using a UNIX Executable as a Driver

The Bourne Shell as a Driver

In some cases the PostScript filter may require some parameters. For example, the filter may allow rotating the output for printing in landscape mode. You can put the filter command in a shell script file and define the shell script as the driver. Normal UNIX process management establishes standard input and output for the executable in the shell script. You must set permissions on the shell script file to allow execution by a UniVerse Administrator. You can put the driver script anywhere you want.

The following examples show the contents of the *sp.config* file, the permissions that are set for the driver script, and the contents of the driver script.

Contents of *sp.config*:

```
LP1 /dev/tty03 DRIVER /usr/spool/uv/landscape BAUD...
```

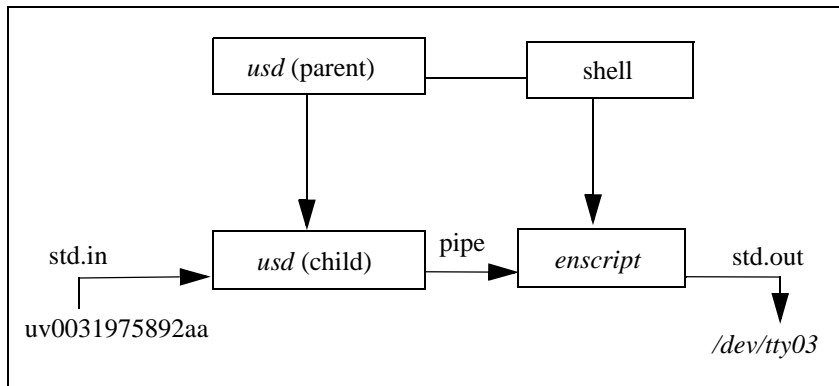
A listing of the driver file shows execute permissions:

```
$ ls -l /usr/spool/uv/landscape
-rwxr-xr-x 1 root      76 Jan 12 17:25 /usr/spool/uv/landscape
```

Contents of */usr/spool/uv/landscape*:

```
/usr/bin/enscript -r
```

The following example shows the flow of data when a Bourne shell driver is used:



Flow of Data Using the Bourne Shell as a Driver

Using a Driver for Remote Printing

Another common use of a driver is to route a print job to a printer on a remote system. In this example the printer LPREM is a printer on another system, *systemb*, networked using TCP/IP. The driver script uses the remote execution command *rsh* to have the UniVerse spooler on *systemb* print the file. The command executed on the remote system is the UniVerse spooler *usp* command, which takes the file on standard input and puts it into the spool queue on the remote system.

Contents of *sp.config*:

```
LPREM /dev/null DRIVER /usr/spool/uv/sysb.drvr BAUD...
```

Contents of */usr/spool/uv/sysb.drvr*:

```
/usr/ucb/rsh systemb /usr/ibm/uv/bin/usp -e -h -F FORMNAME
```

The remote *usp* command should suppress headers because they are supplied by the local machine with the user information.

Note: The remote or network shell command may be *nsh*, *rsh*, or *remsh*. If *rsh* is the remote shell command, be careful to specify the path for *rsh* to ensure that the remote shell, not the restricted shell, is used. Both executables are named *rsh*.

Because no printing is actually done on the local system, the null device is specified as the path (*/dev/null*). You control which printer is assigned on *nodename* by using the appropriate *usp* option, *-F* or *-p*, in the shell script.



Complex Shell Script Drivers

A driver script can be as complex as needed as long as the relationship between standard input and standard output are taken into account. In the following example, the output file is processed by the stream editor filter *sed*, changing all occurrences of 'abc' to 'def' before generating the PostScript output. (The *sp.config* interface characteristic settings are not shown.)

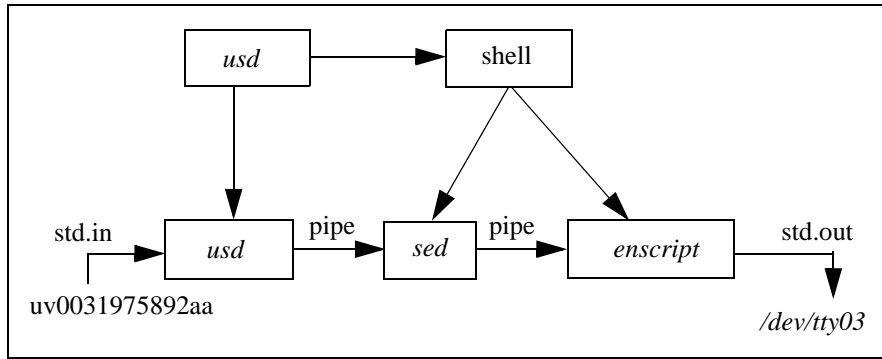
Contents of *sp.config*:

```
LP1 /dev/tty03 DRIVER /usr/spool/uv/sed.drvr BAUD...
```

Contents of */usr/spool/uv/sed.drvr*:

```
sed "s/abc/def/g" | /usr/bin/enscript
```

The following example shows the flow of data when a complex shell script is used:



Flow of Data Using a Complex Shell Script Driver

Setting Interface Characteristics in a Driver

Another use of a driver might be to set interface characteristics by using a shell script containing the UNIX *stty* command. The following example uses *stty* in a script to set the interface characteristics. Since the *usd* process does not set characteristics, the *sp.config* file does not specify any.

Contents of *sp.config*:

```
LPl /dev/tty03 DRIVER /usr/spool/uv/stty.drvr
```

Contents of */usr/spool/uv/stty.drvr*:

```
stty 9600 parenb parodd cs7 opost onlcr <&1
cat -
```

The Bourne shell construct `<&1` instructs the shell to assign standard input for the *stty* command to the assignment for standard output. It is used because *stty* sets the characteristics on the device assigned to standard input. The driver script is executed by the spooler daemon with the printer device assigned as standard output. The `<&1` construct lets you temporarily assign the correct printer device to standard input for the *stty* command. Another approach is to hard code the device address in the script:

```
stty 9600 parenb parodd cs7 opost onlcr </dev/tty03
```


Capturing Spool Output

You can capture the print job to a file or print to several devices at once using the UNIX *tee* filter. The following example captures the spooled output in a file for examination after displaying the printer interface characteristics on the system console at the time of printing.

Contents of *sp.config*:

```
LP1 /dev/tty03 DRIVER /usr/spool/uv/debug.drvr
```

Contents of */usr/spool/uv/debug.drvr*:

```
stty -a <&1 > /dev/console
cat - | tee /tmp/check.spool.out
```

Using Command Line Arguments in Driver Scripts

You can specify any of the following 11 arguments in a driver script:

Argument	Description
\$1	UNIX user ID of the user who spooled the job
\$2	Job ID of the print job
\$3	Size of the print job in bytes
\$4	Job description
\$5	Form assigned to the print job
\$6	UniVerse printer name
\$7	SETPTR (UNIX) line length
\$8	SETPTR (UNIX) page length
\$9	SETPTR (UNIX) eject flag (1 = EJECT, 0 = NOEJECT)
Shift the argument stack down to reference the following two arguments:	
\$1	SETPTR (UNIX) banner flag (1 = print banner, 0 = suppress banner)
\$2	SETPTR (UNIX) USEROPTS options

Driver Script Arguments



These arguments can be useful for printing across a network. Using a convention that the form name is the remote node name and the printer name is set to the desired form name on the remote node, the following shell script would route the job to the desired queue on the remote machine.

Note: *rcmd* is a *UNIX TCP/IP remote execution command* that may be spelled differently on different machines.

Contents of `/usr/spool/uv/remprint.drvr`:

```
/usr/bin/rcmd $5 usp -F $6 -h -e
```

Using the UNIX Spooler with the UniVerse Spooler

The System V UNIX spooler *lp(1)* command assumes it is the only process using the printers. This means that *lp* checks only itself to make sure it is not using a printer before it sends the printer another job. If another process, such as the UniVerse spooler, is using the printer, the UNIX spooler continues to send jobs to the printer as if it were available, and the two print jobs can become mixed. This problem does not occur if you send a print job with the *usd* command after a print job has been sent with the *lp* command, because the UniVerse spooler always checks the status of a printer before sending it a new print job.

The UniVerse spooler is designed to replace the UNIX spooler, so the best solution to this problem is to stop using the UNIX spooler. If you must use both spoolers, there are two ways to solve the problem:

- Change the UNIX *lp* interface file
- Add a DRIVER option to the *sp.config* entry

Changing the UNIX *lp* Interface File

Change the UNIX *lp* interface file so it uses the same external locking file as the UniVerse spooler. There is one *lp* interface file for each print device that *lp* recognizes in the */usr/spool/lp/model* directory. Add the following five lines to the beginning of each *lp* interface file:

```
while [ -f /usr/spool/uv/lock.ttyxx ]
do
sleep 5
done
echo > /usr/spool/uv/lock.ttyxx
```

ttyxx is the printer name defined in the *sp.config* file. These lines cause the spooler to wait until the lock file for the device is removed. The spooler then creates a lock of its own and proceeds with the print job. At the end of the print job the lock file must be removed. This will be done if you add the following line just before the *exit(0)* line in the *lp* interface file:

```
rm /usr/spool/uv/lock.ttyxx
```

Adding a DRIVER Option to the *sp.config* Entry

Add a DRIVER option to the *sp.config* entry that invokes the System V UNIX spooler. A copy of the print job is put in the UNIX spooler queue and printed by the UNIX spooler. The proper concurrency control is maintained as long as the *lp* spooler is not set up with multiple destination queues defining the same device.

Troubleshooting the Spooler

The most common causes of spooler failure are as follows:

- A driver process or a *usd* copy has not terminated correctly and is locking a resource.
- The print file characteristics or the printer configuration is specified incorrectly.

You can fix these problems using the **Spooler** menu in UniAdmin, or by using UNIX shell commands.

Printing Problems

This section describes possible causes (and solutions) when jobs do not print.

Spooler Is Not Running After UNIX Upgrade

UniVerse and UNIX are initialized by *rc* scripts in the */etc* directory.

When the UNIX environment is upgraded, the UNIX installation processes may replace the *rc* script with one that initializes the new UNIX environment. If that is done in a way that does not preserve user changes in the old *rc* script, neither UniVerse nor the spooler are initialized. The simplest remedy is to reinstall the group MAIN from the initialization tape. This requires reauthorization of the license.

When UNIX is upgraded, you must tune the kernel as it was before the UNIX upgrade. Insufficient kernel space for message queue parameters may allow initialization of the UniVerse environment (although not the full complement of users), but not of the spooler. Retune the kernel, reboot, and save or record the key UNIX tunable parameters before upgrading the operating system.

The Spooler Daemon Will Not Start

There are many reasons why the UniVerse spooler daemon (*usd*) does not start. Check for the following causes.

Message Queue Facility Incorrectly Tuned

If you have recently performed a UniVerse installation or upgraded UNIX, it is possible that the UNIX message queue facility is incorrectly tuned. See [“Spooler Is Not Running After UNIX Upgrade”](#) on page 46.

Missing Files After Moving the Spooler Directory

If you have moved the spooler directory, check that you have performed all the steps outlined in [“Moving the Spooler Directory”](#) on page 34.

Verify that the spooler directory has sufficient permissions. All users need permissions to write print files into the spooler directory. The permissions should be as follows:

```
# ls -ld /usr /usr/spool /usr/spool/uv
drwxr-xr-x 33 root      2048 Apr 11 14:34 /usr
dr-xr-xr-x 17 root      2048 Sep  1 12:29 /usr/spool
drwxrwxrwx  2 root      2048 Sep  1 17:35 /usr/spool/uv
```

If you see the following messages in the error log file, it is likely that the *sp.config* file or the *usplog* file is missing:

```
usd: daemon: cannot find sp.config
usd: cannot open daemon log file (usplog)
```

If the *sp.config* file does not exist, follow these steps to recreate it:

1. Choose the **Devices** option from UniAdmin. The **Maintain UniVerse Devices** dialog box appears.
2. Click the **Printers** option.
3. Display the details of a defined printer, but do not make any changes (unless you need to).
4. Click **OK**.
5. Click **Close** to exit the **Devices** option. The **&DEVICE&** file is saved. A new *sp.config* file is created, containing the definition for the printer you chose to view.
6. Repeat steps 3 through 5 to add other printer definitions.

If *usplog* is missing, log in as a UniVerse administrator and recreate it by issuing the following UNIX commands from a system prompt:

```
# cd /usr/spool/uv
# echo z > usplog
# chmod 600 usplog
```

Message: ***Warning: Requested Lock File Already Exists

The UniVerse spooler uses a lock file to indicate that a device is in use. When a printer serves multiple printer queues, the lock file for the device may exist because of activity on a different queue. This warning message occurs when:

- A print job is ready for a queue
- The queue does not have an active print job
- The lock file exists because the device is servicing another queue

Suspending an active print job may fail to remove the lock file. If you try to kill the print job or direct it to another queue, this will not remove the lock file. In this case, remove the lock file with the UNIX command:

```
# rm /usr/spool/uv/lock.xxx
```

lock.xxx is the name of the lock file. If you are not using the default spooler directory, replace */usr/spool/uv* with the path of your spooler directory.

Jobs in Wait State Will Not Go Active

There are many reasons why a job in the Wait state does not go Active. Check for the following possible causes.

Disabled Printing

If printing has been stopped for a printer, the print jobs are not printed until the printer is restarted. Check the **Printer Status** field in the Spooler dialog box. If the status is Disabled, from the **Printer** menu, select **Allow Printing**. A message box appears. Click **OK** to start the printer.

Verify that both printing and queuing are turned on for the print queue. SPOOL -LIST should show P: on Q: on in each queue that should be printing. To turn on printing and queuing, enter the following at the UNIX shell prompt:

```
# usa -p print.queue +o      (enable printing)
# usa -p print.queue +q      (enable queuing)
```

Suspended Print Job

If a print job has been suspended, the other print files in that queue are not printed (made active) until the suspended job is continued or killed.

In the following example, the status for a suspended job is susp:

```
# usa
Printer: lp                Q: on    P: on    Form:
Job # Job description      User name  Pri Forms      Size  Cps
Status Delay
00020 sp.config           root      30           118    1
susp
00021 passwd              root      30          3718    1
wait
```

To free up such a queue, use the following UNIX shell command to put the suspended job into a hold state:

```
# usm -h job.no
```

Mismatched Form Names

If you are using forms, you must ensure that a form mounted on the printer matches that specified for a print job. If the form names do not match, or if you have specified a form for the print job, but it is not mounted on the printer, the print job will never go active.

Note: *Form names are case-sensitive.*

Check the **Form** field on the Spooler window and the **Form** column for a print job. If these do not match, do one of the following:

- Change the form on the printer. See [“Managing Printers”](#) on page 13.
- Change the form associated with the print job. See [“Attaching a Form”](#) on page 23.



The spooler initiates a print job when all the job characteristics match. The UniVerse SPOOL –LIST command displays both the form on the printer and the form associated with the print job:

Printer: lp	Q: on	P: on	Form: LANDSCAPE
Job # Job description	User name	Pri Forms	Size Cps
Status Delay			
00014 sp.config	root	30 MARK	118 1
wait			
00015 passwd	root	30 MARK	3718 1
hold &			

Particular Print Queue Was Specified

If the print job was spooled to a particular print queue with SETPTR,,,,,AT PRINTER *name*, it does not print on another queue unless it is redirected. To direct a print job to a different queue:

1. Select the print job from the Jobs list and click **Modify**. The Modify Job Characteristics dialog box appears.
2. Select a different printer from the Printer Name drop-down list.
3. Click **OK**.

Or use the following UNIX shell command:

```
# usm -p new.queue job.number
```

Jobs in Active State Do Not Print

If print jobs in the active state do not print, you need to verify that the printer is online.

The port could be hung or it could be in use by the UNIX spooler. You cannot check this using UniAdmin. Enter the following UNIX commands at the system prompt:

```
# stty -a </dev/tty.device
# cat >/dev/tty.device
```

tty.device is the name of the printer device.

If either command fails, something is wrong with the port. If *stty* hangs, an unfinished write on the port has filled the UNIX buffer and has not completed. Until the UNIX driver buffer clears, *stty* cannot interrogate the port. If the *cat* command fails, the port has a problem.

SPOOL –LIST and usa Hang

The spooler message queue ID is 0xacea0207. If the correct message queue exists, the commands SPOOL –LIST and usa put their request in the message queue and wait for a reply. The spooler must be running to respond.

Verify that *usd* is not swapped out. Over time, the UNIX scheduler may lower the priority of a daemon process as the quantity of resources consumed grows. Some UNIX implementations give priority to interactive sessions at the expense of processes not associated with a terminal. A busy machine may preempt the spooler from getting the resources needed to respond to a request.

The UNIX *ipcs –qa* command indicates the maximum bytes available in the message queue in the QBYTES column. If there is insufficient space in the message queue to construct a reply message, the daemon ceases to function. This condition requires killing and restarting the spooler. Retune the UNIX kernel to enlarge the message queue space.

The spooler daemon times out when a bad port fails to open. If the spooler hangs trying to open a bad port, the CBYTES increases as more spooler requests are entered. Use the UNIX *ipcs –qa* command to monitor CBYTES. At 10-second intervals, the timer expires and gives the spooler a chance to service other requests. Another *ipcs –qa* command should show a reduction in the number of bytes outstanding in the message queue. Using the UNIX process status command *ps*, interrogate the state of *usd*.

If *usd* is not consuming CPU resources, it may be hung waiting for a bad device. Investigate the associated device from the *ps* command. See if the device responds to the following:

```
# stty -a </dev/tty.device
```

tty.device is the name of the printer device.

Driver Not Found

When the spooler tries to print a job on a queue that specifies a nonexistent driver, the following error message may appear in the error log file:

```
***Error: Unable to open printer driver. Verify that the driver
defined in the sp.config file exists, then re-enable printer (usa
+o -p printer)
```

If you see this error, you need to disable the printer, check that the driver file exists, and then reenable the printer.

To disable printing, from the **UniVerse Spooler** menu, click **Printer**, then click **Allow Printing**.

To check that the driver exists:

1. Choose the **Devices** option from the UniAdmin menu. The **Maintain UniVerse Devices** dialog box appears.
2. Click the **Printers** option.
3. Double-click the printer generating the error from the **Printers** list, or select the printer generating the error from the **Printers** list, then click **Detail**.
4. Check the entry of the **Driver Pathname** field by clicking **Browse**.
5. Click **OK** to save the printer settings.
6. Click **Close** to exit the **Devices** option.

To enable the printer, from the **UniVerse Spooler** menu, click **Printer**, then click **Allow Printing**.

If the spooler directory has been moved and fully qualified paths are used in the *sp.config* file, the spooler daemon searches in the wrong place for the driver. Conversely, if fully qualified paths are not used in *sp.config* and the driver remains in the original directory, the spooler searches in the new spooler directory for the drivers.

Printer Configuration Changes Do Not Take Effect

For printer configuration changes to take effect, the spooler must be instructed to read the *sp.config* file. This can be done by stopping and starting the spooler.

To stop the spooler, from the **UniVerse Spooler** menu, click **Spooler**, then click **Stop Spooler**. A message box appears. Click **Yes**.

To start the spooler, from the **UniVerse Spooler** menu, click **Spooler**, then click **Start Spooler**. A message box appears. Click **Yes**.

When the spooler is started, the *sp.config* file is read to use the new queues that have been specified or reconfigured.

Note: *If you have removed a printer definition and restarted the spooler, the jobs queued on that printer are lost.*



There are two options to the UNIX shell command *usa* that make the spooler reread the *sp.config* file: *-r* and *-R*.

```
# usa -r
```

This command instructs the spooler to read the *sp.config* file, add new queues that have been specified, and change those that have been reconfigured. It does not remove a queue from the current queue configuration if it has been removed from the configuration file. This allows new queues to be added while old queues finish operation. When the spooler is started later, the old queues will be gone.

```
# usa -R
```

This command instructs the spooler to read the *sp.config* file and make the current queue configuration match what is in the configuration file. Queues that have been removed from the configuration file are removed from the current queue configuration.

Getting Incorrect Printout

This section examines situations in which printing occurs but something is wrong with the output.

Specify SETPTR NOEJECT, But Get a Formfeed at End of Report

This is probably a printer issue or a cabling problem. Laser and other sheet-feed printers exhibit this symptom.

To demonstrate that the spooler is not supplying the extraneous formfeed at the end of the print job, set up a print queue in the *sp.config* file:

```
MYQUEUE /tmp/file FORMS MYFORM
```

Set up the print file and print to it. The file */tmp/file* contains the image of the print file when the spooler and UniVerse are done with it. Create a driver script containing the following lines:

```
touch /tmp/file
chmod 666 /tmp/file
usa -R
usp -F MYFORM -e /etc/passwd
```

The *touch* command creates the file. *chmod* gives the file read and write permissions. *usa* rereads the spooler configuration, and *usp* prints the file */etc/passwd*.



Use UNIX *vi* to look for ^L at the end of the file. If the formfeed character is not present, it is being supplied by the physical device.

If the formfeed character is at the end of the file, it is coming from the UniVerse environment. After the SETPTR NOEJECT was specified, another SETPTR was executed. The subsequent SETPTR needs to specify the NOEJECT or the NODEFAULT option. NODEFAULT indicates that all current SETPTR settings should be retained except those explicitly changed. Omitting NODEFAULT causes SETPTR to use the current system defaults.

As UniVerse Administrator in the UV account, you can use SETPTR on channel 0 with any desired settings. Issuing SETPTR.DEFAULT makes these settings the system defaults for all users on all print channels. If SETPTR.DEFAULT is issued before the SETPTR on channel 0, the SETPTR default settings are restored to those that shipped with the UniVerse license.

***Note:** Specify SETPTR NOHEAD, But Still Get Header is a variation of the same problem.*

Specify SETPTR NOHEAD, But Still Get Header

This is a variation of the previous problem (Specify SETPTR NOEJECT, But Get a Formfeed at End of Report).

Print Lines Do Not Return to the Left Side of the Page

The *sp.config* entry for the queue needs to specify OCRNL to change newlines or linefeeds into linefeed and carriage return on output. In addition, OPOST or

LITOUT must be specified to actuate postprocessing. A UNIX environment uses one or the other. The incorrect one will not work.

Postprocessing can be appropriate for parallel printers as well as serial devices. These parameters can also be specified in a driver with an *stty* command. Insert the appropriate line for your system at the beginning of the driver:

```
stty -a ocrnl opost <&1 (SysV)
```

or

```
stty -a -nl -litout <&1 (UCB)
```

If you specify other parameters after these, make sure all of them are correct for the particular UNIX implementation of the *stty* command. An incorrect parameter can cause parameters that follow it to be ignored.

Missing Pages, Lines, or Characters

If you are missing pages, lines, or characters in the printout from the spooler, it is likely that buffers are being overrun in the printer. Check for a protocol problem. The UniVerse spooler can use XON/XOFF or DTR, which is set when you define a printer device.

If XON/XOFF is used, the *sp.config* queue should be specified with XON NOSTARTANY or with `stty ixon -ixany` in the driver. When the printer buffer fills, it emits a stop character, **Ctrl-s**. With XON NOSTARTANY specified, the UNIX driver does not send additional characters until the printer empties enough of its buffer and emits a **Ctrl-q**.

If XON STARTANY ON is specified, the UNIX driver sends additional characters to the printer if any character is received from the printer. Sometimes the driver is slow to respond and continues sending characters to the printer for a time before pausing. The printer buffer can fill completely, and the printer emits another **Ctrl-s** to get the driver to stop sending characters. With STARTANY ON, the second **Ctrl-s** is recognized as a signal to start transmitting additional characters. These are lost.

If these corrections do not solve the problem, verify the cabling.

Jobs Do Not Print in First-In First-Out Order

The spooler normally prints smaller jobs first. To print jobs in first-in, first-out order, select the **Chronological Order** check box in the Configure Spooler dialog box.

Print Jobs Are Intermixed

If you have two different printer definitions that have the same device path, it is possible to define two different lock files to use. Because the lock files are different, the locking semaphore mechanism fails, and the two queues can simultaneously print to the same device. As a result, the print jobs are intermixed.

To prevent this from occurring, avoid using two different lock files for a single device. You can correct this problem in two different ways using the **Devices** option on the UniAdmin menu:

- Do not specify a lock file in the printer definition. The lock file created will be constructed from the device name, resulting in a single lock file used by two queues.
- Edit the printer definitions so that the entry for the lock file is the same.

SP.ASSIGN (UNIX) can try to gain exclusive use of a device. To avoid this, you can specify lock files in the &DEVICE& entry for the device. These must be fully qualified paths. If they are not, the lock file is created in the directory of the account from which the SP.ASSIGN command is issued. If SP.ASSIGN is issued by two users in different accounts, the two lock files will not collide. The two users gain apparent exclusive use of the single device.

Frequently Asked Questions

Question: How do I use the UNIX and the UniVerse spoolers together?

“[Using the UNIX Spooler with the UniVerse Spooler](#)” on page 44 describes how to print to the same device from both UNIX and UniVerse spoolers. The *lp* model script is modified to respect the UniVerse locking protocol, the *lock.device* file.

Question: How can I print to other machines on my network?

“[Using a Driver for Remote Printing](#)” on page 40 describes how to print to other printers on the network.

Question: How do I keep a log of print jobs and printing problems?

See “[Logging Spooler Activity](#)” on page 28 for details on how to create and maintain spooler log files.

Log files grow. If logging is enabled, remember to purge the files occasionally, or turn off logging when the information is no longer needed.

You can use an alternative to spooler logging for one queue that uses a driver script. The following driver captures start and stop times for a print job and can be modified to do more:

```
echo start 'date' >> logfile
cat -
echo stop 'date' >> logfile
```


Backing Up and Restoring Files

Backup Strategies	12-4
Backing Up Individual Files	12-5
Four Ways to Back Up and Restore Files	12-7
Preserving the Integrity of Your Data	12-7
Backing Up Files.	12-8
Backing Up to Multiple Tapes	12-10
Using T.DUMP to Back Up UniVerse Files	12-12
Using uvbackup to Back Up Files	12-13
Specifying the File List	12-14
Restoring Files	12-16
Choosing the Restore Device	12-16
Checking the Backup Details	12-17
The UVRestore Window	12-19
Choosing What to Restore	12-20
Listing an Index of the Backup Image.	12-21
Specifying How to Restore Files	12-22
Using T.LOAD to Restore UniVerse Files	12-24
Using uvrestore to Restore Files	12-25
Specifying Files and Records to Restore	12-25
Excluding Files to Restore	12-26
Display Options.	12-26
Other Options	12-26
Some UNIX Backup and Restore Commands.	12-28
Using cpio to Back Up and Restore Files.	12-28
Using tar to Back Up and Restore Files	12-28

This chapter first suggests some backup strategies, which you can adopt or modify as needed. It then describes the file backup and restoration procedures.

Users accidentally remove their files. Open files are sometimes lost when the system crashes. It is possible to destroy an entire file system. To protect against such problems, you should regularly back up all disk files to an offline storage medium.

How frequently you back up your files depends on how many files users create or change in a specific time span, and on how much data you can afford to lose.

Backup Strategies

You should design a backup strategy that suits your needs. You must weigh the trade-offs between the possibility of losing data and the amount of time it takes to back up and restore files.

The backup strategies described here assume you are using the UVBackup and UVRestore windows of UniAdmin, or the *uvbackup* and *uvrestore* commands from a UNIX shell or an MS-DOS window.

UniVerse provides three kinds of backup:

- Daily
- Weekly
- Full

In daily and weekly backups, you back up only new or modified files. In full backups, you back up all files.

On an active system you might do a full backup of all files weekly in conjunction with daily backups. On a less active system you might do a full backup monthly, followed by daily and weekly backups.

A full backup copies all specified directories and files to a magnetic tape. You keep this backup tape, perhaps for several months, so you can restore the system to the state it was in during any given week (or month).

FULL backup: all UniVerse and O/S files, whether changed or not

Changes to records in UniVerse hashed files

DAILY backups:

Monday changes
Tuesday changes
Wednesday changes
Thursday changes
Friday changes

WEEKLY backups:

If done on: Saves changes made on:

Monday	Mon.
Tuesday	Mon. Tue.
Wednesday	Mon. Tue. Wed.
Thursday	Mon. Tue. Wed. Thu.
Friday	Mon. Tue. Wed. Thu. Fri.

Three Kinds of Backup

Backing Up Individual Files

On a smaller system with only a few users, you might want users to back up their own important files. Users can use operating system commands (such as *tar* or *cpio* on UNIX systems) or the UniVerse T.DUMP command before they log off or at appropriate intervals. With this strategy you might do a full backup only once a month.

In addition, you might want to back up a particular file, directory, or UniVerse account at a user's request—for example, when a project requiring many related files is completed, you can back up those files and then remove them from the system. Or a user might ask that all files in a certain directory be backed up at some particular checkpoint.

MONTHLY

Full backup

AS NEEDED

User A

User B

User C

Directory A

Directory B

File A

File B

File C

File D

Individual Backups

Four Ways to Back Up and Restore Files

You can choose among four methods to back up and restore files:

- Use the UVBackup and UVRestore windows of UniVerse Admin to back up and restore the entire system, the contents of UniVerse account directories, or individual files.
- Use the T.DUMP and T.LOAD commands to back up and restore selected records from UniVerse files.
- Use the *uvbackup* and *uvrestore* commands from a UNIX shell or an MS-DOS window to back up and restore specified directories, UniVerse files, and operating system files.
- On UNIX systems, use the *cpio* or *tar* command to back up or restore any file on your system.

Preserving the Integrity of Your Data

When you use operating system commands such as *cpio*, *tar*, or *backup* to back up your files, you cannot guarantee either the physical or the logical integrity of your backed-up data, unless no other users are logged on.

When you use the UVBackup window, the *uvbackup* command, or T.DUMP to back up files, the physical integrity of your backed-up data is assured.

If you want to guarantee the physical and the logical integrity of your backed-up data, use transaction logging to back up your files. For more information, see *UniVerse Transaction Logging and Recovery*.

Backing Up Files

To back up files, choose **Backup** from the UniAdmin menu. The UVBackup window appears, as shown in the following example:

UniVerse Backup - colt ():UV

Backup Device

☒ Disk Pathname:

☐ Tape Device:

Block Size: 8192

Backup Type:

☒ Full ☐ Weekly ☐ Daily

Reporting:

☒ File Level ☐ Item Level

Backup Label:

Option: Entire System

You can choose the backup device, what to back up, and the backup type from this window. This window contains all the settings required to perform the backup.

To back up files:

1. Choose one of the following destinations for the backed up files:
 - **Disk Pathname.** Enter a path in the text entry field. You can also use the **Browse** button to search the system for a suitable file.



- **Tape Device.** Choose the tape devices to use. Select one or more devices from the Available Devices list (this list contains all the tape devices defined in the &DEVICE& file) and click **Add >**. The chosen devices are listed in the Selected Devices list. The order of devices in this list determines the order in which they are used. You can reorder devices in this list by dragging and dropping. To remove a device, select it from the **Selected Devices** list and click **< Remove**.

***Note:** Options available in the dialog box change dynamically according to your choice to back up to disk or tape.*

2. Enter a value in the **Block Size** field. It must be a multiple of 512. You can also use the arrows to increase or decrease this setting. The default setting is 8192.
3. Choose the backup type by clicking the appropriate option:
 - **Full**
 - **Weekly**
 - **Daily**
4. Choose what to report on the screen during the backup by clicking the appropriate option:
 - **None.** No reporting is done during the backup. However, you are notified at the start and end of the backup.
 - **File Level.** The paths of the files appear on the screen during backup.
 - **Item Level.** The paths of the files appear, and for UniVerse hashed files, the names of records also appear.

All reported output appears in the UniVerse Command Output window.

5. Enter a short description of the backup in the **Backup Label** field. This description helps to identify the backup image when you restore the data.

***Note:** Do not use single or double quotation marks in the description.*



6. Select what to back up from the Option list:
 - **Entire System.** The whole system is backed up.
 - **All UniVerse Accounts.** All the UniVerse accounts defined in the UV.ACCOUNT file are backed up.
 - **A UniVerse Account & Subdirectories.** The specified UniVerse account is backed up. Choose the account from the Account list.
 - **A File in a UniVerse Account.** The specified file from a UniVerse account is backed up. Select the account and file from the Account and File lists. If you want to back up the file dictionary (and not the file data), select the **Dictionary** check box.
 - **A Directory & Subdirectories.** The specified UNIX or Windows directory and its subdirectories are backed up. Enter the name of the directory in the **Pathname** field, or use **Browse...** to search the system for the directory path.
7. Click **Backup** to start the backup. The UniVerse Command Output window appears.
8. When the backup is completed, click **Close** to close the UniVerse Command Output window.

Backing Up to Multiple Tapes

If your backup does not fit on a single tape, you need to specify how to continue the backup. How multiple tapes are used for backup is determined by the order of devices in the **Selected Devices** list in the **UniVerse Backup** dialog box.

Using a Single Device

If you have a single device in the **Selected Devices** list, the backup pauses when the tape is full. The tape rewinds, and you are prompted to enter the name of a backup device to use.

Complete one of the following steps:

- Continue to use the same device. To use the same tape device, remove the first tape and load the next tape, and enter the device name at the prompt.
- Choose to use a different device. To use a different tape device, make sure the tape is loaded and enter the name of the alternative tape device at the prompt.

Using Multiple Devices

If you have more than one device in the **Selected Devices** list, the backup starts using the first selected device in the list. When a second tape is required, the backup continues using the next selected device in the list.

For this to work successfully, you must make sure that you have ordered the devices correctly in the **Selected Devices** list, and that you have loaded the backup tapes. You can add or remove devices using the **Add >** and **< Remove** buttons, and reorder them by dragging and dropping them to a new location in the list.

Using T.DUMP to Back Up UniVerse Files

T.DUMP lets you write UniVerse files, including data files and dictionaries, from disk to tape. You can specify selected or sorted records in a Retrieve selection or sort expression. Before you use T.DUMP, you must assign the tape drive using ASSIGN. When you are done, release it with UNASSIGN.

The simplest form of the command is as follows:

T.DUMP *filename*

If you do not use any options, T.DUMP writes the specified data file to tape. If you want to save the data file and the file dictionary, run T.DUMP for each file separately. T.DUMP puts an end-of-file mark at the end of each operation.

Because T.DUMP is a Retrieve command, you can specify selected or sorted records with a selection or sort expression. For example, the following statement dumps all paragraph records in your VOC file to tape:

```
>T.DUMP VOC WITH TYPE LIKE "PA..."
```

Using *uvbackup* to Back Up Files

Use *uvbackup* (UNIX) from a UNIX shell or an MS-DOS window to save specified files on a daily, weekly, or comprehensive basis. You can specify files on the command line or from standard input. Output from *uvbackup* goes to standard output. The syntax is as follows:

```
uvbackup { -d | -w | -f } [ -b blksize ] [ cachedetail ] [ -cmdfil filename ] [ delay buffers ] [ -l "labeltext" ] [ limit buffers ] [ rev7 ] [ -rev8 ] [ rev93 ] [ -rev94 ] [ -rev95 ] [ sfile ] [ { t device } ... ] [ -v V ] [ - | pathnames ]
```

Specify each option separately. Precede each option with a minus sign (-).

You must specify whether the backup is daily (*-d*), weekly (*-w*), or full (*-f*). You must also specify the paths of the files you want to back up. You can list the paths as part of the command line, or you can use the *-cmdfil* option to specify the name of a file containing a list of paths of files to back up. On UNIX systems you can also use the hyphen (-) to read paths from standard input.

Use the *-b* option to specify the block size in increments of 512 bytes. The minimum block size is 512. The maximum is defined by the configurable parameter BLKMAX. The default is 8192.

Use the *-l* option to specify text to include in the backup image label.

Use the *-rev7*, *-rev8*, *rev93*, *rev94*, and *-rev95* options to make a backup in formats suitable for restoring to UniVerse Releases 7, 8, 9.3, 9.4, and 9.5 respectively.

Note: The *-rev95* option makes a backup suitable for restoring only to UniVerse Releases 9.5.1 through 9.5.1C.

Use the *-s* option with the *-v* or *-V* option to specify a file for capturing screen output.

Use the *-t* option to specify the device to which to write backup data. Use multiple *-t* options to specify a series of devices.

Use the *-limit* option to specify how many shared memory buffers to use for the backup. Use the *-delay* option to specify how many shared memory buffers to fill before flushing the buffer contents to the backup image on tape or disk. Use the *-cachedetail* option to list details about the shared memory cache.



Use the `-v` option to display paths, or use the `-V` option to display paths and record IDs as they are backed up.

Specifying the File List

To specify the files you want to back up, enter their paths in the *uvbackup* command line. On Windows systems, you can use the `-walk` option of the *uvbackup* command to specify all files in the current directory and in all its subdirectories.

On UNIX systems, to specify the directories you want to back up, use the *find* command. *find* is a general-purpose program that searches for files. *find* locates the files anywhere in the directory tree you specify. Use *find* to do the following:

- Determine which files to back up, and give *uvbackup* a list of the files.
- Find files that have not been accessed in a long time, such as all files modified after a given date.
- Find files larger than a specified size.

find lets you specify a file by any combination of the following parameters:

- Name
- Type
- Permission
- User
- Group
- Size
- Access time
- Modify time

The general syntax of the *find* command is as follows:

find *pathnames options*

See the administrator's manual supplied with your UNIX system for complete information about the *find* command.

Here are some examples in which *find* specifies directories to back up. In each example, output from *find* is piped to the *uvbackup* command. Output from *uvbackup* is redirected to a device file, to which the backup device is attached.

The following example backs up all directories and files on the system:

```
$ find / -print | uvbackup -f -v -l "FULL SYSTEM BACKUP" - > /dev/rmt/0
```

The next example backs up records that have changed in all UniVerse hashed files in the */usr/work* directory and its dependencies:

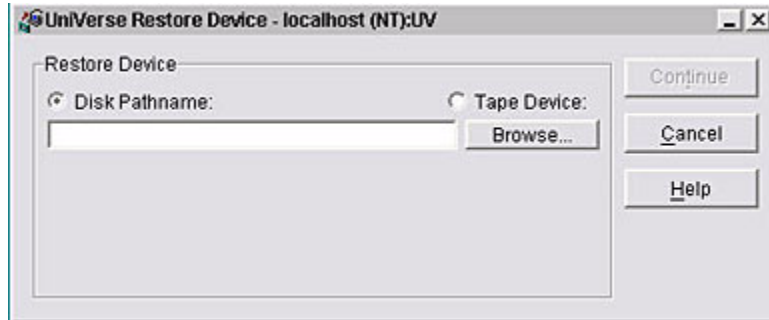
```
$ find /usr/work -print | uvbackup -d -v - > /dev/rmt/0
```

The next example backs up records that have changed in all UniVerse hashed files in the current directory and its dependencies:

```
$ find . -print | uvbackup -w -v - > /dev/rmt/0
```

Restoring Files

To restore files from a backup, choose **Restore** from the UniAdmin menu. The Restore Device window appears, as shown in the following example:



Use this window to choose the restore device and to check the backup details.

When you have chosen the restore device and confirmed that the backup details are correct, the UVRestore window appears. From this window you can:

- Choose the level of reporting
- Choose what to restore
- Edit the restore selection
- Generate an index of the backup
- Specify how to restore the files
- Start the restoration

Choosing the Restore Device

You can restore files from disk or from tape using the **UniVerse Restore Device** dialog box.

To choose the restore device:

1. Click the appropriate device type option:
 - **Disk Pathname.** Enter the name of the file that contains the backup in the text entry field, or click **Browse** to search the system for this file.



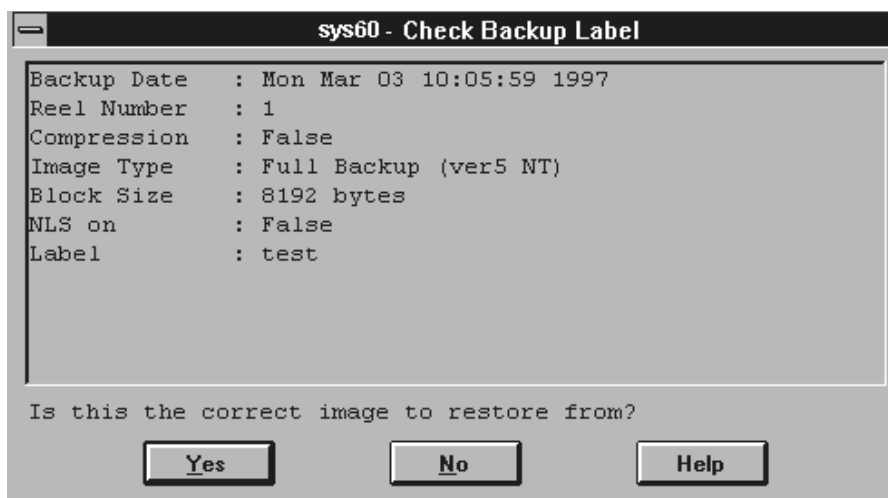
- **Tape Device.** Select one or more devices from the **Available Devices** list (this list contains all the tape devices defined in the &DEVICE& file) and click **Add** . The chosen devices are listed in the **Selected Devices** list. The order of devices in this list determines the order in which they are used during the restoration (if you backed up to multiple tapes). You can reorder devices in this list by dragging and dropping them. To remove a device, select it from the **Selected Devices** list and click **Remove**.

***Note:** Options available in the dialog box change dynamically according to your choice to back up to disk or tape.*

2. If you are restoring from a tape device, check that the backup tapes have been mounted on the selected devices.
3. Click **Continue**. The **Check Backup Label** window appears, and you must check the backup details.

Checking the Backup Details

Before you can restore files, you must check the backup details in the **Check Backup Label** dialog box:



If you are restoring from tape, this window contains the backup details for the first device in the **Selected Devices** list.

This window displays the following backup details:

- **Backup Date.** The date and time the backup was performed.
- **Reel Number.** The tape number. If you backed up to a single tape or to a disk path, the value displayed is 1. If you backed up to multiple tapes, this field displays the number of the tape.
- **Compression.** The level of compression during the backup. This feature is not supported at this release, and so this field always displays **False**.
- **Image Type.** The backup type.
- **Block Size.** The block size (in bytes) used for the backup.
- **NLS on.** The state of NLS when the backup took place.
- **Label.** The backup label. If you backed up the data using the **Backup** option of UniAdmin, this is the text entered in the **Backup Label** field on the UVBackup window.

If the backup details are correct, click **Yes**. The **UniVerse Restore** dialog box appears. If the backup details are incorrect, click **No**. The Restore Device window reappears and you can choose an alternative restore device.

The UVRestore Window

The UVRestore window appears when you confirm the backup details are correct:



Under **Backup Details**, this dialog box lists the restoration source, the date the backup image was made, the type of backup, and the backup image label.

The UVRestore window has four main buttons:

- **Close.** Exits the **Restore** option.
- **Restore.** Starts the restoration.
- **Index.** Displays the index of the backup image.
- **Help.** Invokes the Help system.

Choosing What to Restore

Select what to restore from the **Restore Options** list on the **UniVerse Restore** dialog box. When you select an option, a list of accounts or files to restore appears. To add the name of an account or file, select it and click **Add**. To remove the name of an account or file from the restore selection, select it and click **Remove**.

- **Entire Image.** This option restores the entire backup image.
- **All Accounts in the UV.ACCOUNT File.** This option restores all the accounts listed in the UV.ACCOUNT file. The accounts (except the UV account) are automatically added to the restore selection.
- **Selected UniVerse Accounts.** This option restores accounts selected from those listed in the UV.ACCOUNT file.
- **Files in a UniVerse Account.** This option restores selected files from a UniVerse account. If you want to restore the file dictionary (not the data file), select the **Dictionary** check box before you click **Add**.
- **Records in a UniVerse File.** This option restores selected records in a UniVerse file. Enter the record name in the **Record** field and click **Add**. If you want to add a record from a file dictionary, select the **Dictionary** check box.
- **Selected Directory.** This option restores selected UNIX or Windows directories. To add a directory to the restore selection, enter a directory path in the **Directory** field, or click **Browse** to search the system for a suitable directory. Click **Add**.

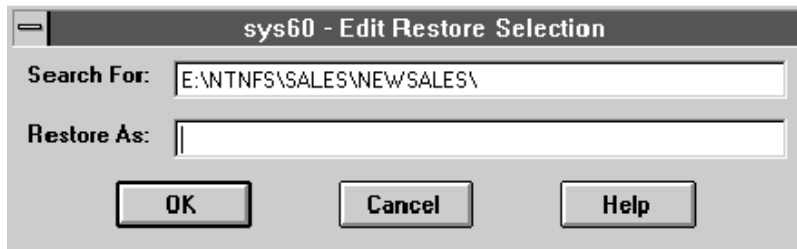


***Note:** The items in the Selection list use paths on the server file system. These paths must match the paths on the tape.*

Editing the Selection List

After you choose what to restore, the record, file, or account details are listed under Selection at the bottom of the window (except the **Entire Image** option). You can add or remove items from this list using the **Add** and **Remove** buttons. You can edit the list using the **Rename** button.

To edit an entry, select it from the list and click **Rename**. The **Edit Restore Selection** dialog box appears:



Use this dialog box to define a different item to restore from the backup, to specify an alternative name or destination for the restored item, or both. This dialog box has two fields:

- **Search For.** Contains the record, file, or account details of the item chosen from the Selection list.
- **Restore As.** This field is empty when the dialog box appears.

To define a different item to restore, edit the **Search For** field. To specify an alternative name or destination for the item to be restored, edit the **Restore As** field.

Click **OK** to update the entry in the Selection list.

If you edited the item in the **Search For** field, the new item replaces the original one in the Selection list.

If you chose an alternative name or destination, the Selection list entry is updated to include an equal sign between the original and new details.



***Note:** The Selection list items use paths on the server file system. You must be certain that these paths match the paths on the tape.*

Listing an Index of the Backup Image

To display an index of the backup image, click the **Index** button on the UVRestore window. The backup index appears in the UniVerse Command Output window.

The backup information displayed in the index depends on the level of reporting you choose:

- **None.** Displays the backup details.
- **File Level.** Displays the paths of the files in the backup.

- **Item Level.** Displays the paths of the files, and for UniVerse hashed files, the record IDs.

Specifying How to Restore Files

You can control how the files are restored by selecting any of these check boxes on the **UniVerse Restore** dialog box:

- **Existing Files Only.** This option restores only files that already exist in the target account. If you do not select this option, all files on the backup are restored, and any files that do not exist in the target account are created.
- **Prompt Before Restoring.** If you choose **Item Level**, this option prompts you to restore each file in turn. To restore a file, enter **Y**. The file is restored, overwriting the existing file on disk. To skip a file, enter **N**. You are then prompted for the next file in the **Selection** list. If you do not select this option, files are restored to the chosen account without prompting.
- **Overwrite Disk Files.** This option determines whether files from the backup overwrite existing files in the chosen account, regardless of the last modification date. If you select this option, the files on the backup overwrite the files on the disk (if the names match). If you clear this option (the default), existing files are overwritten only if the files on the backup have the same (or a more recent) last modification date.

Under **Reporting** you can choose how much detail to display on the screen during the restoration:

- **None.** No reporting is done, but you will be notified at the start and end of the restoration.
- **File Level.** The paths of the files are displayed on the screen.
- **Item Level.** The paths of the files are displayed, and for UniVerse hashed files, record IDs are also displayed.

All the reports appear in the UniVerse Command Output window when restoration starts.

Using T.LOAD to Restore UniVerse Files

T.LOAD lets you restore files from tape that were saved with the T.DUMP command. Before you use T.LOAD, assign the tape drive using ASSIGN. When you are done, release it with UNASSIGN.

The simplest form of the command is as follows:

T.LOAD *filename*

If you do not use any options, T.LOAD copies all the records in the tape file assigned to magnetic tape unit 0 to the data file on disk.

Use the MTU keyword to indicate a magnetic tape unit other than 0. The syntax is MTU *mtu*. The MTU keyword uses a 3-digit numeric value (*mtu*). For details about the MTU keyword, see the *UniVerse User Reference*.

Using *uvrestore* to Restore Files

Use *uvrestore* (UNIX) from a UNIX shell or *uvrestore* (Windows Platforms) from an MS-DOS window to restore specified UniVerse accounts, files, or records saved by a previous *uvbackup* procedure. You can also restore an entire system. The syntax is as follows:

```
uvrestore [-F pathname [=newpathname]] [ R record [=newrecord]] ] ]  
[ X pathname ] [-b blksize] [-i [b]] [+I] [-l] [-L] [-n] [-p]  
[-P n] [-rehash] [-s file] [-startb block] [ { -t device } ... ] [-U]  
[-v | -V] [-verify] [- | imagepath]
```

Specify each option separately. Precede each option with a minus sign.

You must specify the path of the restore image. You can specify the path of a device (such as */dev/rmt/0*) or of a file. Append */** to *filename* to restore all files in a directory. On UNIX systems you can use the hyphen (-) to specify that the path be supplied from standard input.

If you do not specify any other options, *uvrestore* restores the entire backup image.

Specifying Files and Records to Restore

Use one or more *-F* options to restore one or more files. Be sure that *pathname* matches the path saved in the image. For example, if you used the *find* command to specify the file list for *uvbackup*, the ORDERS file might be stored on the image as */usr/SALES/ORDERS* or as *./ORDERS*. To restore the first file, you would use the following *-F* option:

```
-F /usr/SALES/ORDERS
```

To restore the second file, you would use the following:

```
-F ./ORDERS
```

Use one or more *-R* options with one *-F* option to restore one or more records from a hashed file. The record ID must match the record ID saved in the image. For example, to restore records 10006 and 10007 from the ORDERS file, use the following options:

```
-F ./ORDERS -R 10006 -R 10007
```

To restore a record from a type 1 or type 19 file, use the `-F` option (remember that records in type 1 and type 19 files are implemented as operating system files). For example, to restore the program MYPROG from the BP file, use the following option:

`-F ./BP/MYPROG`

Excluding Files to Restore

Use one or more `-X` options to exclude one or more files from being restored.

Display Options

Use the `-i` option to list the contents of the backup image without restoring anything. Use the `-ib` option to show the blocks in addition to the paths. (Use `-ib` with the `-startb` option to start restoring from a particular block.) Use the `-L` option to display the image label without restoring anything.

Use the `-v` option to list paths, or use the `-V` option to list paths and record IDs as they are restored. Use the `-l` option to display the image label before restoring files.

Other Options

Specifying the Block Size. Use the `-b` option to specify the block size. The minimum block size is 512, the maximum is defined by the configurable parameter BLKMAX. The default is 8192.

Specifying the Backup Image Source

Use the `-t` option to specify the device or file from which to read the backup data. You can use multiple `-t` options to specify a series of devices or files.

Specifying the Starting Block. Use the `-startb` option to specify the block to start restoring from. (Use `-startb` with the `-ib` option to start restoring from a particular block.)

Restoration Options. Use the *-n* option when you are restoring from a full backup image to prevent *uvrestore* from creating files automatically if they do not exist on disk. Use the *-rehash* option when you are restoring from a full backup image to force the rehashing of records as they are restored. Use the *-U* option if you want *uvrestore* to overwrite disk files with the same names as those being restored. On Windows systems, use the *-nodrv* option to strip the drive letter from restored paths so you can restore files onto a different disk.

Having uvrestore Prompt You

Use the *-p* option if you want *uvrestore* to prompt you before restoring each file or record.

For full details about the *uvrestore* (UNIX) and the *uvrestore* (Windows Platforms) commands, see the *UniVerse User Reference*.

Some UNIX Backup and Restore Commands

UNIX programs you can use for backup and restoration are *cpio*(1) and *tar*(1). You can use *cpio* to back up selected files or all files on the system. *tar* is more useful for backing up a limited set of files rather than for complete dumps. For example, users might back up their own files onto a private storage medium using these commands. For complete information about how to use *cpio* and *tar*, see the UNIX documentation supplied with your system.

Using *cpio* to Back Up and Restore Files

The *cpio* command copies all files in a given list or input pipeline to an archival device. Backup device names vary from system to system. For correct device names, see the administrator's guide for your UNIX system. If the backup requires more than one backup volume, the system asks you to mount another volume when the first is full.

You can use the *find* command to specify which files you want to back up. For more information, see [“Specifying the File List”](#) on page 13.

The following guidelines apply:

- Label the backup volume before performing the backup.
- All other users should be off the system when you perform the backup. This ensures the physical integrity of the files you are backing up.

If your backup volume is a diskette, format the diskette.

You can also use *cpio* to restore any directories or files backed up with *cpio*. When you restore a *cpio* backup, the contents of the backup volume overwrite the contents of existing directories and files.

Using *tar* to Back Up and Restore Files

The UNIX *tar*(1) command copies a specified list of files to or from a tape in ASCII. A *tar* tape is the best means of transferring files from system to system. You can specify the following options with the *tar* command:

- The *c* or *r* option writes to a tape.

- The *x* option reads from a tape.

If you specify the name of a directory in the file list, *tar* copies all files in that directory and all of its subdirectories. For example, the following command performs a complete backup of */usr*:

```
% tar c /usr
```

Note: Most versions of *tar* do not support multivolume backups.



Managing Data Replication

Replication	13-4
Hot Standby	13-5
Setting Up Data Replication	13-6
The Replication Window	13-8
Menu Bar.	13-8
Toolbar	13-10
Left Pane	13-10
Right Pane	13-11
Configuring and Managing Data Replication	13-12
Managing a Publishing System	13-13
Configuring the Publishing System	13-13
Starting and Stopping the Publishing System	13-15
Publishing Files.	13-15
Managing a Subscribing System	13-24
Configuring the Subscribing System	13-24
Starting and Stopping the Subscribing System	13-25
Creating a Subscriber's List of Publishing Systems	13-25
Subscribing Files	13-27
Managing Hot Standby Operations	13-35
Configuring a Hot Standby Subscriber	13-35
Turning On Fail-Over Mode	13-37
Reconciling the Hot Standby with the Publisher	13-37
Some Restrictions	13-40
What to Do When Disk Space Fills Up.	13-41
Removing Obsolete Replication Log Files	13-42
What to Do When Replication Fails.	13-43

Beta Beta

The UniVerse Data Replication service provides two operations:

- Replication
- Hot standby

Replication maintains one or more read-only copies of UniVerse files for data backup or distribution. The copies can be on one or more computer systems.

Hot standby is a special case of replication, in which a system that maintains copies of active UniVerse files can provide read/write versions of the replicated files to users, should the original files be unavailable.

Note: Replication works only with files created or resized on Release 9.4 or later of UniVerse. To replicate files created on older systems, resize the files first.





Replication

UniVerse data replication provides an automatic and reliable way to deliver read-only copies of UniVerse files to other UniVerse systems.

The system where the source data resides is called the *publisher*. A system requesting copies of file updates from the publisher is called a *subscriber*.

Subscribing files on a subscriber are read-only. Users logged on to the subscriber cannot modify the replicated data.

Note: A publisher can also be a subscriber of another publisher's files. However, a publisher cannot publish replicated (subscribing) files.

Hot Standby

You can configure data replication to create hot standby (also called hot backup) functionality, with the publisher acting as the primary UniVerse server and the subscriber acting as the development system or dormant backup system. The hot standby subscriber must be a different computer system from the publisher.

If the publisher's disk integrity is compromised or the hardware is unusable, you use UniVerse Admin to switch from using the publisher to using the hot standby system.

If a crash occurs on the publisher, you use UniVerse Admin on the hot standby system to switch to fail-over mode, thus making the subscriber read/write-enabled. Users can then log on to the hot standby system and run their applications as usual. Later, the administrator can reconcile the originally published files on the publisher with all updates and changes made to the files on the hot standby system.



Setting Up Data Replication

Before you can configure a subscriber, you must set up the replicated (subscribing) database on it. Then you configure the publisher and all subscribers. Finally, you start publishing on the publisher and subscribing on the subscribers.

Note: *You must configure the publisher and all subscribers before you start publishing and subscribing, otherwise the replication system will start up in the crashed state.*

To set up a data replication system:

1. Before setting up and configuring your publishing and subscribing systems, ensure that no users can access the files you intend to publish.
2. On the publisher, resize all files you want to publish that were created on Release 9.3 or earlier of UniVerse.
3. On the subscriber, create the UniVerse accounts to contain the replicated (subscribing) files.
4. On the subscriber, create the subscribing UniVerse files that will store the replicated data. Permissible file types are types 2 through 18 (static hashed files), type 25 (B-tree files), type 30 (dynamic hashed files), and distributed files. These files need not have the same file names as the corresponding files on the publisher, nor need they be of the same file type. The structures of distributed files on the publisher and subscriber should match.



Note: *You cannot replicate type 1 or type 19 files, or secondary indexes.*

5. Before configuring and starting the publishing system, ensure that:
 - Data in the subscriber's files is identical to data in the publisher's files.
 - No users can access the replicated files.
6. Configure the publisher and all subscribers.



7. Set the value of the UDRMODE configurable parameter to 1, then stop and restart UniVerse.
 - To set UDRMODE to 1, see [Changing Configurable Parameter Values](#) in Chapter 4, “[Configurable UniVerse Parameters.](#)”
 - To unload shared memory, shut down UniVerse. For the procedure, see Chapter 3, “[System Startup and Shutdown.](#)”
 - Start UniVerse again. For the procedure, see Chapter 3, “[System Startup and Shutdown.](#)”

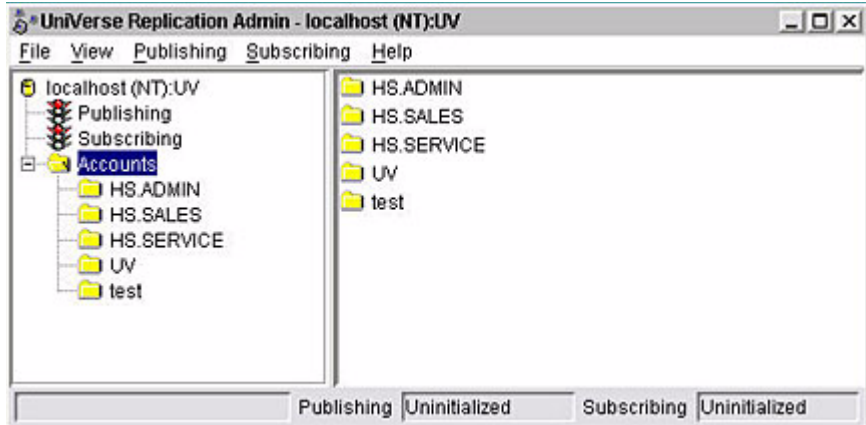
***Note:** After you configure publishing and subscribing, no published or subscribing files can be modified until you start publishing and subscribing on all systems.*

8. Start publishing on the publisher, and start subscribing on all subscribers.

The following sections describe how to use the Replication window of UniAdmin to configure and manage data replication.

The Replication Window

To set up and manage replication services, choose **Replication** from the UniAdmin menu. The Replication window appears, as shown in the following example:



The Replication window has the following components:

- Menu bar
- Toolbar
- Left pane
- Right pane
- Status bar

Menu Bar

The menu bar has five menus:

- File menu
- View menu
- Publishing menu
- Subscribing menu
- Help menu

File Menu

The File menu has two options:

- **Repair.** Repairs a damaged file.
- **Close.** Exits the Replication window.

View Menu

The View menu has seven options:

- **Toolbar.** Hides or displays the toolbar.
- **Status Bar.** Hides or displays the status bar.
- **Large Icons, Small Icons, List, and Details** change the way the file list appears.
- **Refresh.** Refreshes the file list display.

Publishing Menu

The Publishing menu has eight options:

- **Publish.** Publishes the selected file.
- **Unpublish.** Unpublishes the selected file.
- **Properties.** Displays details about a published file.
- **Start Publishing.** Starts the publishing system.
- **Stop Publishing.** Stops the publishing system.
- **Resume Publishing.** Restarts the publishing system.
- **Configure.** Specifies log file and information file settings.
- **Sync.** Updates the publisher with hot standby data.

Subscribing Menu

The Subscribing menu has eight options:

- **Subscribe.** Subscribes the selected file.
- **Unsubscribe.** Unsubscribes the selected file.
- **Properties.** Displays details about a subscribing file.

- **Start Subscribing.** Starts the subscribing system.
- **Stop Subscribing.** Stops the subscribing system.
- **Configure.** Specifies information file settings.
- **Systems.** Specifies publishers to subscribe to.
- **Fail Over.** Makes subscribing files writable.

Help Menu

The Help menu has two options:

- **Contents.** Displays UniAdmin online help.
- **Replication.** Displays online help for data replication.

Toolbar

The toolbar has eight buttons:

- **Publish.** Publishes the selected file.
- **Subscribe.** Subscribes the selected file.
- **Unpublish/Unsubscribe.** Unpublishes or unsubscribes the selected file.
- **Properties.** Displays details about the selected file.
- The next four buttons correspond to the four options on the View menu that control the way the file list is displayed.

Left Pane

The left pane lists the following:

- The host name of the computer system to which UniAdmin is connected.
- A list of all accounts containing currently published UniVerse files. If the traffic signal icon is red, publishing is uninitialized. If it is green, publishing is enabled.
- A list of all accounts containing currently subscribing UniVerse files. If the traffic signal icon is red, subscribing is uninitialized. If it is green, subscribing is enabled.

- A list of all UniVerse accounts on the system to which UniAdmin is connected.

Right Pane

The right pane lists the following:

- If **Publishing** or **Subscribing** is selected in the left pane, all UniVerse accounts with published or subscribing files
- If **Accounts** is selected in the left pane, all UniVerse accounts with publishable or subscribable files
- If an account under **Publishing** is selected in the left pane, all currently published files in that account
- If an account under **Subscribing** is selected in the left pane, all currently subscribing files in that account
- If an account under **Accounts** is selected in the left pane, all UniVerse files in that account that can be published or subscribing

Configuring and Managing Data Replication

There are two parts to data replication:

- A publishing system keeps track of updates to a published database and makes information about those updates available to subscribing systems across the network. You replicate read-only copies of published UniVerse files on subscribing systems.
- A subscribing system receives information about updates on published source files across the network.

***Note:** A special case of subscribing called hot standby lets you track updates to published UniVerse files on a subscribing system and make those subscribing copies into fully read/write files in the event that the primary publishing system becomes temporarily unavailable.*



Managing a Publishing System

You can publish the following file types:

- Types 2 through 18 (static hashed files)
- Type 25 files (B-tree files)
- Type 30 files (dynamic hashed files)
- Multiple data files
- Distributed files

You cannot publish the following:

- Type 1 or type 19 files
- Secondary indexes

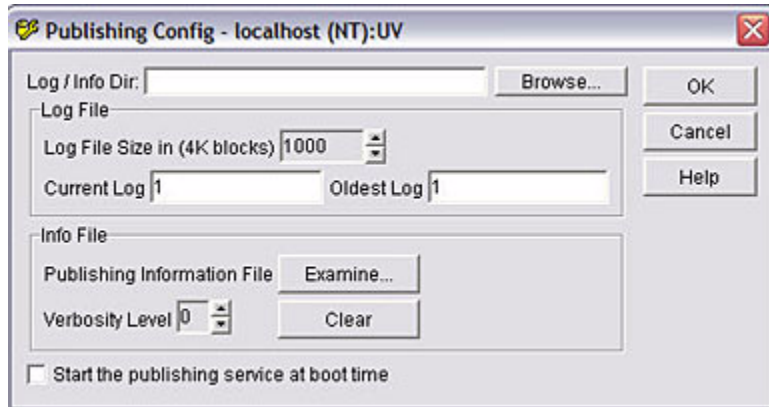
Configuring the Publishing System

You must configure a publishing system before you can publish its files. You can configure the following options for publishing systems:

- The location of the directory containing replication log files
- The size of the replication log files
- The number of the current replication log file
- Whether or not publishing services start up when UniVerse starts

Complete the following steps to configure a publishing system:

1. Choose **Configure** from the **Publishing** menu. The **Publishing Config** dialog box appears, as shown in the following example:



2. Specify the log file directory by entering the path of an existing directory in the **Log/Info Dir.** field, or click **Browse** to search the system for a suitable directory.

The log directory contains the log files that store updates to published files before they are sent to subscribers for replication. The log directory also contains an information log file.

***Note:** Log files are created automatically.*

3. Specify the size of replication log files in 4-kilobyte blocks. The default is 1000 blocks.
(Optional) Specify the number of the current log file. If UniVerse Admin finds an existing log file, it increments its number.
4. (Optional) Specify the number of the oldest log file. To determine which is the oldest log file:
 - In an MS-DOS window, change to the Log/Info directory.
 - List all files named uvdrln, where *n* is a unique number.
 - The oldest log file is the uvdrln file where *n* is the lowest number in the set.



5. (Optional) Set the verbosity level for the publishing information file. It can be a number from 0 to 9. The default is 0.

***Note:** Use the publishing information file only for debugging. For details about the publishing information file, see “[The Publishing Information File](#)” on page 22.*

6. (Optional) To start the publishing service at boot time, select the check box at the bottom of the dialog box.
7. Click **OK** to save your changes and exit the **Publishing Config** dialog box. Click **Cancel** to exit the **Publishing Config** dialog box without saving changes.

Starting and Stopping the Publishing System

To start the publishing system, choose **Start Publishing** from the **Publishing** menu. The status bar displays **Enabled**.

To stop the publishing system, choose **Stop Publishing** from the **Publishing** menu. The status bar displays **Uninitialized**.

Publishing Files

You can publish:

- One file at a time
- Selected files
- All files in an account

Publishing One UniVerse File

Complete the following steps to publish one UniVerse file in an account:

1. From the left pane, double-click **Accounts** (or click the + sign before it), then select one of the accounts listed. A list of UniVerse files you can replicate appears in the right pane.

2. From the right pane, select a file, then complete one of the following steps:
 - Choose **Publish** from the **Publishing** menu.
 - Click **Publish** on the toolbar.
 - Right-click the file and choose **Publish** from the menu

The Publish dialog box appears with the **Filename**, **Dictname**, **Account**, and **File Type** fields filled in for you:

garcia- Publish

Published File

Filename: ☒ Publish

Dictname: ☐ Publish

Account:

Desc:

FileType:

Access List

System Name

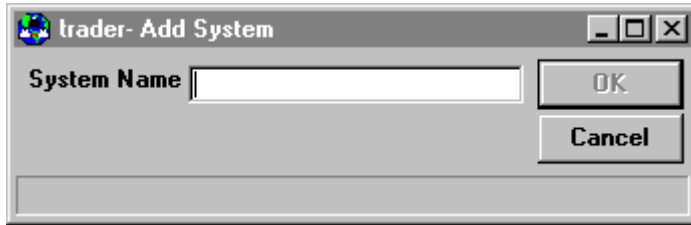
Subscribing Files

File	Account	Type

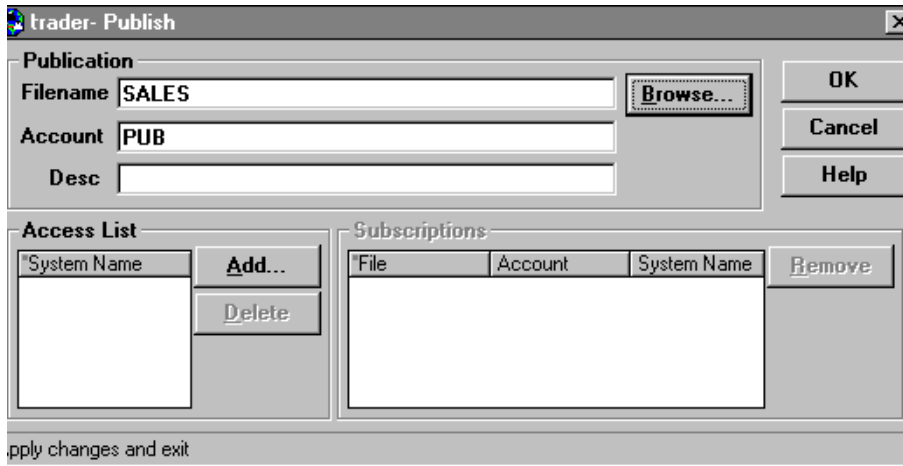
Buttons: **Publish**, **Cancel**, **Help**, **Add...**, **Delete**, **Remove**

3. (Optional) Select **Publish** next to **Dictname** if you want to replicate the file dictionary as well as the data file.
4. (Optional) Enter a description of the file in the **Desc** field.
5. By default all subscribing systems can access any files you publish. To restrict access to a finite list of published files, add a system name to the **Access List**:

6. Click **Add...** . The Add System dialog box appears.



- Enter a system name, then click **OK**.
- Repeat steps n and n until you specify all systems you want to have access to the publication.
- Click **OK** in the Add System dialog box to return to the Publish dialog box.



7. Click **Publish** to publish the file and exit the Publish dialog box. The letter P appears on the file icon in the Replication window.
Click **Cancel** to exit the Publish dialog box without publishing the file.

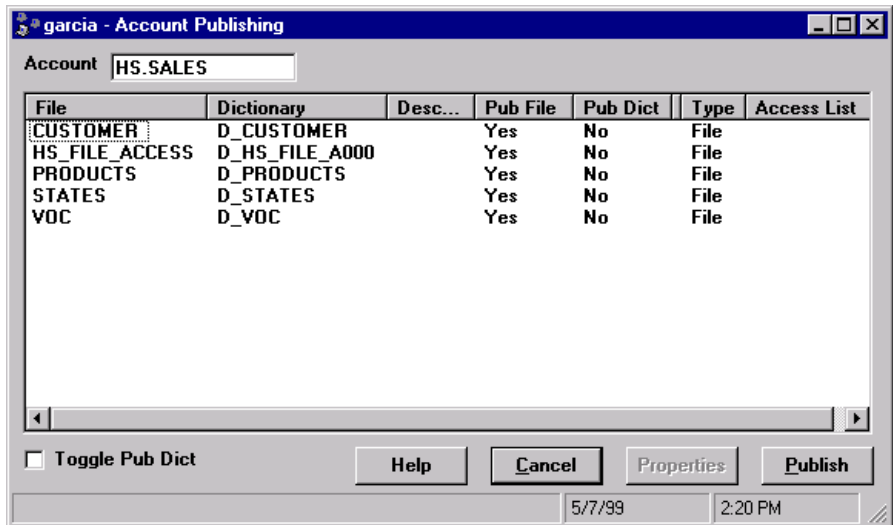
Publishing Multiple UniVerse Files

To publish selected UniVerse files or all files in an account:

1. Do one of the following:

- To publish selected files:
 - From the left pane, double-click **Accounts** (or click the + sign before it).
 - Select one of the accounts listed. A list of UniVerse files you can publish appears in the right pane.
 - To select a range of files, select the first file in the range, then hold down the **Shift** key and select the last file in the range.
 - To select random files, hold down the **Ctrl** key and select the files you want to publish.
 - To publish all files in an account:
 - From the left pane, select **Accounts**. A list of accounts appears in the right pane.
 - From the right pane, select the account whose files you want to publish.
2. Do one of the following:
- Choose **Publish** from the Publishing menu.
 - Click **Publish** on the toolbar.
 - From the right pane, right-click the selected files or the selected account, then choose **Publish** from the menu.

The Account Publishing dialog box appears:



1. You can now do any of the following:
 - Click **Publish** to publish all selected files.
 - Select **Toggle Pub Dict** to publish all selected data files and their file dictionaries.
 - Double-click a file in the list, or select a file and click **Properties** to display the Publish dialog box. This lets you customize publication of the selected file. You can modify the following:
 - The file description
 - Whether or not to publish the file dictionary
 - The list of systems that can subscribe to this file
 - Follow the procedure described in [Publishing One UniVerse File](#).
 - Click **Cancel** to exit the Account Publishing dialog box without publishing any files.

Viewing and Modifying Published Files

To view a list of currently published files:

1. From the left pane, double-click **Publishing** (or click the + sign before it), then select one of the accounts listed. A list of published UniVerse files in the account appears in the right pane. The Published Dictionary column lists any published file dictionaries.

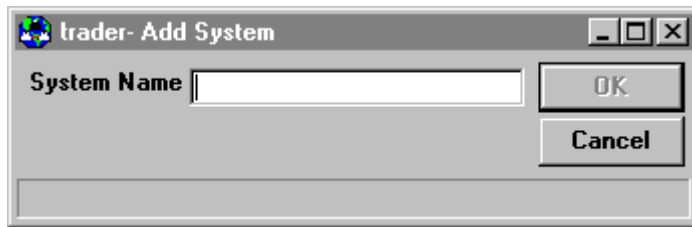
File type is one of the following:

Type	Description
File	Hashed file (types 2 through 18), B-tree file (type 25), dynamic file (type 30).
Q Pointer	File referenced by a Q-pointer in the VOC file.

Published UniVerse File Types

2. From the right pane, right-click a file and choose **Properties** from the menu to view information about the file. The Publish dialog box appears. This dialog box displays information about the published file. You can modify the following:
 - Publish file dictionary (check box)
 - File description

- Access list
- Subscriptions list
- 3. Select **Publish** next to **Dictname** to replicate the file dictionary as well as the data file.
- 4. Enter a description of the file in the **Desc** field.
- 5. By default all subscribing systems can access any files you publish. To restrict access to a finite list of published files, add a system name to the **Access List**:
 - Click **Add...** . The Add System dialog box appears.



- Enter a system name, then click **OK**.
- Repeat steps n and n until you specify all systems you want to have access to the publication.
- Click **OK** to exit the Add System dialog box.

6. All files that subscribe to this file are listed in the **Subscribing Files** list. To unsubscribe a file, select it and click **Remove**. If there are no other subscribing files, logging of file updates ceases immediately.

The screenshot shows a Windows-style dialog box titled "trader- Publish". It is divided into several sections. The top section, "Publication", contains three text input fields: "Filename" with the value "SALES", "Account" with the value "PUB", and "Desc" which is empty. To the right of the "Filename" field is a "Browse..." button. To the right of the input fields are three stacked buttons: "OK", "Cancel", and "Help". Below the "Publication" section are two more sections. The "Access List" section on the left contains a table with one header "System Name" and an empty body. To the right of this table are two buttons, "Add..." and "Delete". The "Subscriptions" section on the right contains a table with four columns: "File", "Account", "System Name", and "Remove". The "File", "Account", and "System Name" columns have empty rows, while the "Remove" column has a button. At the bottom of the dialog box is a button labeled "Apply changes and exit".

7. Click **OK** to save your changes and exit the Publish dialog box. Click **Cancel** to exit the Publish dialog box without saving changes.

Unpublishing Files

When you unpublish a file, all subscriptions to that file are cancelled. To unpublish files:

Do one of the following:

- To select individual files:
 - From the left pane, double-click **Publishing** or **Accounts** (or click the + sign before it).
 - Select one of the accounts listed. A list of published UniVerse files in that account appears in the right pane.
 - To select a range of files, select the first file in the range, then hold down the **Shift** key and select the last file in the range.
- To select random files, hold down the **Ctrl** key and select the files you want to unpublish.

- To select all files in an account:
 - From the left pane, select **Publishing** or **Accounts**. A list of accounts appears in the right pane.
 - From the right pane, select the account whose files you want to unpublish.
- 1. Do one of the following:
 - Choose **Unpublish** from the Publishing menu.
 - Click **Unpublish/Unsubscribe** on the toolbar.
 - From the right pane, right-click the selected files or the selected account, then choose **Unpublish** from the menu.
- 2. When asked if you are sure you want to unpublish the files, click **Yes** or **No**. If you click **Yes**, all selected files are unpublished, the filenames are removed from the Publishing list, and all subscriptions to the files are cancelled.

The Publishing Information File

The publishing information file is useful for debugging. Normally you do not need to log information to this file. We recommend you log information to this file only on the advice of an IBM support specialist.

To activate logging to the publishing information file, set the verbosity level to a number between 1 and 9. To deactivate logging, set the verbosity level to 0. 0 is the default setting.

Setting the Verbosity Level

To set the verbosity level:

1. Choose **Configure** from the Publishing menu. The Publishing Config dialog box appears.
2. Under **Info File**, set the **Verbosity Level** to a number between 0 and 9. The higher the verbosity level, the more information is logged.
3. Click **OK** to exit the Publishing Config dialog box.

Viewing Information Files

To view the contents of the publishing information file:

1. Choose **Configure** from the Publishing menu. The Publishing Config dialog box appears.
2. Under **Info File**, click **Examine...** . The Publishing Info File window appears, listing the contents of the publishing information file.

Clearing Information Files

To clear the contents of the publishing information file:

1. Choose **Configure** from the Publishing menu. The Publishing Config dialog box appears.
2. Under **Info File**, click **Clear**. The contents of the publishing information file are cleared.

Managing a Subscribing System

The subscribing system handles the configuring and enabling of subscribing operations and the distributing of replicated records.

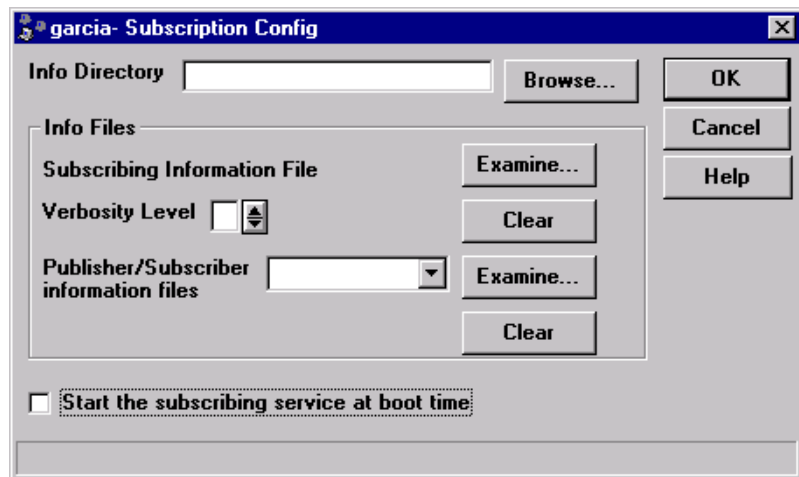
Configuring the Subscribing System

You must configure the subscribing system before you can subscribe to published files. You can configure the following options for the subscribing system:

- The location of the subscribing information files
- Whether or not the subscribing service starts up when UniVerse starts

To configure a subscribing system:

1. Choose **Configure** from the Subscribing menu. The Subscription Config dialog box appears:



2. Specify the information file directory by entering the path of an existing directory in the **Info Directory** field. You can also use **Browse...** to search the system for a suitable directory. The Info directory contains the subscription information files.



3. (Optional) Set the verbosity level for the subscribing information files. It can be a number from 0 to 9. The default is 0.

***Note:** Use the subscribing information files only for debugging. For details about these files, see [Subscribing Information Files](#).*

4. (Optional) To choose whether to start the subscribing service at boot time, select the check box at the bottom of the window.
5. Click **OK** to save your changes and exit the Subscription Config dialog box. Click **Cancel** to exit the Subscription Config dialog box without saving changes.

Starting and Stopping the Subscribing System

To start the subscribing system, choose **Start Subscribing** from the Subscribing menu. The status bar displays **Enabled**.

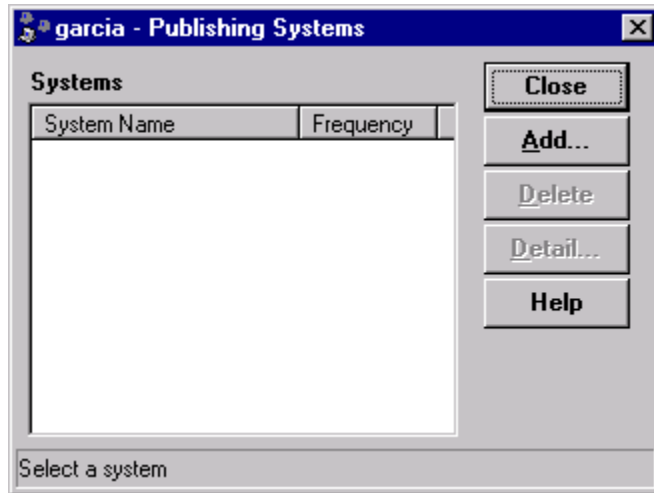
To stop the subscribing system, choose **Stop Subscribing** from the Subscribing menu. The status bar displayed **Uninitialized**.

Creating a Subscriber's List of Publishing Systems

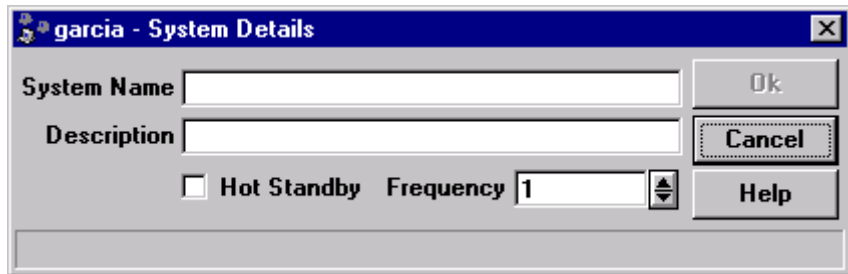
Before a subscribing system can subscribe to published files, you must first create a list of publishing systems available to the subscriber. Next you subscribe to the files published by these systems.

To add a publishing system to the subscriber's list of publishing systems:

1. Choose **Systems** from the Subscribing menu. The Publishing Systems dialog box appears:

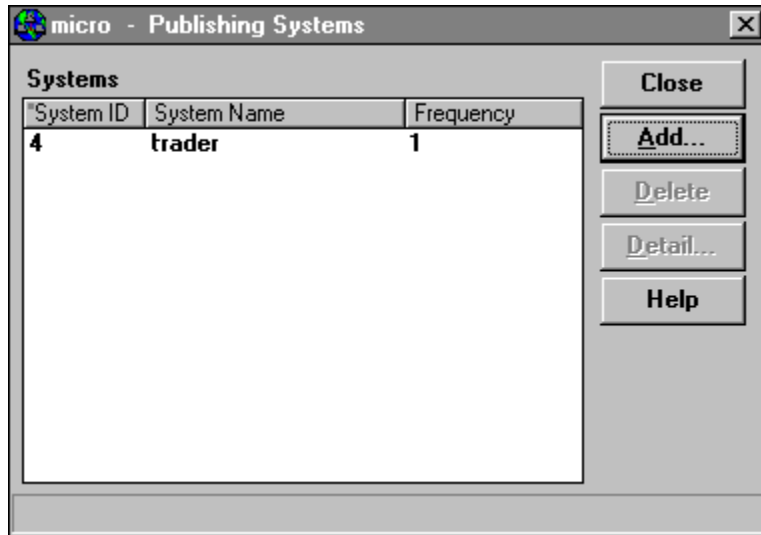


2. Click **Add...** . The System Details dialog box appears:



3. Enter the name of the publishing system in the **System Name** field.
4. (Optional) Enter a description of the system in the **Description** field.
5. Select a file update frequency from the **Frequency** field. This is the number of minutes to elapse between logging updates. It can be between 1 and 1,440 minutes (24 hours). 1 is the default.
6. (Optional) If you want the subscriber to be the hot standby system for this publishing system, select **Hot Standby**. For information about hot standby operations, see [Configuring a Hot Standby Subscriber](#).

7. Click **Ok**. The publishing system is added to the Systems list.



8. Repeat steps 2 through 7 to add the names of all publishing systems whose files you want to subscribe to.
9. When you finish, click **Close** to exit the Publishing Systems dialog box.

Subscribing Files

You can subscribe:

- One file at a time
- Selected files
- All files in an account

Subscribing One UniVerse File

To subscribe one UniVerse file in an account:

1. From the left pane, double-click **Accounts** (or click the + sign before it), then select one of the accounts listed. A list of UniVerse files you can subscribe appears in the right pane.
2. From the right pane, select a file, then do one of the following:

- Choose **Subscribe** from the Subscribing menu.
- Click **Subscribe** on the toolbar.
- Right-click the file and choose **Subscribe** from the menu.

The Subscribe dialog box appears with the **Filename**, **Dictname**, **Account**, and **File Type** fields filled in for you:

The screenshot shows a Windows-style dialog box titled "garcia - Subscribe". It is divided into two main sections. The top section, "Subscribing File", contains five text input fields: "Filename" (containing "CUSTOMER"), "Dictname" (containing "D_CUSTOMER"), "Account" (containing "HS.SALES"), "Desc" (empty), and "File Type" (containing "File"). To the right of these fields are two checkboxes, both labeled "Subscribe"; the top checkbox is checked. To the right of the entire dialog box are three buttons: "Subscribe", "Cancel", and "Help". The bottom section, "Published File", contains five empty text input fields: "System Name", "Filename", "Dictname", "Account", and "Desc". To the right of the "System Name" field is a button labeled "Specify...".

3. (Optional) Select **Subscribe** next to **Dictname** if you want to subscribe the file dictionary as well as the data file.
4. (Optional) Enter a description of the file in the **Desc** field.
5. Click **Specify...** to display the Available Publications dialog box.
 - Double-click the name of a publisher to list all accounts with published files.
 - Double-click the name of an account to list all its published files.
 - Click the name of the file to which you want to subscribe. Information about the published file appears at the bottom of the Subscribe dialog box.
6. Click **OK** to subscribe the file and exit the Subscribe dialog box. The letter S appears on the file icon in the Replication window.

Click **Cancel** to exit the Subscribe dialog box without subscribing the file.

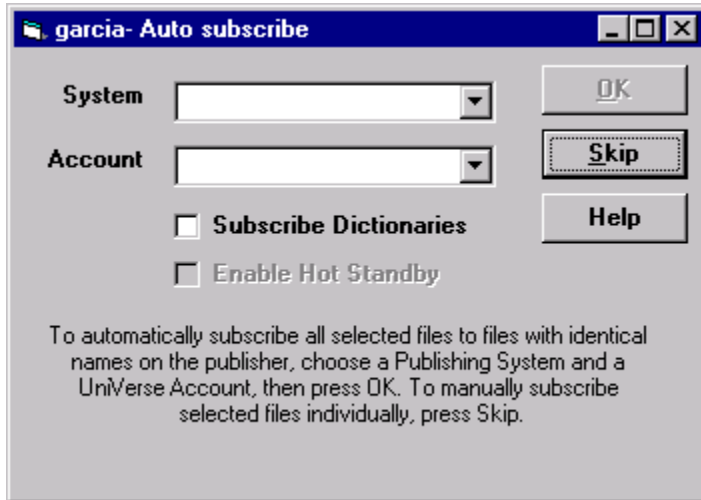
Subscribing Multiple UniVerse Files

To subscribe selected UniVerse files or all files in an account:

1. Do one of the following:
 - To subscribe selected files:
 - From the left pane, double-click **Accounts** (or click the + sign before it).
 - Select one of the accounts listed. A list of UniVerse files you can subscribe appears in the right pane.
 - To select a range of files, select the first file in the range, then hold down the **Shift** key and select the last file in the range.
 - To select random files, hold down the **Ctrl** key and select the files you want to subscribe.
 - To subscribe all files in an account:
 - From the left pane, select **Accounts**. A list of accounts appears in the right pane.
 - From the right pane, select the account whose files you want to subscribe.
2. Do one of the following:
 - Choose **Subscribe** from the Subscribing menu.
 - Click **Subscribe** on the toolbar.

- From the right pane, right-click the selected files or the selected account, then choose **Subscribe** from the menu.

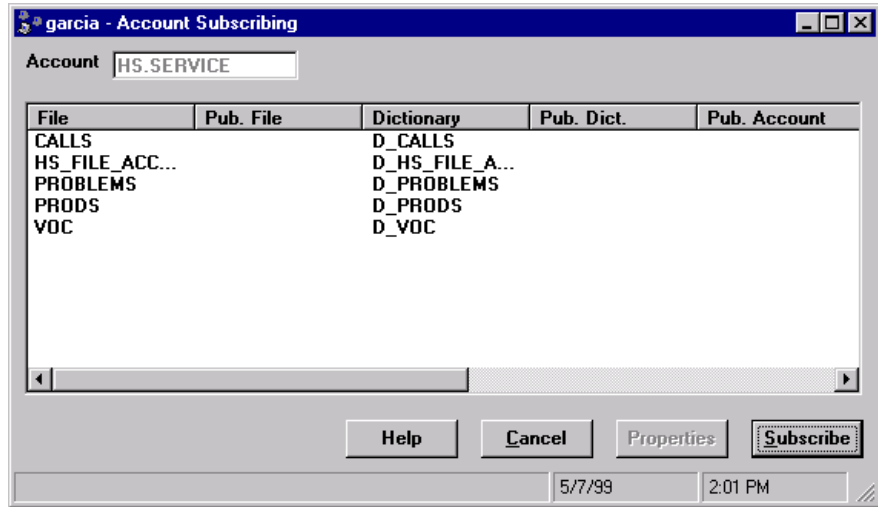
The Auto subscribe dialog box appears:



***Note:** Use this dialog box only if the files on the publisher are identical to the files on the subscriber. If they are not, press **Skip** and proceed to step 7.*

3. Select a publishing system.
4. Select the UniVerse account containing the files to which you want to subscribe.
5. (Optional) Select **Subscribe Dictionaries** to subscribe all selected data files and their file dictionaries.
6. (Optional) Select **Enable Hot Standby** if your subscriber is to be a hot standby system.

7. Click **OK**. The Account Subscribing dialog box appears:



8. You can now do any of the following:

- Click **Subscribe** to subscribe all selected files.
- Double-click a file in the list, or select a file and click **Properties** to display the Subscribe dialog box. This lets you customize subscription of the selected file. You can modify the following:
 - The file description
 - Whether or not to subscribe the file dictionary
 - The published file this file subscribes to
- Follow the procedure described in [Subscribing One UniVerse File](#)
- Click **Cancel** to exit the Account Subscribing dialog box without subscribing any files

Viewing and Modifying Subscribing Files

To view a list of currently subscribing files:

1. From the left pane, double-click **Subscribing** or **Accounts** (or click the + sign before it), then select one of the accounts listed. A list of subscribing UniVerse files in the account appears in the right pane. The Subscribing Dict column lists any subscribing file dictionaries. The Pub. System column lists the name of the system where the published file resides.

File type is one of the following:

Type	Description
File	Hashed file (types 2 through 18), B-tree file (type 25), dynamic file (type 30).
Q Pointer	File referenced by a Q-pointer in the VOC file.

Subscribing UniVerse File Types

2. From the right pane, double-click a file to view information about the file. The Subscribe dialog box appears. This dialog box displays information about the subscribing file.
3. (Optional) Select **Publish** next to **Dictname** if you want to subscribe the file dictionary as well as the data file.
4. (Optional) Enter or modify the description of the file in the **Desc** field.
5. Click **OK** to save your changes and exit the Subscribing dialog box. Click **Cancel** to exit the Subscribing dialog box without saving changes.

Unsubscribing Files

To unsubscribe files:

6. Do one of the following:
 - To select individual files:
 - From the left pane, double-click **Subscribing** or **Accounts** (or click the + sign before it).
 - Select one of the accounts listed. A list of subscribing UniVerse files in that account appears in the right pane.

To select a range of files, select the first file in the range, then hold down the **Shift** key and select the last file in the range.

To select random files, hold down the **Ctrl** key and select the files you want to unsubscribe.

- To select all files in an account:
 - From the left pane, select **Subscribing** or **Accounts**. A list of accounts appears in the right pane.
 - From the right pane, select the account whose files you want to unsubscribe.
- 7. Do one of the following:
 - Choose **Unsubscribe** from the Subscribing menu.
 - Click **Unpublish/Unsubscribe** on the toolbar.
 - From the right pane, right-click the selected files or the selected account, then choose **Unsubscribe** from the menu.
- 8. When asked if you are sure you want to unsubscribe the selected files, click **Yes** or **No**. If you click **Yes**, the files are unsubscribed and the letter S is removed from the file icon.

Subscribing Information Files

Information files are useful for debugging. The subscribing system has two information files:

- Subscribing information file, which logs information about the subscribing system.
- Publisher/Subscriber information file, which logs information about connections between subscribing and publishing systems.

Normally you do not need to log information to these information files. We recommend you log information to these files only on the advice of an IBM support specialist.

To activate logging to the subscribing information files, set the verbosity level to a number between 1 and 9. To deactivate logging, set the verbosity level to 0. 0 is the default setting.

Setting the Verbosity Level

To set the verbosity level:

1. Choose **Configure** from the Subscribing menu. The Subscribing Config dialog box appears.

2. Under **Info File**, set the **Verbosity Level** to a number between 0 and 9. The higher the verbosity level, the more information is logged.
3. Click **OK** to exit the Subscribing Config dialog box.

Viewing Information Files

To view the contents of the subscribing information file:

1. Choose **Configure** from the Subscribing menu. The Subscribing Config dialog box appears.
2. Under **Info File**, click **Examine...** . The Subscribing Info File window appears, listing the contents of the subscribing information file.

Clearing Information Files

To clear the contents of the subscribing information file:

1. Choose **Configure** from the Subscribing menu. The Subscribing Config dialog box appears.
2. Under **Info File**, click **Clear**. The contents of the subscribing information file are cleared.



Managing Hot Standby Operations

You can configure only one subscriber to be the hot standby system for a publisher. The hot standby system must be a different computer system from the publishing system it backs up.

A hot standby subscriber can take over for a publisher that suffers a fault or failure rendering it inoperable. When the publisher fails, you can turn on fail-over mode on the hot standby subscriber. Fail-over mode converts all replicated files from being read-only files to read/write-enabled files.

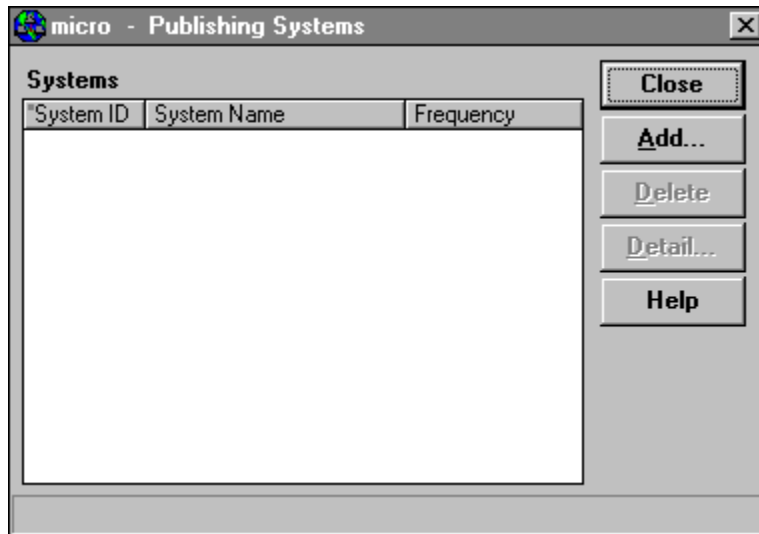
***Note:** If you want users to log in to the hot standby subscriber and run their applications as they run them on the publisher, you must use the same file and account names on the hot standby subscriber as the ones the publisher uses.*

Configuring a Hot Standby Subscriber

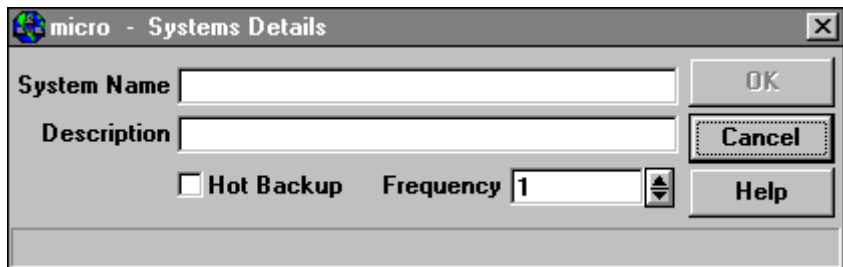
You configure a hot standby subscriber in two steps.

1. You add the name of the publishing system you want to hot-backup to the subscriber's Publishing Systems list, selecting the **Hot Standby** check box on the Publishing Systems dialog box.

- Choose **Systems** from the Subscribing menu. The Publishing Systems dialog box appears.



- Click **Add...** . The Systems Details dialog box appears.



- Enter the name of the publishing system in the **System Name** field.
- (Optional) Enter a description of the system in the **Description** field.
- Select **Hot Standby**.
- Click **OK**. The publishing system is added to the Systems list.

2. You subscribe to one or more published files, selecting the **Hot Standby** check box on the Subscribe dialog box:

The screenshot shows a Windows-style dialog box titled "garcia - Subscribe". It is divided into two main sections: "Subscribing File" and "Published File".

Subscribing File Section:

- Filename:** CUSTOMER
- Dictname:** D_CUSTOMER
- Account:** HS.SALES
- Desc:** (empty field)
- File Type:** File
- Checkboxes:** ☒ Subscribe, ☐ Specify..., ☒ Hot Standby

Published File Section:

- System Name:** strider
- Filename:** CUSTOMER
- Dictname:** (empty field)
- Account:** HS.SALES
- Desc:** (empty field)

Buttons: On the right side, there are three buttons: "Subscribe", "Cancel", and "Help".

Turning On Fail-Over Mode

If your primary publisher fails, you manually enable fail-over mode on the hot standby subscriber. To do this:

1. Choose **Fail Over** from the Subscribing menu. The Fail Over dialog box appears.
2. Choose the name of the publishing system for which to activate fail-over mode.
3. Click **OK**. The files on the subscriber are now read/write-enabled.

Reconciling the Hot Standby with the Publisher

When the primary publisher is available again, you must reconcile the contents of the subscribing files on the hot standby subscriber with the published files on the original publishing system.



Note: When reconciling subscribing files with published files, you need at least the same amount of space as the largest single file you are reconciling. You may need even more space than that, perhaps as much as twice the size of the largest file.

Here is an outline of the reconciliation process:

1. Stop publishing on the original publisher.
2. Prepare the hot standby subscriber to be reconciled with the publisher.
3. Run the **Sync** command on the original publisher.
4. Reset configurations on the original publisher.



Note: You can reconcile only those published files for which a hot standby subscriber system is configured. You cannot reconcile any files published from the failed publishing system that were only replicated but not hot-backed-up.

Stopping Publishing on the Original Publisher

1. Choose **Stop Publishing** from the Publishing menu.
2. Set the value of the UDRMODE configurable parameter to 0. For the procedure, see [Changing Configurable Parameter Values](#).
3. To unload shared memory, shut down UniVerse. For the procedure, see [System Startup and Shutdown](#).
4. Start UniVerse again. For the procedure, see [System Startup and Shutdown](#).

Preparing the Hot Standby Subscriber to Be Reconciled

1. On the hot standby subscriber, make sure there are no users logged on to the system.
2. Wait for all file updates to complete.
3. Choose **Stop Subscribing** from the Subscribing menu.

Running the Sync Command on the Original Publisher

1. On the original publisher, choose **Sync...** from the Publishing menu. The Sync dialog box appears, listing the hot standby system.
2. Click **Sync Up...** to start the reconciliation process. The original publisher connects to the hot standby subscriber and compares each record of each hot-backed-up file with the originally published version, and implements the updates in the original published files.

When the reconciliation process is complete, the subscribing files on the hot standby system are made read-only, and all subscriptions to the original publisher are restored.

Resetting Configurations on the Original Publisher

To return the original publisher to full publishing status:

1. Set the value of the UDRMODE configurable parameter to 1. For the procedure, see [Changing Configurable Parameter Values](#).
2. To unload shared memory, shut down UniVerse. For the procedure, see [System Startup and Shutdown](#).
3. Start UniVerse again. For the procedure, see [System Startup and Shutdown](#).

Some Restrictions

You cannot use the following UniVerse commands or statements on published or subscribing files:

- ALTER.TABLE
- CLEAR.FILE
- CNAME
- DELETE.FILE
- DROP.TABLE
- RESIZE

You cannot use triggers with published or subscribing files.

What to Do When Disk Space Fills Up

If replication log files on a publishing system are not regularly cleared, they can fill up all available disk space. When this happens, the state of replication on the publisher changes from Enabled to Full, and all published files become inaccessible.

One reason your log files might fill up is that one or more subscribing systems are not receiving replicated data for some reason. The publisher's log files are cleared only when all subscribing systems have received all replicated data. To resume replication activity in such cases, you need to fix the problem on the subscribing system so that it is receiving replicated data again. This clears the publisher's log files, thus freeing up disk space. When there is enough space to continue, you can resume replication.

To resume replication activity:

1. Do one or both of the following:
 - Remove unneeded files from the publishing system's disk to free up disk space.
 - Fix the problem on the subscribing systems so they can continue to receive replicated data.
2. On the publisher, choose **Resume Publishing** from the Publishing menu.
3. On the subscriber, choose **Start Subscribing** from the Subscribing menu.

Removing Obsolete Replication Log Files

When you start up UniVerse on a machine where replication is enabled, normally all existing replication log files are removed. Replication log files are automatically removed only under the following conditions:

- The replication log file is not being used
- All replication log file entries have been sent to all subscribers
- The replication log file is not marked as “crashed”

In some cases, some replication log files may remain on the system. For example, a subscriber may not receive all its updates because it is off-line.

You can remove obsolete replication log files manually.

What to Do When Replication Fails

When either a publishing or a subscribing system fails for any reason, you need to ensure that the replicated data on all subscribing systems is identical with the published data on the publishing system before you restart replication. In the event of any system crash, do the following:

1. Fix the problem on the systems that crashed.
2. Restart UniVerse. For the procedure, see [System Startup and Shutdown](#).
3. Ensure that published data is identical with replicated data on all subscribers.
4. (Optional) Do one or both of the following:
 - On a publishing system, restart the publishing system if it is not configured to start automatically. Choose **Start Publishing** from the Publishing menu.
 - On a subscribing system, restart the subscribing system if it is not configured to start automatically. Choose **Start Subscribing** from the Subscribing menu.



***Note:** If you are fixing a problem on a subscriber only, you need not stop the publishing system on the publisher. Once the subscriber resumes operation, all of the publisher's replicated data is sent to it.*

Monitoring System Activity

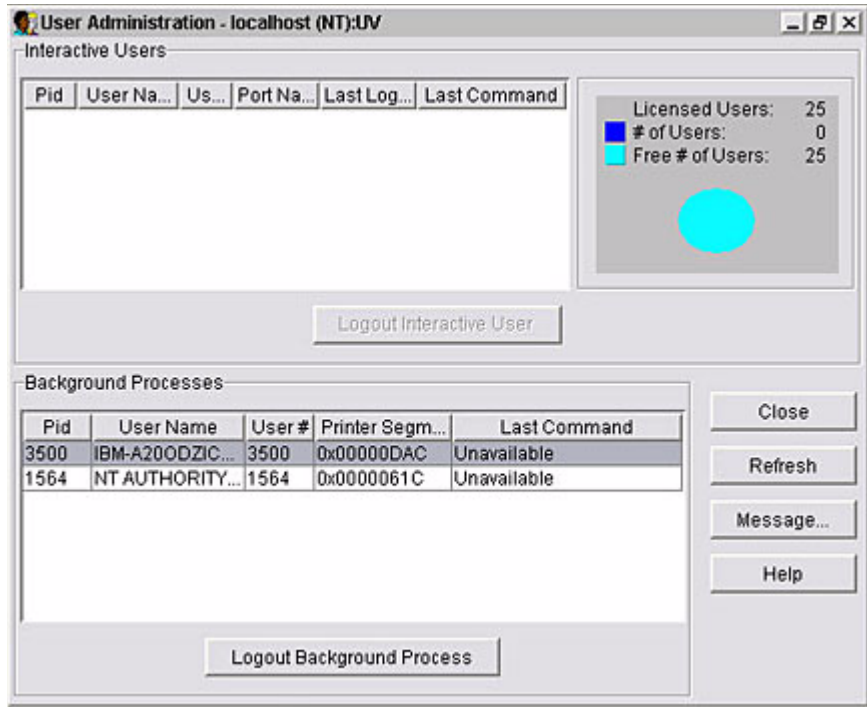
Listing Active UniVerse Processes and Jobs	14-3
Listing UniVerse Jobs with PORT.STATUS	14-6
Terminating a Process	14-7
Examining Shared Memory	14-8
Semaphore Table	14-9
File Lock Table	14-10
Group Lock Table	14-11
Update Record Lock Table	14-11
User Process Control Locks	14-11
Dynamic File Table	14-12
UniVerse Configuration	14-12
General System Information	14-12
Catalog Shared Memory	14-13
Printer Memory Segment	14-13
Examining Disk Usage on UNIX Systems	14-15
Monitoring Disk Usage on UNIX Systems	14-15
System Files that Grow	14-17
Monitoring Response Time on UNIX Systems	14-18
Keeping Directory Files Small	14-18
Running Programs During Off-Hours	14-19
Monitoring Errors on UNIX Systems	14-20

One of the more important jobs of the system administrator is to monitor activity on the system (such as disk use and CPU use) and to deal with bottlenecks and other potential problems *before* they impact users. This section describes ways to find out who is doing what in UniVerse and at the operating system level. This chapter describes the following:

- How to get information about all current UniVerse processes and jobs
- How to terminate user and phantom processes
- How to examine the use of shared memory
- How to monitor disk usage
- How to monitor CPU response time
- How to monitor error reporting

Listing Active UniVerse Processes and Jobs

To view UniVerse processes, choose **Users** from the UniAdmin main menu. The **User Administration** window appears, as shown in the following example:



This window contains a snapshot of the user and background processes at the time the window was invoked. To view the current user and background processes, click **Refresh**.

From this window you can also:

- Send a message to users (see [Sending Messages with UniAdmin](#) in Chapter 17, “[Sending Messages to Users](#)”).
- Terminate a process (see “[Terminating a Process](#)” on page 7).

The **User Administration** window is divided into two main areas:

- Interactive Users

■ Background Processes

Interactive Users

The following information appears for each user process:

Parameter	Description
Pid	The process ID. On a Windows, system this is the UniVerse user number. It is the same as the value shown in the User # column.
User Name	The user's login name.
User #	The user's UniVerse user number.
Port Name	On a UNIX system this is the device path of the session. On a Windows system, this is the user type and UniVerse user number, for example, <code>console:124</code> .
Last Logged In	The date and time the user logged on.
Last Command	The last command the user issued (if known).

User Process Information

Background Processes

The following information appears for each background process:

Parameter	Description
Pid	The process ID. On a Windows system this is the UniVerse user number. It is the same as the value shown in the User # column.
User Name	The user's login name.
User #	The user's UniVerse user number. This column is displayed for Windows systems only.
Printer Segment	The address of the printer shared memory segment.
Last Command	The last command issued (if known).

Background Process Information

This window also has the following options:

- **Close.** Exits the window.
- **Refresh.** Displays the current user and background processes.
- **Message.** Displays the **Send Message** dialog box.
- **Help.** Invokes the Help system.
- **Logout Interactive User.** Logs out a selected user.
- **Logout Background Process.** Logs out a selected background process.

Listing UniVerse Jobs with PORT.STATUS

The UniVerse PORT.STATUS command is a diagnostic tool that lists currently active UniVerse jobs on the system. The syntax is as follows:

```
PORT.STATUS [ USER name ] [ PORT number ] [ DEVICE pathname ]
[ PID process# ] [ FILEMAP ] [ LAYER.STACK ] [ MFILE.HIST ]
[ LOCK.HIST ] [ { ENABLE | DISABLE } LOCK.HIST ] [
ODBC.CONNECTIONS ] [ LPTR ]
```

The PORT.STATUS command with no options produces a report that looks like this:

```
>PORT.STATUS
```

```
There are currently 5 UniVerse sessions; 5 interactive, 0 phantom
```

Pid..	User name.	Who	Port name.....	Last command processed.....
6002	walton	13	/dev/ttyp2	PORT.STATUS
5047	cad	16	/dev/ttyp8	LIST TECH.CONTACTS <<I2,Enter vendor id>>
5812	tamman	26	/dev/ttypd	SELECT IB WITH DESC LIKE '...LOCATE...'
5844	tamman	47	/dev/ttyq8	LIST IB LONG.DESC FIXED.AT.ANY
5822	tamman	50	/dev/ttyq7	LIST IB LONG.DESC FIXED.AT.ANY

The Who column lists the port number, and the Port name column lists the device path of the session. For complete details about PORT.STATUS, see the *UniVerse User Reference*.

Terminating a Process

You can terminate a user or background process using the UniVerse User Administration window.

To terminate a user process:

1. Choose **Users** from the UniVerse Admin Control Panel. The UniVerse User Administration window appears.
2. Choose the user from the Interactive Users list.
3. Click **Logout Interactive User**. A message box appears.
4. Click **Yes**. An attempt is made to log the user off the server. The UniVerse User Administration window is updated.

To terminate a background process:

1. Choose **Users** from the UniVerse Admin Control Panel. The UniVerse User Administration window appears.
2. Choose the process from the Background Processes list.
3. Click **Logout Background Process**. A message box appears.
4. Click **Yes**. The chosen process is immediately terminated and the UniVerse User Administration window is updated.

Examining Shared Memory

The shared memory analysis utility lets you examine information in the disk and the printer shared memory segments so you can diagnose problems if the system hangs.

The disk shared memory segment is a global work area that must be present for UniVerse to work. The printer shared memory segment is local to each user. It stores user environment information, such as TERM settings and printer channel attributes, that must be available to all processes a user creates.

You use the **analyze.shm** command from a UNIX shell or an MS-DOS window, or you use the ANALYZE.SHM command in UniVerse. The basic syntax of the command is as follows:

analyze.shm { *options* }

You must use at least one option on the command line. Specify options with a minus sign (–) followed by the letter of the option. You can combine options as in the operating system command *analyze.shm –sfgr*. Note that the UniVerse commands require lowercase options. *options* are any of the following:

Option	Description
–a[<i>seg #</i>]	All information
–b	Catalog shared memory
–c	UniVerse configuration
–d	Dynamic file table
–f	File lock table
–g	Group lock table
–l	Logging system information
–L	NLS locale information
–M	NLS map information
–n	Raw numbers and empty table entries (slots that are not in use)
–p[<i>seg #</i>]	Printer memory segment

analyze.shm Options

Option	Description
-r	Update record lock table
-R	Replication information
-s	Semaphore table
-t [0]	UniVerse configurable parameter values
-u	User process control locks
-x	General system information
-z	Number of nodelocked and network licenses. This option must be used with -c.

analyze.shm Options (Continued)

The following sections describe the options.

Semaphore Table

The -s option displays the information sorted by semaphore type. Alternately, you can use the SEMAPHORE.STATUS command in the UV account. The semaphore types are as follows:

- File lock semaphores
- Group/update record lock semaphores
- Login semaphore
- Port status semaphore
- Dynamic file semaphore
- Transaction logging semaphore

These semaphores store information related to file concurrency control, login count, current active UniVerse jobs, current split load and related statistics for dynamic files, and changes made to files for system backups. A sample report looks like the following:

```
$ analyze.shm -s
File access      State Netnode Owner Collisions Retrys
Semaphore #      1      0      0      0          0      0
.
.
.
Semaphore #     23      0      0      0          0      0
Group access      State Netnode Owner Collisions Retrys
Semaphore #      1      0      0      0          0      0
Semaphore #      2      0      0      0          0      0
.
.
.
Semaphore #     22      0      0      0          0      0
Semaphore #     23      0      0      0          0      0
Login              State Netnode Owner Collisions Retrys
Semaphore #      1      0      0      0          0      0
Port status        State Netnode Owner Collisions Retrys
Semaphore #      1      0      0      0          0      0
Type 30 file       State Netnode Owner Collisions Retrys
Semaphore #      1      0      0      0          0      0
Transaction log    State Netnode Owner Collisions Retrys
Semaphore #      1      0      0      0          0      0
```

File Lock Table

If you use the `-f` option when there are active locks or semaphores, you see the following fields: i-node (Inode), device (Device ID), lock type (LTYPE), and signature (Signature).

If there are no active locks or semaphores, you see the following message:

```
$ analyze.shm -f
No locks or semaphores active.
```


Group Lock Table

When there are active group locks, the following fields are displayed: i-node (Inode), device (Device ID), group address (GRPAD), signature (Signature) and lock word (LWORD). Here is an example:

```
$ analyze.shm -g
Group Locks:
Slot #   Inode   Device ID      GRPAD   Signature      LWORD
    6     44511         116 0x00000000 0x0000004E 0x00000001
```

Update Record Lock Table

If there are active update record locks, the fields displayed are i-node (Inode), device (Device), group address (G-Address), and key (Item-ID). From UniVerse, the LIST.READU EVERY command produces similar output.

```
$ analyze.shm -r
Active Group locks:
Device  Inode. Netnode Userno  Lmode  G-Address.  Readus
    5   42320         0     26   8  IN          6B000      1
    5   24572         0     26  21  IN           400      1

Active Record Locks:
Device  Inode. Netnode Userno  Lmode  Pid  LoginId  Item-ID.....
    5   42320         0     26     8  RU          2437
    5   24572         0     26     8  RU          &NEXT.AVAILABLE&
```

Adding the `-n` option displays information for each slot in the table, including unused slots that are available.

User Process Control Locks

The `u` option displays all the user-controlled locks. You see either the state of the lock or the number of the user holding the lock. LIST.LOCKS from UniVerse produces an identical report.

```
$ analyze.shm -u

0:----- 1:----- 2:----- 3:----- 4:----- 5: 66      6:-----
7:-----
8:----- 9:----- 10:----- 11:----- 12:----- 13:----- 14:-----
15:-----
16:----- 17:----- 18:----- 19:----- 20:----- 21:----- 22:-----
23:-----
```

```

24:----- 25:----- 26:----- 27:----- 28:----- 29:----- 30:-----
31:-----
32:----- 33:----- 34:----- 35:----- 36:----- 37:----- 38:-----
39:-----
40:----- 41:----- 42:----- 43:----- 44:----- 45:----- 46:-----
47:-----

```

Dynamic File Table

The `d` option displays the active dynamic (type 30) file control blocks:

```

$ analyze.shm -d

Dynamic Files:
Slot # Inode Device Ref Count Splitload Mergeload Curmod
Basemod Largerec Filespl Nextsplit
    0 42320      5      1      19      80      50
   357      256 590748     102
    1 26964     116      2      19      80      50
   150     128 247476     23

```

Again, use the `-n` option to see unused slots in the table.

UniVerse Configuration

To see the UniVerse configurable parameters for the current UniVerse directory, use the `-c` option. The output is similar to CONFIG ALL.

Note that you also see the installed packages. Configurable parameters that have been changed from the default settings are flagged with an asterisk (`*`). Numeric fields are in decimal format (WIDE0 is in hexadecimal format).

General System Information

To see the remaining fields in the disk shared memory segment, use the `x` option:

```

$ analyze.shm -x

General System Information:
Login count: 27
Base address for printer segment: 0xBF800000
Universe home directory: /usr/ibm/uv/
Shared Catalog: 0
Active Logging: 0
Active log roll forward: 0
Logging I/Os allowed: 0
log_set: 0

```

```

Next available transaction ID: 1
log_volno: 0
log_shutdown: 0
Spare5: 0
.
.
.
Spare14: 0
Semaphore debugging: 0
Uvnetd debugging: 0
Feature3: 0
Feature4: 0
.
.
.
Feature15: 0
Feature16: 0

```

Catalog Shared Memory

To see the contents of catalog shared memory, use the *b* option:

```

$ analyze.shm -b

State of shared memory: 4
Number of programs loaded into shared memory: 3
Size      References  Users   Pathname
933       0             0       /usr/ibm/uv/catdir/*gtar*GTAR.858/7.1
895       0             0       /usr/ibm/uv/catdir/*gtar*GTAR.858/7.2
1055      0             0       /ul/gtar/PGMS.O/GTAR.9663.2

```

The *b* option is useful when you want to see what programs are in catalog shared memory, which programs are in use, and which are available. For each program loaded into shared memory, you see the size, references, user count, and path.

Printer Memory Segment

You must be a UniVerse Administrator to examine all the printer segments. Otherwise, you can examine only the printer memory segments that you own.

The *-p* option displays a user's printer memory segment followed by the segment's ID in decimal (the hexadecimal format indicates the key of the segment).

If you specify the segment ID after the `-p` option, the ID is the shared memory key of that segment. If you do not specify the segment ID, the segment of the shared memory analysis tool is used and you see the default printer settings.

The `-p` option displays a memory map showing used and unused sections for the user's printer shared memory segment 609. The terminal driver (BDS or SYSV) is included.

Examining Disk Usage on UNIX Systems

The amount of information users can store on the disk is limited by two factors. First is the physical capacity of the disk. Second is the allocation of i-nodes, the data structures used for keeping track of the location of files.

It is easy for users to fill even a very large disk quickly. Free disk blocks and free i-nodes can be a problem. If the free i-node count falls below 100, the system spends most of its time rebuilding the free i-node array. If a file system runs out of space, the system prints `no space` messages and does little else. To avoid problems, keep at least 15% of the file system free to allow for the creation of temporary files and other daily changes.

The amount of necessary free space can vary greatly depending on your installation, so take the time to learn the disk usage patterns on your system and be aware if the free space suddenly shrinks dramatically.

Also, a number of administrative files (such as the system accounting files in */usr/adm*) may need to be periodically compacted. In addition, file system unload and reload can sometimes make small improvements in problems with disk space.

Monitoring Disk Usage on UNIX Systems

Use the `df(1)` command regularly to list the amount of free disk space available. For example:

```
% df
Filesystem      kbytes    used   avail capacity  Mounted on
/dev/ioc/cdisk00a  19102   12932   4258    75%      /
/dev/ioc/cdisk00c  47950   41056   2098    95%     /usr
/dev/ioc/cdisk00g 289094  244550  15634    94%     /cs
/dev/ioc/cdisk01a  19158   13436   3806    78%     /mktg
/dev/ioc/cdisk01c  48070   42060   1202    97%     /qa
/dev/ioc/cdisk01g 289094  244694  15490    94%     /rd
```

You can display the same information with the UniVerse AVAIL command. In addition, execute the *du*(1) command daily during off-peak hours. Keep the output for later comparison. This lets you spot users who are rapidly increasing their disk usage. The following example shows partial output of the *du* command:

```
% du
4      ./lost+found
4      ./usr/lost+found
182    ./usr/adm
2390   ./usr/bin
1      ./usr/crash
29     ./usr/dict/papers
402    ./usr/dict
1081   ./usr/etc
13     ./usr/hosts
14     ./usr/include/net
.
.
.
461    ./ul/zeke
21895  ./ul
1      ./tmp
```

If you are running out of disk space, do one of the following:

- Add an additional disk or use a larger disk.
- Remove unnecessary files from the disk and reorganize those that remain.

The first step is to ask all users to delete any unnecessary files. Files not needed online can be backed up to tape and restored when you need them. Use the *find*(1) command to locate inactive (or large) files. The following command sends mail to *root*, listing the names of files neither written nor accessed in the last 90 days:

```
% find / -mtime +90 -atime +90 -print | mail root&
```

This command finds all user files if you are logged in as *root*.

System Files that Grow

Certain system files automatically grow if they are not watched and periodically compacted. Some examples follow:

Accounting File	Description
<i>/etc/wtmp</i>	Login accounting information.
<i>/usr/adm/acct</i>	Process accounting. This file gets big quickly. Remove it to disable accounting.
<i>/usr/adm/cronlog</i>	Status log of commands executed by <i>cron</i> (1). This log file is optional. If you create it, <i>cron</i> logs its activity in it.
<i>/usr/spool</i>	Spooling directory for line printers, <i>uucp</i> (1C), etc., and whose subdirectories should be compacted as described below. Note especially: <i>/usr/spool/uucp/LOGFILE</i> <i>/usr/spool/uucp/SYSLOG</i> <i>/usr/spool/uucpublic/*</i> <i>/usr/spool/uv/act.log</i> <i>/usr/spool/uv/err.log</i>
<i>/usr/adm/messages</i>	Error logging file.
<i>/tmp</i>	Temporary files of all kinds.

Systems File That Grow

You can delete accounting files after the programs that generate summary data have been run. Other system log files can be printed out or backed up to tape before deletion.

Monitoring Response Time on UNIX Systems

After disk space, the biggest problem you may have is overuse of CPU time by certain system processes. Some processes require an inordinate amount of processing. In a multiuser system, you may need to require that certain processing be done during off-hours. The system can initiate processes automatically at a specified time, as described later in this chapter. System response time can also suffer as a result of disk fragmentation or poor organization of a file system.

When the system is slow, find out why. The *who*(1) command lists the people logged on to they system. The *ps*(1) command shows what they are doing.

From UniVerse, you can print a list of the current users with the LISTU command. The STATUS command displays information about which disks are currently active, what phantom tasks are running, and which users are logged on to they system.

The following pointers can help improve response time on your system:

- Keep directory files small
- Run programs during off-hours

Keeping Directory Files Small

Directories larger than 5K bytes (320 entries) are very inefficient because of file system indirection. For example, a large */usr/mail*(1) or */usr/spool/uucp*(1) directory can really slow the system down. The following command finds large directories:

```
% find / -type d -size +10 -print | more
```

Removing files from directories does not make the directories smaller, but you can compact directories. The following example shows the sequence of commands to compact the directory */usr/mail*(1):

```
% mv /usr/mail /usr/omail
% mkdir /usr/mail
% chmod 777 /usr/mail
% cd /usr/omail
% find . -print | cpio -pdlm
% cd ..
% rm -rf omail
```


Running Programs During Off-Hours

The *cron*(8) program can be used to run programs during off-hours. This is especially important for time-consuming programs whose output is not immediately needed, such as the following:

- Accounting
- File system administration
- Long-running user-written shell procedures

Monitoring Errors on UNIX Systems

UniVerse logs user and UniVerse system errors in a central text file, *errlog*, located in the UV account directory. All errors that pass through the internal functions *warning()*, *mwarning()*, *fatal()*, or *mfatal()* are logged. The severity of logged errors varies considerably, from mistyped commands to serious system errors.

Create the *errlog* file using commands such as the following:

```
$ touch /usr/ibm/uv/errlog
$ chmod 666 /usr/ibm/uv/errlog
```

By default, the *errlog* file has room for 100 logged entries. The first line of the file is not an error entry: it specifies the number of the next entry to be written. After the 100th error is logged, the 101st error overwrites the first error entry.

Here is a portion of a sample *errlog* file:

```
015    REC1
Mon Nov 16 10:23:13  19 root Illegal verb "SUN.SPORT".
Mon Nov 16 10:29:17  19 root Verb "D" is not in your VOC.
Mon Nov 16 15:12:03  47 paul Program "BSHIPtest":
Mon Nov 16 15:12:50  69 tim Program "T2":
```

Each logged error contains the following information:

- System date and time when the error was logged
- Port number of the user who caused or encountered the error
- Login name of the user who caused or encountered the error
- Text of the error message (up to 256 characters)

Error messages such as `A fatal error has occurred in UniVerse` or `Abnormal termination of UniVerse` indicate a problem in the UniVerse system itself. If you get a message like one of these, write down the exact text of the message, note what process caused the message, and call IBM Customer Support with the problem.

UniVerse File Utilities

Administering UniVerse Files	15-4
Listing Files in a UniVerse Account	15-5
View File Properties	15-6
View File Statistics.	15-11
Running File Diagnostics.	15-13
Repairing Damaged Files.	15-17
The Format Conversion Utility	15-21
Converting the Format of Data Files and UniVerse BASIC Code . . .	15-21
The uvfixfile Utility	15-28
Verifying File Integrity	15-28
Fixing a Corrupt Hashed File	15-29
Examining File Statistics	15-30
Using Interactive Mode	15-30

Beta Beta

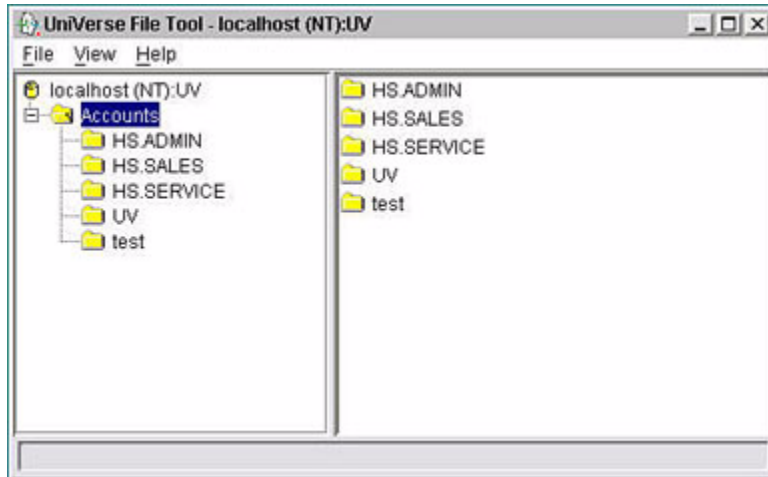
There are a number of utilities you can use to keep your files at peak efficiency. This section describes three of these utilities:

- File Tool option on the UniAdmin Control Panel
- Format conversion utility (FORMAT.CONV)
- *uvfixfile* utility

Use the File Tool option of UniAdmin for general file administration. Use the format conversion utility to import files and BASIC object code from different hardware platforms. Use the *uvfixfile* utility to repair broken static hashed files. For information about other UniVerse file maintenance commands and techniques, see *UniVerse System Description*.

Administering UniVerse Files

To administer UniVerse files, choose **File Tool** from the UniAdmin menu. The **File Tool** window appears, as shown in the following example:

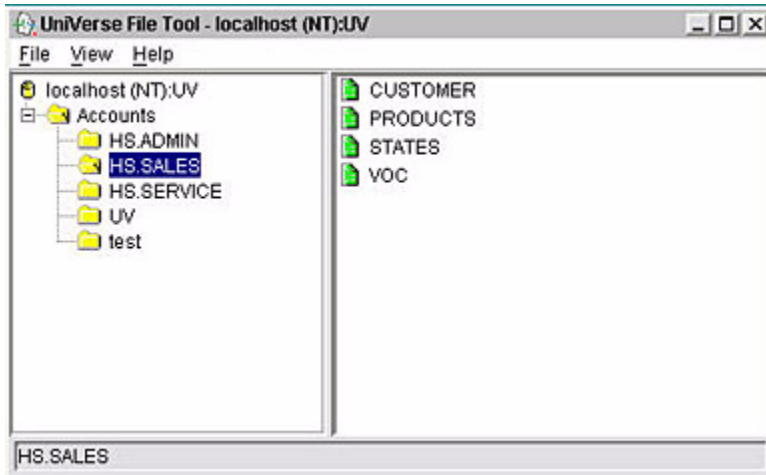


The tasks you can perform from this window include:

- Listing all files in all UniVerse accounts
- Listing file properties and statistics
- Running file diagnostics
- Repairing damaged files

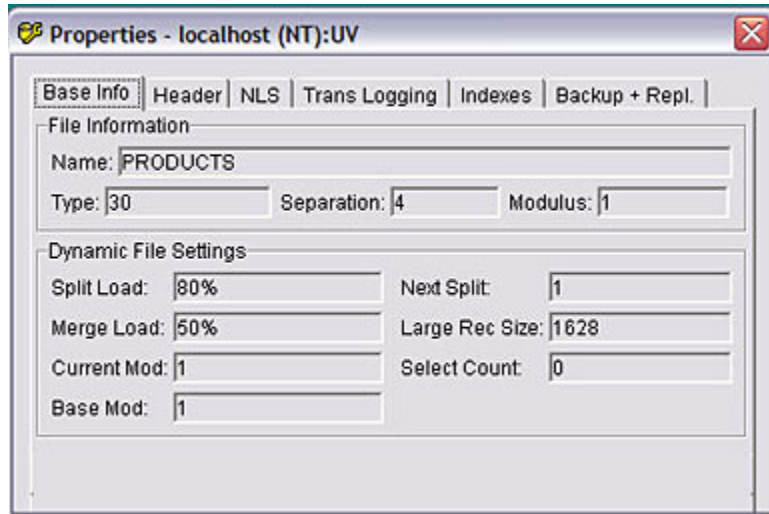
Listing Files in a UniVerse Account

From the **UniVerse File Tool** window, select the account for which you want to view files. All files for that account appear in the right pane of the window, as shown in the following example:



View File Properties

To view the properties of a file, select the file for which you want to view properties in the right pane of the **UniVerse File Tool** dialog box, then from the **File** menu, click **Properties**. The **Properties** dialog box appears, as shown in the following example:



Base Information

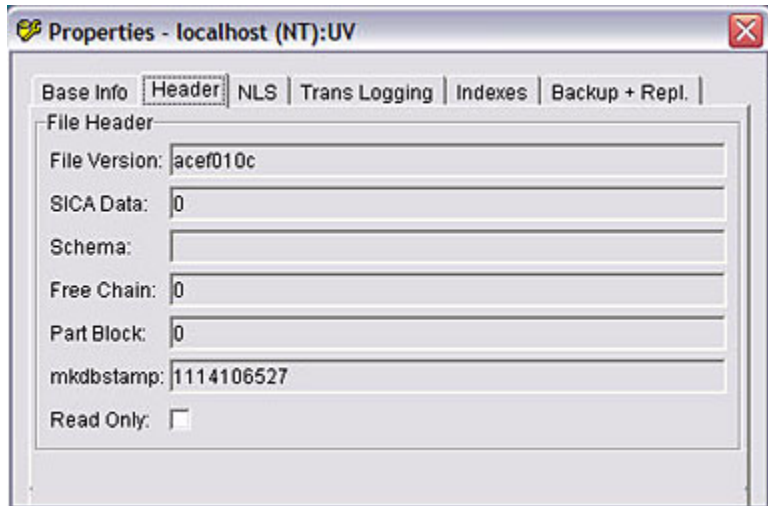
From the **Properties** dialog box, select the **Base Info** tab. UniAdmin displays the following information about the file:

- File name
- File type
- Separation and modulus of static hashed file
- Dynamic file parameters

For a detailed description of UniVerse files, see the *UniVerse System Description*.

Header Information

From the **Properties** dialog box, select the **Header Info** tab. A dialog box similar to the following example appears:



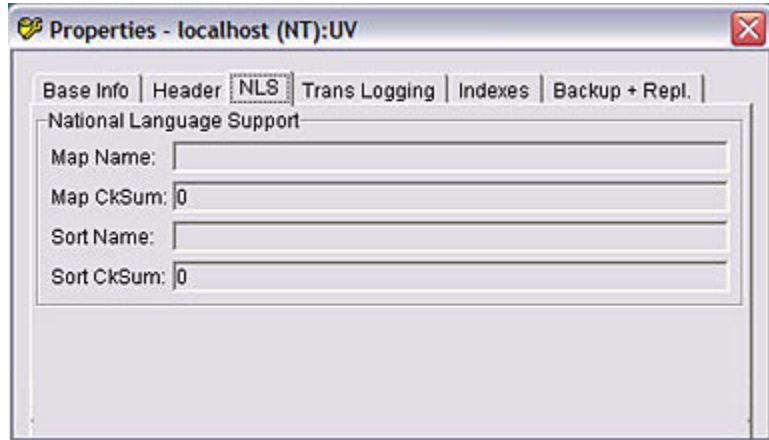
UniAdmin displays the following header information for the file:

- File version
- SICA and schema name, if the file is a table
- Free chain
- Part block
- mkdbstamp

If the **Read Only** check box is selected, the file is read-only.

National Language Support (NLS) Information

From the **Properties** dialog box, click the **NLS** tab. A dialog box similar to the following example appears:



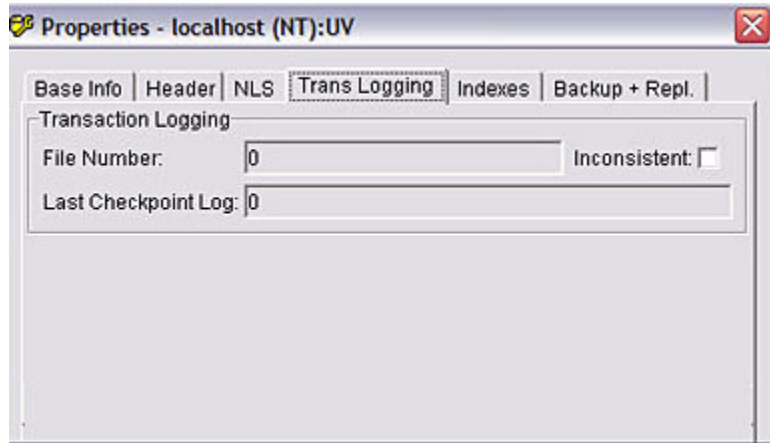
UniAdmin displays the following information about NLS:

- In the **Map Name** box, the name of the character set map associated with the file
- Map checksum
- In the **Sort Name** box, the Collate convention that determines how to sort file data
- Sort checksum

For detailed information about NLS, see the *NLS Guide*.

Transaction Logging Information

From the **Properties** dialog box, click the **Trans Logging** tab. A dialog box similar to the following example appears:



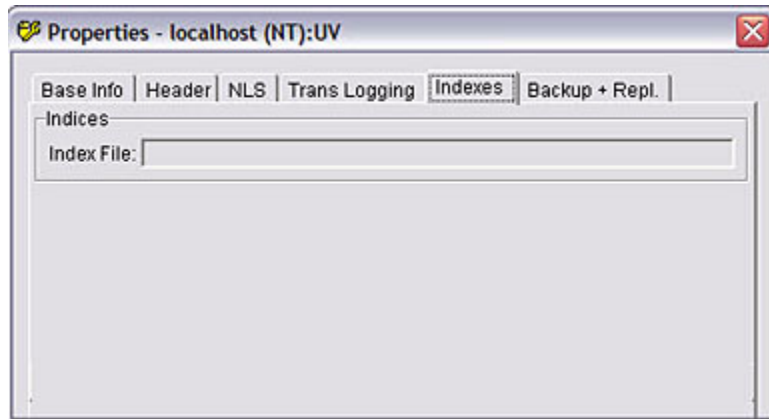
UniAdmin displays the following information about transaction logging:

- File number
- Number of the last checkpoint log

If the **Inconsistent** check box is selected, and file is inconsistent.

Indexes Information

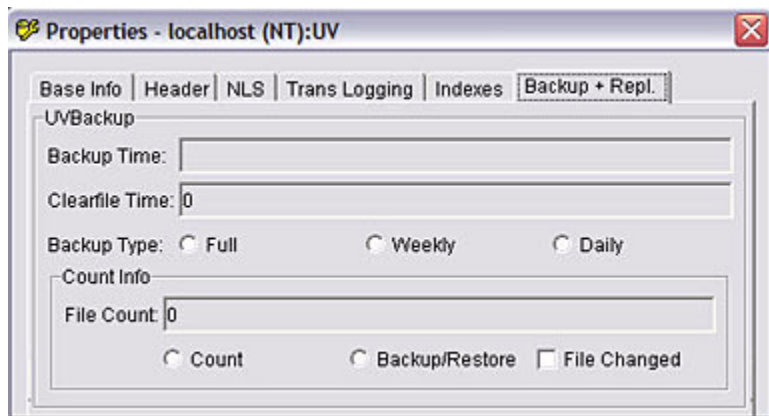
From the **Properties** dialog box, click the **Indexes** tab. A dialog box similar to the following example appears:



If the file has a secondary index, UniAdmin displays the name of the index in the **Index File** box.

Backup and Replication Information

From the **Properties** dialog box, click the **Backup + Repl.** tab. A dialog box similar to the following example appears:



UniAdmin displays the following information about backup and replication:

- In the **Backup Time** box, the date and time of the last backup.
- In the **Clearfile Time** box, the date and time the last CLEARFILE command was executed against the file.
- The type of backup, either full, weekly, or daily.
- In the **File Count** box, the number of records in the file, counted by either the last COUNT command executed against the file, the last full backup, or the last restore.

If the **File Changed** check box is selected, the file count may be out of date because the file has been changed since the last file count.

- In the **Replication** area, **Stat** indicates whether the file is a published file, a subscription file, or a failed-over file. This area also lists the replication ID in the **ID** box.

View File Statistics

To view statistics about a file, from the **UniVerse File Tool** window, select the file for which you want to view statistics, then click **File**, then click **Statistics**. A dialog box similar to the following example appears:

File Information	
Name:	PRODUCTS
Date:	
Type:	30
Separation:	4
Modulus:	1

Statistics	
Reads:	0
ReadUs:	0
Writes:	0
Write Updates:	0
Oversize Reads:	0
Oversize Writes:	0
Overflow Reads:	0
Deletes:	0
Selects:	0
ReadLs:	0
Opens:	0
Cleadfiles:	0
Write to locked:	0
Writes blocked:	0
ReadU lock conflict:	0
ReadL conflicts:	0
Compression:	0

File Information

In the **File Information** area of the **Statistics** dialog box, UniAdmin displays the following information:

- File Name
- Date of the last update
- File separation if a static hashed file
- Modulus if a static hashed file

File Statistics

UniAdmin displays the following statistics about the file you selected:

Field	Description
Reads	Total number of READ operations on the file.
ReadUs	Total number of READU operations on the file.
Writes	Total number of WRITE operations on the file.
Write Updates	Total number of WRITEU operations on the file.
Oversize Reads	Total number of READ operations executed against large records.
Oversize Writes	Total number of WRITE operations executed against large records.
Overflow Reads	Total number of READ operations that accessed overflow buffers.
Deletes	Total number of DELETE operations on the file.
Selects	Total number of SELECT operations on the file.
ReadLs	Total number of READL operations on the file.
Opens	Total number of OPEN operations on the file.
Clearfiles	Total number of CLEARFILE operations on the file.

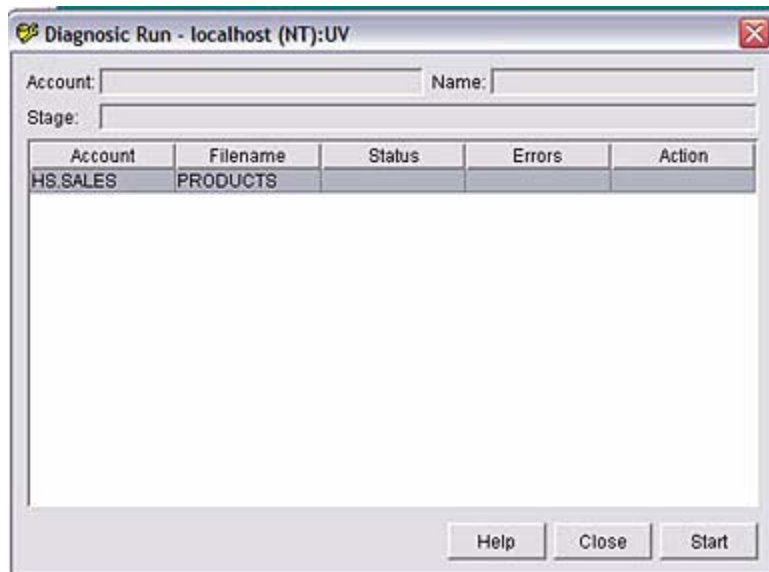
UniAdmin File Statistics

Field	Description
Write to Locked	Total number of WRITE operations executed against a locked record.
Writes Blocked	Total number of WRITE operations blocked by a record lock.
ReadU lock conflict	Total number of READU operations that failed because of an existing record lock.
ReadL conflicts	Total number READL operations that failed because of an existing record lock.
Compressions	Total number of free operations that compacted a group after a record was deleted.

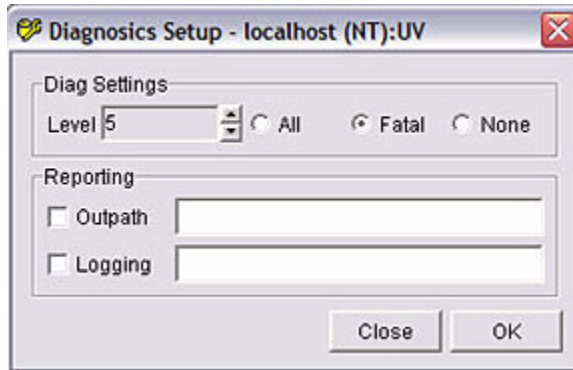
UniAdmin File Statistics (Continued)

Running File Diagnostics

To run file diagnostics against a file, from the **UniVerse File Tool** window, select the file for which you want to run diagnostics, then click **File**, then click **Diagnostics**. A dialog box similar to the following example appears:



To begin running diagnostics, click **Start**. The **Diagnostics Setup** dialog box appears, as shown in the following example:



Use the **Diagnostics Setup** dialog box to specify how much diagnostic testing to perform on the file, how much diagnostic detail to list in the error report window, and where to store output from the diagnostic test.

Determine Diagnostic Level

In the **Diag Settings** area, select the level of diagnostic detail you want to produce in the **Level** box. The lowest diagnostic level is 1, while the highest diagnostic level is 10. The default value is 5. The higher this level setting, the longer the diagnostics test takes to complete. Use the **Level** arrows to select the diagnostic level.

Specify Types of Errors Report

In the **Diag Settings** area, select the types of errors you want to appear in the **Error Report** window. Select one of the following values:

- **All** – List all diagnostic details in the **Error Report** window.
- **Fatal** – List only fatal errors in the **Error Report** window.
- **None** – Do not list any errors in the **Error Report** window.

Specify Output Location

If you want to save a report of irreparable groups and record blocks detected by the diagnostic test, select **Output**, then enter the full path to a directory where you want to store the output.

If you want to store a copy of the error report, select **Logging**, then enter the full path to a directory where you want to store the report.

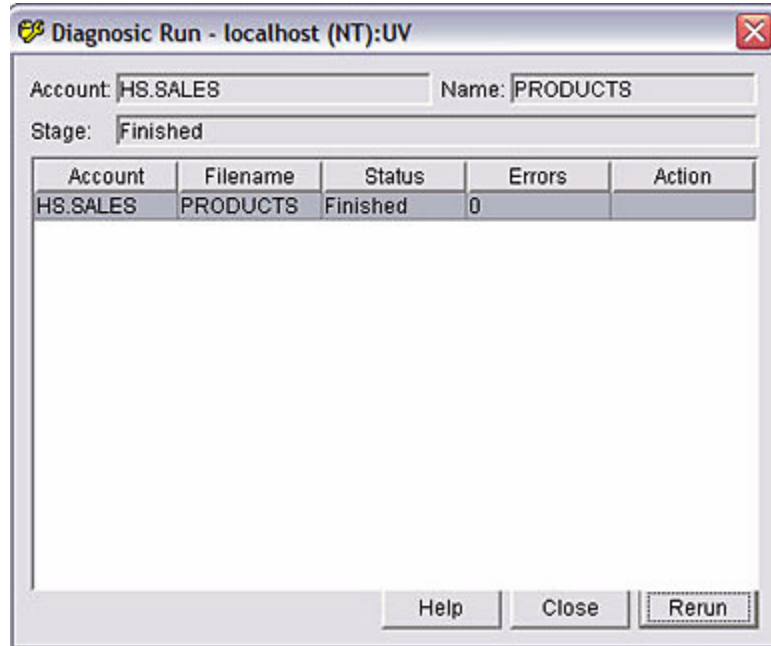
If you do not specify **Output** or **Logging**, the output and error report are stored in the directory where the file currently resides.

Click **OK** to save your changes, or click **Close** to exit without saving changes. The diagnostics program starts.

Diagnostics Test

After you run the diagnostics program, the **Diagnostic Run** dialog box displays the account name, the file name, and the progress of the program. It also displays the number of errors encountered and specifies what action to take if an error is detected.

The following example illustrates the **Diagnostic Run** dialog box:



Viewing Errors

If the diagnostics test program detects an error, click the account name to display the Error Report window.

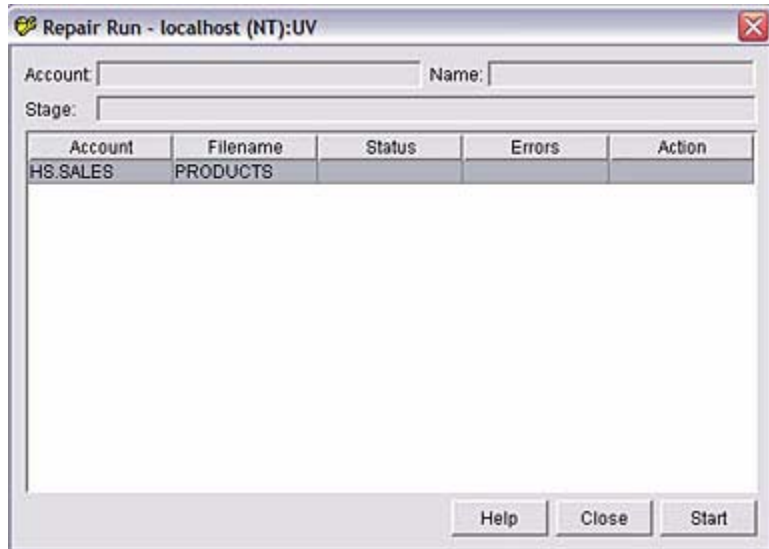
The **Error Report** windows displays the following information:

- The error type
- A description of the error
- The group number where the error occurred
- The record block number where the error occurred

After repairing the damaged file, click **Rerun** to rerun the diagnostic program.

Repairing Damaged Files

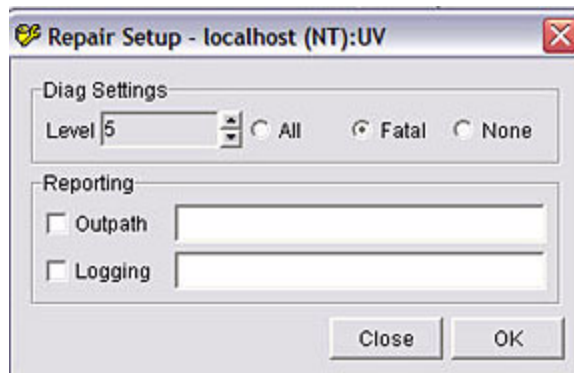
To run file diagnostics against a file, from the **UniVerse File Tool** window, select the file you want to repair, then click **File**, then click **Repair**. A dialog box similar to the following example appears:



The **Repair Run** dialog box is titled "Repair Run - localhost (NT):UV". It features a title bar with a yellow icon and a red close button. Below the title bar are three input fields: "Account:", "Name:", and "Stage:". Below these fields is a table with five columns: "Account", "Filename", "Status", "Errors", and "Action". The first row of the table contains the text "HS.SALES" under "Account" and "PRODUCTS" under "Filename". The remaining three columns in this row are empty. Below the table is a large empty rectangular area. At the bottom right of the dialog box are three buttons: "Help", "Close", and "Start".

Account	Filename	Status	Errors	Action
HS.SALES	PRODUCTS			

Click **Start**. The **Repair Setup** dialog box appears, as shown in the following example:



The **Repair Setup** dialog box is titled "Repair Setup - localhost (NT):UV". It features a title bar with a yellow icon and a red close button. Below the title bar is a section labeled "Diag Settings" containing a "Level" dropdown menu set to "5" and three radio buttons: "All", "Fatal" (which is selected), and "None". Below this is a section labeled "Reporting" containing two checkboxes: "Outpath" and "Logging", each followed by an empty text input field. At the bottom right of the dialog box are two buttons: "Close" and "OK".

Use the **Repair Setup** dialog box to specify how much diagnostic testing to perform on the file, how much diagnostic detail to list in the error report window, and where to store output from the diagnostic test.

Determine Diagnostic Level

In the **Diag Settings** area, select the level of diagnostic detail you want to produce in the **Level** box. The lowest diagnostic level is 1, while the highest diagnostic level is 10. The default value is 5. The higher this level setting, the longer the diagnostics test takes to complete. Use the **Level** arrows to select the diagnostic level.

Specify Types of Errors Report

In the **Diag Settings** area, select the types of errors you want to appear in the **Error Report** window. Select one of the following values:

- **All** – List all diagnostic details in the **Error Report** window.
- **Fatal** – List only fatal errors in the **Error Report** window.
- **None** – Do not list any errors in the **Error Report** window.

Specify Output Location

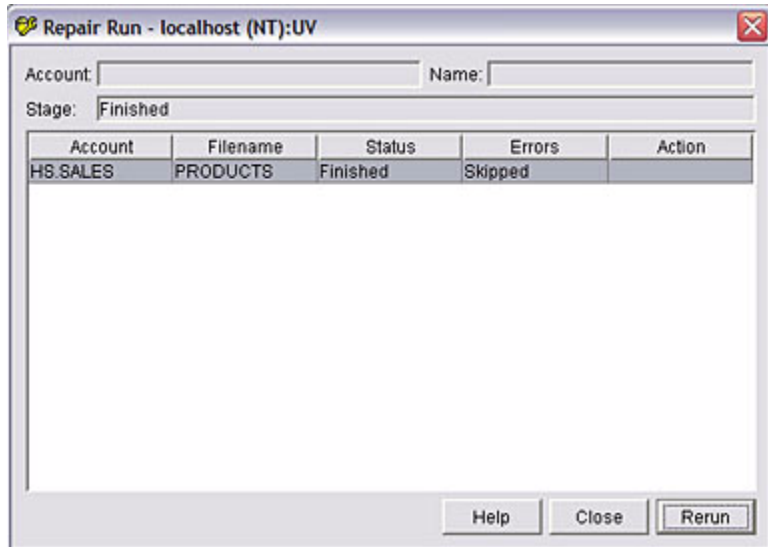
If you want to save a report of irreparable groups and record blocks detected by the diagnostic test, select **Output**, then enter the full path to a directory where you want to store the output.

If you want to store a copy of the error report, select **Logging**, then enter the full path to a directory where you want to store the report.

If you do not specify **Output** or **Logging**, the output and error report are stored in the directory where the file currently resides.

Click **OK** to save your changes and run the Repair program, or click **Close** to run the Repair program with default settings. The repair program starts.

When the Repair program completes, a dialog box similar to the following example appears:



The **Repair** dialog box displays the following information:

- The account name where the file resides
- The file name
- The current stage of the program
- The number of errors encountered
- What action to take if an error is found

If an error is encountered, click the account name to display the **Error Report** window.

Rerun Repair Program

After you execute the Repair program, click **Rerun** to reexecute the program to ensure no errors remain in the file.

Exit the Program

Click **Close** to exit the dialog box and return to the **File Tool** window.

- Repair the file if it is damaged

The Format Conversion Utility

The format conversion utility converts UniVerse database files, tables, and BASIC object code from one machine's storage format to another. Use this utility when you are transferring data files and tables between machines with different architectures. You can also use this utility when you are transferring tables from one schema to another on the same system. You can use the *format.conv* command from a UNIX shell or an MS-DOS window, or you can use the *FORMAT.CONV* command in UniVerse.

You can use these commands in one of three ways:

- To convert the storage format of data files and BASIC object code.
- To prepare tables for transfer to another schema, on either the same or a different system.
- To convert the storage format of tables and reconstitute them in the schema to which they have been transferred.

See *UniVerse SQL Administration for DBAs* for a description of how to export and import tables from one schema to another.

Converting the Format of Data Files and UniVerse BASIC Code

The command options let you specify the following:

- The target machine class, which determines the byte order of the converted output
- The target machine type for low-order and high-order byte storage
- Conversion of database files from formats compatible with later UniVerse releases to formats compatible with earlier UniVerse releases
- Whether or not to display information on your screen during processing

You can convert the file and transport it to the target machine, or you can transport the file and convert it on the target machine. Once a file has been converted, you can at any time use the format conversion utility to reconvert it. You can use the format conversion utility any number of times on the same file. The format conversion utility does not alter the logical contents of a file, such as the stored data. It alters only its physical format by modifying byte-ordering.

Converting to the Current Machine Format

The format conversion utility changes the format of files or BASIC programs from a machine having a different architecture so they can be used on the current machine.

The command to convert either a data file or a file containing BASIC object code to use on the current machine has the following syntax:

format.conv *filename*

filename specifies the name of the file whose format you want to convert. If you do not specify a file name, the format conversion utility reads file names from standard input. For example, the following command converts a file called *anyfile* to the format used on the current machine:

\$ format.conv anyfile

The format conversion utility identifies the byte storage format of the current machine. At the UniVerse prompt you can use the FORMAT.CONV command. The command syntax is identical for operating system and UniVerse commands. Be careful to type a lowercase letter when specifying an option for the UniVerse FORMAT.CONV command.

Converting to a Different Machine Format

You can use the format conversion utility to prepare data files or BASIC programs when you want to export them from the current machine to a machine having a different architecture. You must specify which machine format or class to use. (The default is the class of the machine on which you are running.) The basic syntax of the command is as follows:

format.conv *-format.option filename*

-format.option specifies the format of the target machine. The following sections describe the format options.

Explicitly Specifying the Machine Class

Use the following form of the *format.conv* command to convert a file to the format of the specified machine class:

format.conv -mclass filename

Specify a minus sign followed by the letter *m* followed by the machine *class* number (no spaces are allowed). Valid class numbers are 0, 1, or 16.

For example, the following command converts *basic.file* (which contains BASIC object code) to use on a class 16 machine:

\$ format.conv -m16 BP.O/basic.file

If you specify multiple options and include *-mclass*, always specify this option last. For example, the following command specifies Release 6 format while converting *data.file* to use on a class 16 machine:

\$ format.conv -6m16 data.file

For more information about specifying the *-6* option, see [“Converting to a Different Machine Format”](#) on page 22.

Specifying Low-Order Byte Addressing Format

Use the *-u* option of the *format.conv* command to convert a file to low-order byte addressing format (big-endian), or “unix” (u) format:

format.conv -u filename

When you use the *-u* option to convert a data file, the conversion command assumes a machine class of 0 or 16. For a data file, the *-u* option produces the same result as the *-m0* or *-m16* option. For example, the following command converts *data.file* to use on a class 0 or class 16 machine:

\$ format.conv -u data.file

When you use the *-u* option to convert a file containing BASIC object code, the conversion command assumes a machine class of 0. For a file containing BASIC object code, the *-u* option produces the same result as the *-m0* option. For example, the following command converts *basic.file* to use on a class 0 machine:

\$ format.conv -u BP.O/basic.file

For information about converting a file containing BASIC object code to use on a class 16 machine, see [“Explicitly Specifying the Machine Class”](#) on page 23.

Specifying High-Order Byte Addressing Format

Use the `-x` option of the `format.conv` command to convert a file to high-order byte addressing format (little-endian), or “xinu” (x) format.

```
format.conv -x filename
```

When you use the `-x` option to convert a file, the conversion command assumes a machine class of 1. The `-x` option produces the same result as using the `-m1` option. For example, the following command converts `data.file` to use on a class 1 machine:

```
$ format.conv -x data.file
```

Specifying Format Codes and Machine Class Options

For information about these topics, see the `FORMAT.CONV` command in *UniVerse User Reference*.

Converting a File to an Earlier UniVerse Release Format

If you want to move a file from a later UniVerse release to an earlier one, you must use the format conversion utility to convert the files from the later file format to a format compatible with the earlier release. The procedure differs depending on which releases of UniVerse are involved.

Release 6 Through Release 9.5.1B

To convert a file from a later release of UniVerse to an earlier one is a three-step process:

1. On the machine running the later release of UniVerse, use the `FORMAT.CONV` command with the `-6` option to convert the file to UniVerse Release 6 format.
2. Transfer the converted file to the machine running the earlier release of UniVerse.
3. On the machine running the earlier release, use the `RESIZE` command on the file to make it compatible with the current release of UniVerse.

Release 9.5.1C or Later

To convert a file created or resized on a 9.5.1C or later release of UniVerse to a format compatible with Release 9.5.1B or earlier is a four-step process:

1. On the machine running the later release of UniVerse, use the `FORMAT.CONV` command with the `-o` option to convert the file to UniVerse Release 9.5.1B format.
2. To convert the file in 9.5.1B format to a format compatible with Release 9.4.1 or earlier, use the `FORMAT.CONV` command with the `-6` option.
3. Transfer the converted file to the machine running the earlier release of UniVerse.
4. On the machine running the earlier release, use the `RESIZE` command on the file to make it compatible with the current release of UniVerse.

Syntax

The basic syntax of the command to convert a data file to UniVerse Release 6 is:

`format.conv -6 filename`

A minus sign (`-`) immediately precedes the 6 (no spaces are allowed).

filename specifies the name of the file whose format you want to convert. If you do not specify *filename*, the format conversion utility reads file names from standard input.

Alternately, you can use the `FORMAT.CONV` command at the UniVerse prompt. For example, the following commands convert *beta.file* from Release 9.5.1C format to Release 6 format:

```
> FORMAT.CONV -o beta.file  
> FORMAT.CONV -6 beta.file
```

When you use the `-6` option to convert a file to Release 6 format, the *format.conv* command performs the following actions:

- It changes item padding for the stored file.
- It writes a file revision level identifier in the file header which is stored in the front of the file. For example, Release 6 files have the following file-level identifier (in hexadecimal form): `0xacef0106`.



Once you convert a file to Release 6 format, you can at any time use the *format.conv* command to convert the file from one machine's storage format to another. And you can use the RESIZE command to convert the file from Release 6 format to the file format of current release of UniVerse on your system.

Note: *Security constraints prohibit using the format conversion utility to convert UniVerse SQL tables to an earlier file format. If you want to convert tables to an earlier file format, follow these steps:*

1. Use the CREATE.FILE command to create a temporary file.
2. Copy the contents of the table to the temporary file. (The copy operation fails if the user who is trying to copy the table does not have SQL SELECT privilege.)
3. Use the format conversion utility with the *-6* option to convert the temporary file to Release 6 format.

Silent or Verbose Output

You can specify *silent* or *verbose* options when you are using the format conversion utility. By default, the format conversion utility echoes its operation to the user's display screen. (The default setting is *verbose*.)

The basic syntax of the format conversion utility command with an output option is as follows:

format.conv *-output.option filename*

A minus sign (*-*) immediately precedes the output code (no spaces are allowed). Valid entries are *-s* (for silent) or *-v* (for verbose).

filename is the name of the file whose format you want to convert. If you do not specify a file name, the format conversion utility reads file names from standard input. For example, the following command silences output while converting *my.file* to use on the current machine:

\$ format.conv -s my.file

Or, for example, the following command toggles output on while reconverting *my.file* to use on a class 1 machine:

\$ format.conv -vx my.file

This command displays the following on the screen (with the verbose setting on):

Processing 'my.file'.

Converting All Files in an Account

In the UNIX environment you can use the format conversion utility to convert all the files in an existing account by using the UNIX *find* command. To do that, use the following command:

```
$ find . -print | format.conv -options
```

This form of the command finds all the files in the current directory and lists the file names on the user's terminal screen. The | causes the standard output of the *find* command (in this case, the filenames) to become the standard input to the *format.conv* command. (The format conversion utility reads file names from standard input.)

The *uvfixfile* Utility

A corrupted hashed file can cause an application to terminate abnormally. The *uvfixfile* utility lets you reestablish the structural integrity of a damaged hashed file. *uvfixfile* does not, however, recover all lost data in a damaged file, because the nature of file corruption makes it difficult to determine the validity of the data in the file.

The *uvfixfile* utility verifies a hashed file's integrity, examines file statistics, and repairs most broken files. However, *uvfixfile* cannot detect certain file breaks, and therefore it cannot fix them.

You can invoke the *uvfixfile* utility from a UNIX shell, an MS-DOS window, or the UniVerse environment. All users can use *uvfixfile* to verify file integrity and generate file statistics, but only a UniVerse Administrator can fix damaged files with *uvfixfile*.

Verifying File Integrity

Use *uvfixfile* to verify the integrity of dynamic or static hashed files. If you suspect a file is broken, use the following syntax at an operating system prompt:

uvfixfile -f pathname

In the UniVerse environment, enter one of the following:

UVFIXFILE filename

UVFIXFILE PATH pathname

filename is the name of a UniVerse file as defined in the VOC file; *pathname* is the relative or absolute path of the file.

uvfixfile traces through the file's groups, identifies problem groups, and reports their location to standard error. *uvfixfile* checks a file's structural integrity in three stages:

1. *uvfixfile* verifies the basic frame of the file. It checks the physical byte count against the file's original modulo and separation to see if the file is truncated.
2. For each primary group in the file, *uvfixfile* traces the chain of record blocks, verifying forward links, backward links, and flagwords.
3. *uvfixfile* checks the free buffer chain to ensure that all group buffers are accounted for.

If *uvfixfile* reports any problems, you can try to fix the file. Follow the instructions in [Fixing a Corrupt Hashed File](#).

Fixing a Corrupt Hashed File

To fix a broken UniVerse file, do the following:

1. Log on as a UniVerse Administrator.
2. Make a copy of the broken file. (IBM strongly recommends that you perform this step.)
3. At an operating system prompt, use the following syntax:

uvfixfile -f *pathname* -fix

In UniVerse, use the following syntax:

UVFIXFILE *filename* FIX

UVFIXFILE PATH *pathname* FIX

If *uvfixfile* cannot follow a record block link, it tries to find the next link. If it cannot find one, it sets an end-of-group marker at the last valid position and the group is truncated at its last valid record block. If an entire group buffer is unlinked, *uvfixfile* tries to relink the buffer after checking all other groups. In most cases it is easy to tell which primary group owns the orphaned group buffer because all record IDs found in the group should hash to the same primary group.

Examining File Statistics

The following is an example of using UVFIXFILE with the STATS option (the operating system form of the command is uvfixfile with the -s option) to examine file statistics. These commands may work when the UniVerse command FILE.STAT does not.

```
>UVFIXFILE VOC STATS
```

```
File Statistics
pathname      = VOC, size = 61440 bytes

file revision  = 0xacef0108
modulo        = 0x00000017 (000000023)
separation    = 0x00000004 (000000004)
file type     = 0x00000003 (000000003)
free list     = 0x00000000 (000000000)
header size   = 0x00000800 (000002048)
```

```
group size      = 0x00000800 (000002048)
```

```
Beginning TRACE of VOC.  
TRACE of VOC completed.
```

```
Scanning overflow buffers.  
Scan complete.
```

```
23 group(s) processed.  
29 group buffer(s) processed.  
820 record(s) processed.  
Number of data bytes = 38200.
```

The Revision field displays the magic number of the file.

Using Interactive Mode

Interactive mode lets you to look at the file more selectively.

Warning: *You should have some understanding of UniVerse internals before you use interactive mode. For your own protection, do not use interactive mode on anything except a copy of the file.*

To use interactive mode, use the following syntax at an operating system prompt:

```
uvfixfile -f pathname -i
```

In UniVerse, use the following syntax:

```
UVFIXFILE filename I
```

```
UVFIXFILE PATH pathname I
```

A colon prompt (:) appears. In interactive mode you can do the following:

- Trace through one or more groups in a file
- Step through one group buffer one record block at a time
- Reset one or more group buffers
- Open files specified by path or UniVerse file name
- Display the group to which a record ID hashes
- Map the primary and overflow buffer layouts

For more information about interactive commands, see the *UniVerse User Reference*.



Trace Command

The *trace* command checks a group or range of groups. The amount of information displayed is determined by the verbosity level (*-v*, *VLEVEL*).

The following example shows the output of a *trace* command. Note the difference in output when a different verbosity level is used.

```
# uvfixfile -f VOC -i
: trace 5

1 group(s) processed.
2 group buffer(s) processed.
43 record(s) processed.
Number of data bytes = 2416.
: vlevel 4
verbosity level set to 4.
: trace 5

Processing primary group: 5
.
.
.
>> Statistics for primary group 5.
>> Used  bytes = 1952
>> Free  bytes = 96
>> Total bytes = 2048

Processing overflow group: 25
0x0000c800 F:0x0000c9d0 B:0x000029d0 (0x00000000) W:0x00000400
SIZE = 0x000001d0
0x0000c9d0 F:0x00000000 B:0x0000ce30 (0x0000c800) W:0x00004000
SIZE = 0x00000630
>> Statistics for overflow group 25.
>> Used  bytes = 464
>> Free  bytes = 1584
>> Total bytes = 2048

1 group(s) processed.
2 group buffer(s) processed.
43 record(s) processed.
Number of data bytes = 2416.
: q
```

For more information, see *uvfixfile* or *UVFIXFILE* in the *UniVerse User Reference*.

Step Command

In interactive mode, use the *step* command when you want to examine one record block at a time. The *step* command provides the same information as the *trace* command. Verbosity level of the *step* command is at least 5. When you enter the *step* command, the prompt changes to *step>*. You can then press **Return** to display each record block one at a time.

The following example shows *step* command output:

```
# uvfixfile -f VOC -i
: step 5
0x00002800 F:0x00002818 B:0x00000018 (0x00000000) W:0x00000403
SIZE = 0x00000018
step> <Return>
0x00002818 F:0x00002848 B:0x00002830 (0x00002800) W:0x00000003
SIZE = 0x00000030
step> <Return>
0x00002848 F:0x0000285c B:0x0000280c (0x00002818) W:0x00000403
SIZE = 0x00000014
step> v 5
verbosity level set to 5.
0x00002848 F:0x0000285c B:0x0000280c (0x00002818) W:0x00000403
SIZE = 0x00000014
Record block contains pads.
Backup flag.
step> <Return>
0x0000285c F:0x00002874 B:0x00002850 (0x00002848) W:0x00000003
SIZE = 0x00000018
Backup flag.
step> <Return>
0x00002874 F:0x00002894 B:0x0000287c (0x0000285c) W:0x00000403
SIZE = 0x00000020
Record block contains pads.
Backup flag.
step> <Return>
0x00002894 F:0x000028a8 B:0x00002860 (0x00002874) W:0x00004000
SIZE = 0x00000014
Record block is free.
step> cont
0x000028a8 F:0x000028d4 B:0x000028b8 (0x00002894) W:0x00000003
SIZE = 0x0000002c
Backup flag.
0x000028d4 F:0x000028ec B:0x000028b0 (0x000028a8) W:0x00000003
SIZE = 0x00000018
Backup flag.
0x000028ec F:0x00002914 B:0x000028fc (0x000028d4) W:0x00000403
SIZE = 0x00000028
Record block contains pads.
Backup flag.
.
.
```

```

>> .
>> Statistics for primary group 5.
>> Used   bytes = 1952
>> Free   bytes = 96
>> Total  bytes = 2048

Processing overflow group: 25
0x0000c800 F:0x0000c9d0 B:0x000029d0 (0x00000000) W:0x00000400
SIZE = 0x000001d0
Record block contains pads.
0x0000c9d0 F:0x00000000 B:0x0000ce30 (0x0000c800) W:0x00004000
SIZE = 0x00000630
Record block is free.
>> Statistics for overflow group 25.
>> Used   bytes = 464
>> Free   bytes = 1584
>> Total  bytes = 2048

1 group(s) processed.
2 group buffer(s) processed.
43 record(s) processed.
Number of data bytes = 2416.
: a
>

```

For more information about the `uvfixfile step` command, see the *UniVerse User Reference*.

Set Command

Use the `set` command at the `step>` prompt to change the values of the record block header.

Warning: *Do not make changes to the record block header unless you are sure you know what you are doing. Otherwise you risk further damage to your files.*

The record block comprises three sections: the forward link, the backward link, and the flagword. To change the forward link, enter the following:

```
step>set flink n
```

To change the backward link, enter the following:

```
step>set blink n
```

To change the flagword, enter the following:

```
step>set flagword n
```



n is a number in hexadecimal format. To change the base to decimal, enter the following:

```
step>set decimal
```

To change the base to hexadecimal, enter the following:

```
step>set hex
```

Executing UniVerse Commands

Executing a Command	16-3
The UniVerse Command Output Window	16-6
Using the Command History	16-8
Reexecuting Commands	16-8
Editing a Command	16-8
Saving Commands	16-9

This chapter describes how to execute UniVerse commands from UniAdmin. As you issue commands, a command history is created, which you can use to reexecute a command or save commands to the VOC file.

Executing a Command

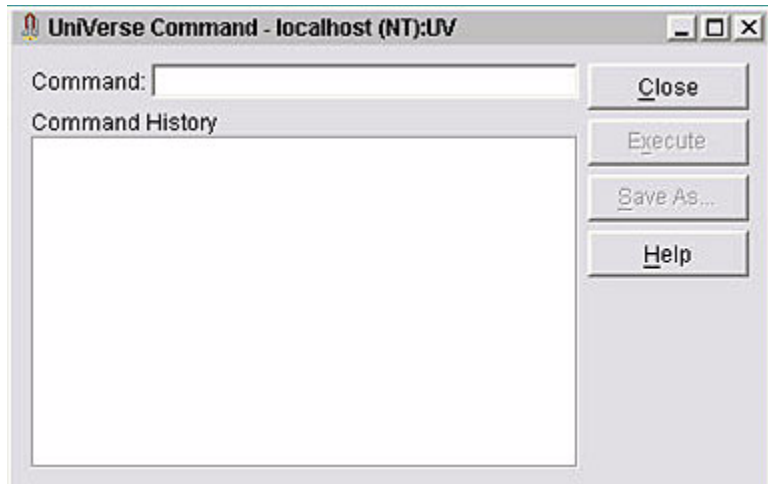
Complete the following steps to execute a UniVerse command:

1. Select one of the following methods to access the **UniVerse Command** dialog box:
 - From the **UniAdmin** window, double-click **UniVerse Command**.
 - From the **UniAdmin** menu, select **Admin**, then click **UniVerse Command**.
 - From the **UniAdmin** toolbar, click the **Manage Transaction Logging** icon, as shown in the following example.



UniVerse Command

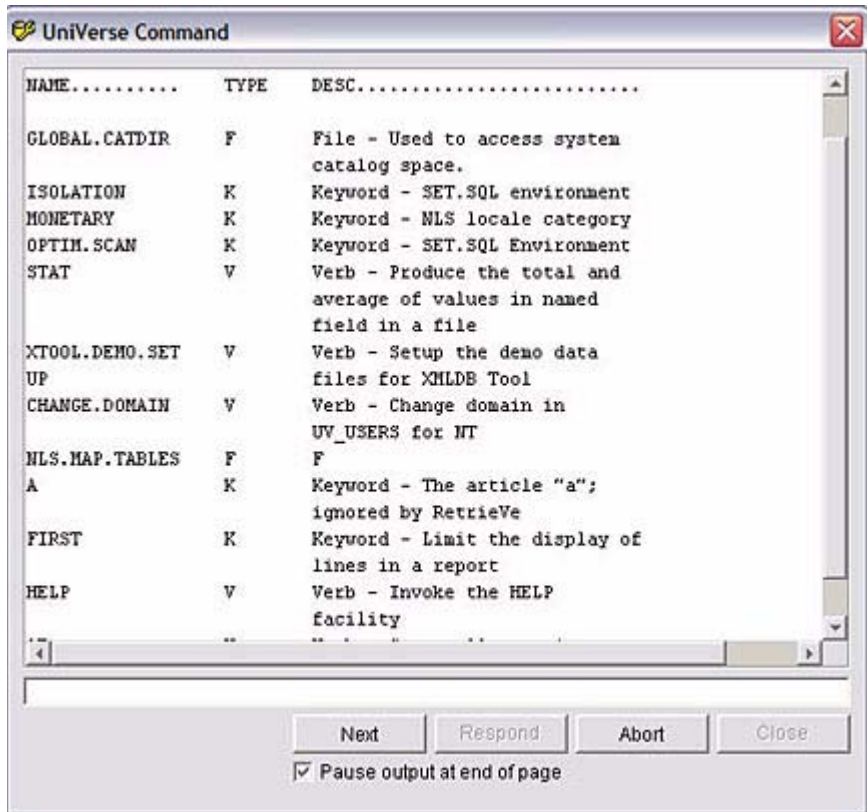
The UniVerse Command window appears, as shown in the following example:



1. Enter the UniVerse command in the **Command** field.

2. Click **Execute**. The result of the command appears in the **UniVerse Command** output window.

The following example illustrates the output from the LIST VOC command:



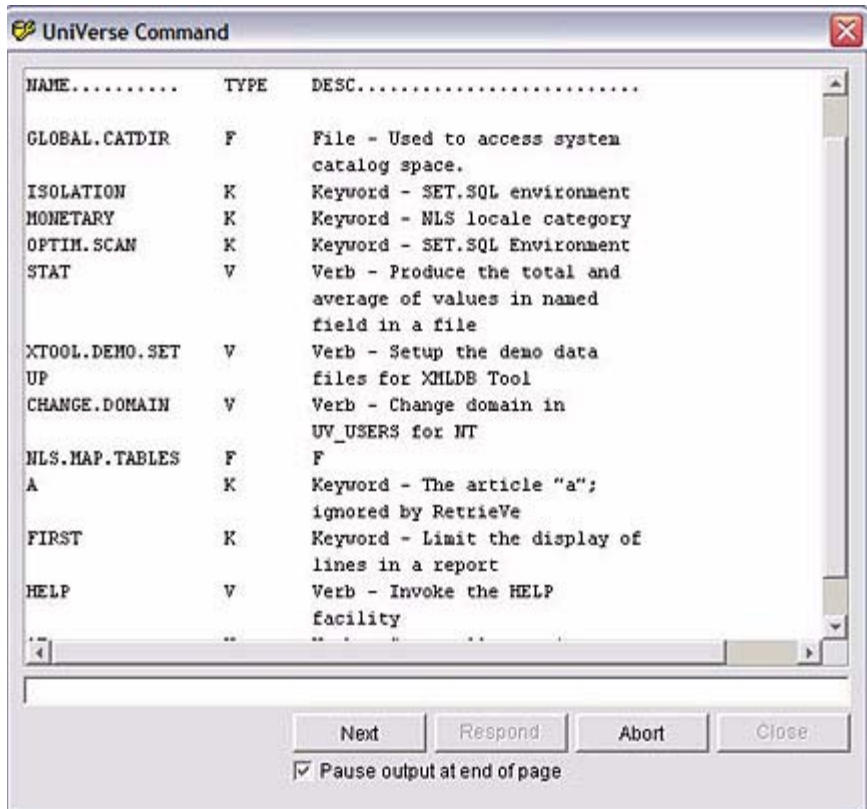
3. If the command output requires more than one page, click **Next** to continue the display. Clear the **Pause output at end of page** check box if you do not want the output to pause at the end of every page.
4. If the command requires user input, enter an appropriate reply in the text entry box, then click **Respond**. Repeat entering input until no further input is required.
5. To quit from command output, click **Abort**.
6. Click **Close** to close the **UniVerse Command Output** window. You can enter more commands, use the command history, or exit the dialog box.



***Note:** Use the **UniVerse Command** option to execute only a few UniVerse commands, because it is not a command shell.*

The UniVerse Command Output Window

The **UniVerse Command** output window displays the output of a UniVerse command. The output displayed can be a result of using the **UniVerse Command**, **Backup**, **Restore**, or **Transaction Logging** options of UniAdmin.



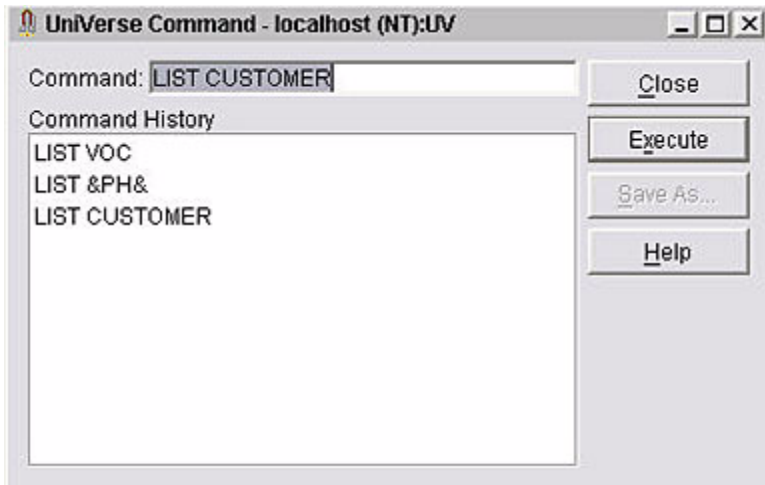
This window has the following components:

- **Output display area.** This is the main part of the window and contains the result of the UniVerse command. The number of lines displayed in this area depends on the size of the window. When the number of lines exceeds the display area, the output is paused. Click **Next** to continue the output. Increase the window size (or use the scroll bars) if you want to see more output.

- Text entry field. This field is used if the UniVerse command requires further user input. Enter an appropriate reply and click **Respond**.
- Buttons. There are four buttons on this window:
 - **Next**. Continues paused output.
 - **Respond**. Enters the response in the text entry field.
 - **Abort**. Aborts a command. This button is active when output is paused or when user input is required.
 - **Close**. Closes the UniVerse Command Output window.
- Check box. The **Pause output at end of page** check box affects how UniVerse pages the output. If selected (the default), output is paged and you must click **Next** to continue. If cleared, the output is not paged and scrolls continuously.

Using the Command History

You can reexecute, edit, or create sentences or paragraphs from commands listed in the **Command History** area of the **UniVerse Command** dialog box. The following example illustrates command history:



Reexecuting Commands

Complete the following steps to reexecute a command listed in **Command History**:

1. Double-click the command in the **Command History** portion of the **UniVerse Command** dialog box. The command appears in the **Command** box.
2. Click **Execute**. The result of the command appears in the **UniVerse Command Output** window.

Editing a Command

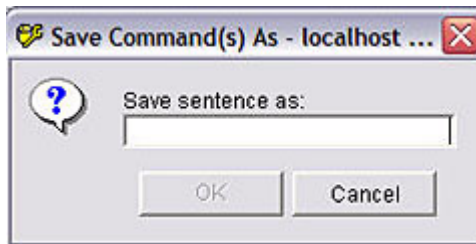
Complete the following steps to edit a command that appears in the **Command History** area of the **UniVerse Command** window.

1. Double-click the command in the **Command History** area. The command appears in the **Command** box.
2. Click the **Command** field and make your changes to the command.
3. Click **Execute** to run the new command. UniVerse adds the new command to the **Command History** area, and the result appears in the **UniVerse Command Output** window.

Saving Commands

Complete the following steps to save commands to the VOC file:

1. Select the commands you want to save from the **Command History** area of the **UniVerse Command** window.
2. Click **Save As**. The **Save Command(s)** dialog box appears, as shown in the following example:



3. If you are saving one command, enter a unique name for the sentence. If you are saving multiple commands, enter a unique name for the paragraph.
4. Click **OK** to save the new sentence or paragraph in the VOC file.

Sending Messages to Users

Sending Messages with UniAdmin	17-4
Sending Messages on UNIX Systems	17-4
Sending Message on Windows Platforms.	17-5
The UNIX write Command	17-6
The MESSAGE Command	17-7
Message of the Day on UNIX Systems	17-8

Beta Beta

You can communicate with users using four different mechanisms:

- UniAdmin
- UNIX *write*(1) command
- MESSAGE command
- UNIX message of the day

These mechanisms let you communicate directly or set up system processes to send messages at a specific time or when a particular event occurs.

Sending Messages with UniAdmin

You can send broadcast messages to one or more users from the **User Administration** dialog box. The options available from this dialog box vary depending on whether you are sending a message to users using a UNIX or Windows platform server.

Sending Messages on UNIX Systems

If you are connected to a UNIX server, you can send messages to a single user, all users, or to the system console. Complete the following steps to send a message.

1. From the **User Administration** dialog box, click **Message**. The **Send Messages** dialog box appears, as shown in the following example:



2. Choose the users that you want to receive the message. Valid options are:
 - **All Users** – The message will be sent to all users logged on to the system, not just those logged on to UniVerse.
 - **System Console**
 - **User name or Tty** – Using this option, you can enter the name of a user or tty, or select a user from this list. This list contains all of the users in the **Interactive Users** list.

Note: *If you select a user before you click **Message**, the **User name or Tty** option is automatically set, and the user's name is selected.*

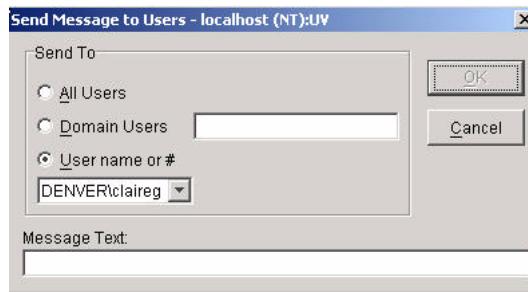
3. Enter your message in the **Message Text** box.
4. Click **OK**. The message appears in the user's current session if the user has messaging enabled.



Sending Message on Windows Platforms

If you are connected to a Windows server, you can send messages to a single user, all UniVerse users, or the UniVerse users in a domain you specify. Complete the following steps to send a message:

1. Click **Message**. The **Send Message** dialog box appears, as shown in the following example:



2. Choose the users to which you want to send the message. Valid options are:
 - **All Users** – All the UniVerse users.
 - **Domain Users** – When you choose this option, you must enter the name of a domain.
 - **User Name or #** – When you choose this option, you can enter the name or number of a user, or you can choose a user from the list. This list contains all users in the **Interactive Users** list.

Note: *If you select a user before you click **Message**, the **User name or #** option is automatically set, and the user's name is selected.*

3. Enter your message in the **Message Text** box.
4. Click **OK**. The message appears in the user's current session if the user has messaging enabled.



The UNIX *write* Command

The UNIX *write*(1) command lets you send a message directly to any system user. This command writes to the terminal to which a user is logged in, even if the user is running an active process (such as the text editor) that uses the terminal.

Users can use the *mesg*(1) command to turn off permission for *write*. Depending on your need to communicate quickly with users, you may want to establish guidelines for using these commands. You should generally use *write* only for time-critical messages. If you use it too frequently, users may tend to turn off this capability.

The MESSAGE Command

If you are working in the UniVerse environment, you can use the MESSAGE command as you would the UNIX *write* command. It interrupts whatever a user is doing and displays a message on the screen. Use the `-STATUS` option to display the status of receive mode for all users.

Message of the Day on UNIX Systems

On UNIX systems when a user logs in to the system, the login procedure prints out a message of the day, if any messages containing today's date are found in the file */etc/motd*.

You can change the contents of this file using the text editor. Two messages are shown:

```
9/23: Reboot at 5 pm today
9/24: Down for PM from 9 to 11 am tomorrow
```

The */etc/motd* file should be used for important messages. Obvious possibilities are messages about scheduled down-time for maintenance, cleanup messages for space-low file systems, or other useful warnings.

You should keep messages in this file short and to the point so that users are not delayed unnecessarily.

Adding Capabilities to UniVerse

Adding UniVerse BASIC Applications	18-3
Managing Catalog Space	18-5
Initializing System Catalog Space	18-5
Checking the Status of the Catalog	18-6
Displaying Catalog Contents.	18-6
Deleting Programs from the Catalog	18-7
Managing Catalog Shared Memory	18-9
Setting Up Catalog Shared Memory	18-9
Defining Programs to Run in Shared Memory	18-11
Adding Programs to the SHM.TO.LOAD File	18-12
Removing a Program from the SHM.TO.LOAD File	18-12
Loading Programs into Catalog Shared Memory	18-13
Using Programs Stored in Catalog Shared Memory	18-15
Modifying Programs in Catalog Shared Memory.	18-16
Updating a Program in Shared Memory	18-16
Removing a Program from Shared Memory	18-17
Removing the Catalog Shared Memory Segment.	18-18
Adding Commands to the VOC File.	18-19

Although UniVerse includes many programs and utilities, you can make additional programs available to users. These include applications written in UniVerse BASIC and non-UniVerse applications written for the UNIX or Windows environment but accessible to UniVerse users.

This chapter discusses system administration considerations when making additional programs available to UniVerse users:

- Adding UniVerse BASIC applications
- Managing the catalog space used by UniVerse BASIC
- Managing catalog shared memory for UniVerse BASIC programs
- Adding non-UniVerse programs to VOC files

Adding UniVerse BASIC Applications

UniVerse BASIC is the principal programming language in the UniVerse environment. It has many features that are designed to work in the database management environment. It allows the creation of specialized business applications.

UniVerse BASIC source code is stored in a type 1 or type 19 file. When you compile a UniVerse BASIC program, object code is created and stored in another type 1 or type 19 file whose name is the same as the source code file with the suffix .O added. To UniVerse, each program appears as a record in the source file. To the operating system, the program file is a directory, and each source program is a separate file in that directory. This means you can create program source code using either the UniVerse Editor or a text editor such as *vi*.

Source code is compiled with the BASIC command. Here is the syntax:

BASIC *filename* [*programs* | *] [*options*]

filename is the name of a type 1 or type 19 file containing BASIC programs.
programs are the record IDs of the programs to compile. An * (asterisk) specifies all programs in the file.

Use the UniVerse RUN command to run compiled programs. Here is the syntax:

RUN [*filename*] *program* [*options*]

UniVerse automatically appends the suffix .O to *filename* to access the file containing the object code. If *filename* is omitted, BP.O is assumed to contain the object code.

program is the record containing the object code. It should have the same name as the record that contains the source code.

If programs are cataloged, you can invoke and run them directly from the UniVerse prompt; you need not use the RUN command to run cataloged programs. Cataloging programs and object modules to the system catalog space allows all users to share them.

To catalog a program, that is, to copy a compiled program to catalog space, use the CATALOG command (see *UniVerse User Reference*). All external subroutines *must* be cataloged before they can be called from within a BASIC program with the CALL statement. Cataloging other program types is optional.

Unless you catalog the program globally, an entry is made in your VOC file pointing to the record ID of your program in the nonhashed file you set up for it.

You can also use the UniVerse Editor or ReVise to add UniVerse BASIC programs to the VOC file.

Managing Catalog Space

Users developing UniVerse BASIC programs generally catalog their own modules. The administrator's job is to initialize the system catalog and to periodically purge it of unused entries to improve its performance. There is no fixed limit on the size of catalog space. However, the more entries there are, the slower the access, so unused modules should not remain cataloged.

The catalog contains normally and globally cataloged programs. It does not contain locally cataloged programs, which are for use in a particular account. Normally and globally cataloged programs are accessible to all users.

Normally cataloged programs have names of this form:

**account*catalog.name*

Globally cataloged programs have catalog names with *, -, \$, or ! as the initial character.

Globally and normally cataloged programs are stored in the *catdir* file in the UV account directory. The file descriptor GLOBAL.CATDIR references the *catdir* file from any UniVerse account.

Initializing System Catalog Space

The INITIALIZE.CATALOG command lets you completely delete the system catalog space and reinitialize it.



Warning: *The impact of the INITIALIZE.CATALOG command is enormous. You should be absolutely sure that you want to initialize the catalog space. If the slightest doubt exists, you should not respond with Y to the Continue Initialization prompt.*

This example reinitializes the system catalog space:

```
>INITIALIZE.CATALOG
*****
*                                     WARNING
*
*      You are about to destroy the system Catalog space. This *
may have a significant effect upon the user community.
*
*****
Continue Initialization of the Catalog space? (Y/N)Y

Initialization of CATALOG space completed.
```

Checking the Status of the Catalog

Use the VCATALOG command to compare the object code of a program in the catalog with the object code contained in your account. It verifies that the cataloged version of the program is identical to the object code in your account. VCATALOG has the following syntax:

VCATALOG [*filename* [[*catalog*] [*program*]]] [**LOCAL**]

filename is the name of the file that contains the source code for the program being compared. *catalog* is the name of the program in the system catalog space. *program* is the record in the file *filename.O* that contains the object code of the program to be compared. **LOCAL** specifies a locally cataloged program.

Displaying Catalog Contents

Use the MAP command to display the contents of the system catalog. It displays each program and subroutine in the catalog in alphabetical order. MAP also displays the following:

- Date the item was cataloged
- Account name from which the item was cataloged
- Number of times the item has been run since being cataloged
- Number of variables
- Number of common segments
- Number of bytes of executable object code
- Size of the cross-reference table in bytes

- Total number of bytes, consisting of the executable object code, the cross-reference table, and the symbol table

Use the MAKE.MAP.FILE command to create a UniVerse file called &MAP&. This file contains the data displayed by the MAP command. By creating the &MAP& file, you can use Retrieve commands to produce reports on the catalog space.

Here's an example of a listing produced by MAP:

```
>MAP
Catalog Name      Date   Who Ref Var Seg   Obj    CR   Size
*DEMO*AD.NAME     040590 DEMO  0  21  4   2063   5316  7399
*DEMO*AMT.PCT.DISP 040590 DEMO  0   7  0    242    408   670
*DEMO*ANT.UPDATE   040590 DEMO  1  21  0    761   1052  1833
*DEMO*AP.OPENS     040590 DEMO  0   2  1   1039   1328  2387
*DEMO*AR.BAL.CLEAR 040590 DEMO  0   2  1    231   2160  2411
*DEMO*AR.BAL.DISP  040590 DEMO  0   9  1    486   2420  2926
*DEMO*AR.TYPE      040590 DEMO  0  20  3    869   4284  5173
```

The two columns of main interest to you are the date the item was put in the catalog and the number of references. If the item has been cataloged for some time and has never been referenced, it may be appropriate to delete it from the catalog. In addition you can check for duplicate entries.

If a &MAP& file exists, you can use Retrieve to produce reports on the contents of the catalog space. (The &MAP& file is generally contained in the UV account, with pointers to it from other accounts.) Because the &MAP& file is out of date when someone adds a program to the catalog space, create the &MAP& file just before producing a Retrieve report on its contents.

Deleting Programs from the Catalog

In a development environment it is not unusual to find several versions of the same program in the catalog. Delete outdated versions with the DELETE.CATALOG command. DELETE.CATALOG removes a cataloged program from the system catalog space and makes it unavailable to subsequent calls. The DELETE.CATALOG command has the following syntax:

DELETE.CATALOG *catalog.name*

For example, to delete the module *DEMO*AR.TYPE, enter:

```
>DELETE.CATALOG *DEMO*AR.TYPE
```

Managing Catalog Shared Memory

You can load cataloged UniVerse BASIC programs into shared memory and run them from there. Shared memory reduces the amount of memory needed for multiple users to run the same program at the same time. The program also starts a little faster because it's already in memory.

For example, if 21 users are running the same UniVerse BASIC program at the same time without catalog shared memory, and the program code requires 50 Kbytes of memory, the total amount of memory used by everyone running that program is 21×50 , or 1050, Kbytes. On the other hand, if the program is loaded into catalog shared memory, all 21 users can run one copy of the program, which uses only 50 Kbytes of memory. In this example catalog shared memory saves 1000 Kbytes (1 megabyte) of memory.

Setting Up Catalog Shared Memory

Before users can use programs in catalog shared memory, the administrator must designate which programs are available and load them into shared memory. UniAdmin simplifies this process. You can specify any BASIC programs to be run in shared memory, including those listed in the system catalog space.

Use the **UniVerse Catalog Shared Memory** dialog box to manage catalog shared memory.

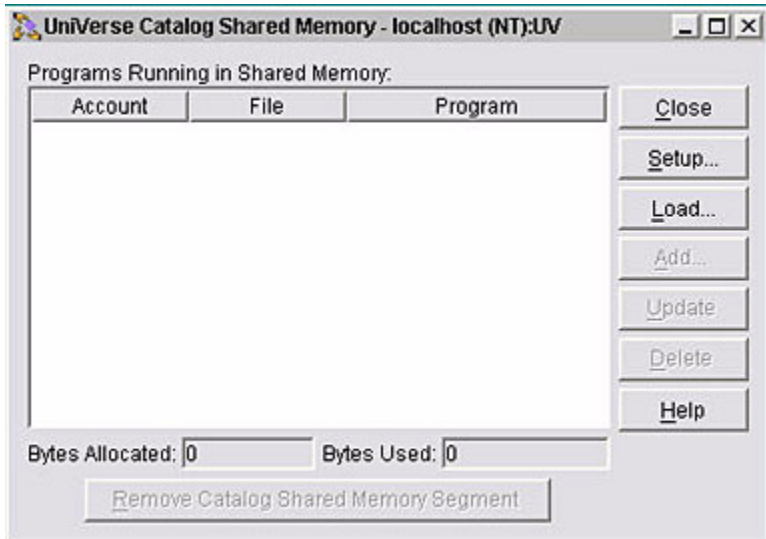
Select one of the following methods to access the **UniVerse Catalog Shared Memory** dialog box:

- From the UniAdmin window, double-click **Shared Programs**.
- From the UniAdmin menu, select **Admin**, then click **Shared Programs**.
- From the UniAdmin toolbar, click the **Manage Shared Programs** icon, as shown in the following example.



Manage Shared Programs

The following example illustrates the **UniVerse Catalog Shared Memory** dialog box:



The **UniVerse Catalog Shared Memory** dialog box contains a list of the programs currently running in shared memory. It also displays the amount of shared memory allocated, in bytes, and how much of this memory is currently being used.

Note: *The Catalog Shared Memory window is empty if shared memory is not loaded.*

The tasks you can perform from this window include:

- Defining programs to run in shared memory
- Loading catalog shared memory
- Modifying programs in shared memory
- Removing the catalog shared memory segment



Defining Programs to Run in Shared Memory

The amount of memory available for catalog shared memory is limited. This limit varies from machine to machine. The administrator determines how best to use this space by designating the programs to run from catalog shared memory.

The SHM.TO.LOAD file in the UV account directory contains details, such as account, file, and program names, of the programs selected to run in shared memory. When shared memory is loaded, this file is read to determine which programs are to be loaded. An appropriate amount of shared memory is then loaded.

***Note:** Changes made to the SHM.TO.LOAD file take effect only the next time you start UniVerse or load shared memory.*

To define the programs to be run in shared memory, click **Setup** from the **UniVerse Catalog Shared Memory** dialog box. A dialog box similar to the following example appears:



The dialog box titled "Setup SHM.TO.LOAD File" contains the following elements:

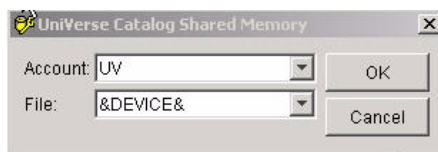
- Programs in Current File:** A list box containing the following programs: IADDS, IAMLC, IANDS, IASYN, IBINARY.CONVERT, IB (highlighted), ICATS, ICHARS, ICHECK.TYPE1.ID, ICLEAR.PROMPTS, and ICOMMAND EDITOR. A tooltip "Select program to add" is visible over the "IB" entry.
- Buttons:** "Add >" and "< Remove" buttons are located between the two list boxes.
- Selected Programs:** An empty table with the following headers: Account, File, Program, and Size.
- Current File:** Fields for "Account:" (containing "UV") and "File Name:" (containing "catdir"). A "Change..." button is located below the File Name field.
- Memory (bytes):** Fields for "Required:" (containing "0") and "Maximum:" (containing "2097152").
- Buttons:** "OK", "Cancel", and "Help" buttons are located at the bottom right.

This dialog box enables you to choose programs to add to the SHM.TO.LOAD file. If the SHM.TO.LOAD file does not exist, a warning message window appears. You must acknowledge this message before you can choose any programs. UniVerse creates the SHM.TO.LOAD file when you click **OK**.

Adding Programs to the SHM.TO.LOAD File

Complete the following steps to add a program to the SHM.TO.LOAD file:

1. Choose one or more programs from the **Programs in Current File** list. This list displays the programs in the catalog space.
2. Click **Add**. The updated **Selected programs** list includes the additional programs.
3. If you want to choose a program in a file other than *catdir* (the catalog space), click **Change**. The **Change Current File** dialog box appears, as shown in the following example:



4. Select a new account and file from the **Account** and **File** lists.
5. Click **OK**. The **Setup SHM.TO.LOAD File** dialog box reappears with the updated program list.
6. Select one or more programs from the **Programs in Current File** list.
7. Click **Add**. The updated **Selected Programs** list includes the additional programs.
8. Click **OK** to save the changes and close the **Setup SHM.TO.LOAD File** dialog box.

Removing a Program from the SHM.TO.LOAD File

Complete the following steps to remove a program from the SHM.TO.LOAD file:

1. Click **Setup** from the **UniVerse Catalog Shared Memory** dialog box. The **Setup SHM.TO.LOAD File** dialog box appears.
2. Select one or more programs to remove from the **Selected Programs** list.

3. Click **Remove**. The **Selected Programs** list and the memory details are updated to reflect the deletions.
4. Click **OK** to save the changes and close the **Setup SHM.TO.LOAD File** dialog box.

Loading Programs into Catalog Shared Memory

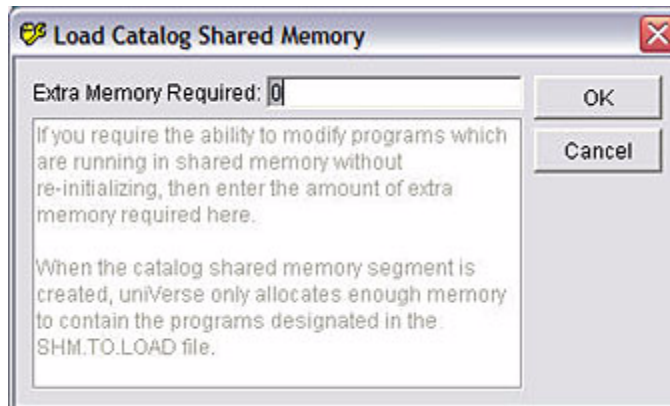
After designating the programs to be loaded into catalog shared memory, you must create the shared memory segment and load the programs into memory. This is also routinely done each time UniVerse starts.

When you load shared memory, you can do one of the following:

- Load just enough memory to run the programs in the SHM.TO.LOAD file.
- Load additional memory, which is then available if you want to modify the programs you are running.

Complete the following steps to load shared memory:

1. From the **UniVerse Catalog Shared Memory** dialog box, click **Load**. The **Load Catalog Shared Memory** dialog box appears, as shown in the following example:



2. If you want to load additional (spare) memory, enter a new value in the **Extra Memory Required** box. This defines the additional amount of shared memory, in bytes, to load.

3. Click **OK**. The existing shared memory is destroyed and the shared memory is reloaded.

The Waiting for Catalog Shared Memory Message

When UniVerse starts on UNIX systems, it tests to see if catalog shared memory is being loaded. UniVerse waits for loading to complete before allowing users to log on to the system. While waiting, UniVerse displays the message “Waiting for Catalog Shared Memory...”

If this message appears and catalog shared memory is **not** being loaded, you should execute the **uv** command with the *-admin -c* options. To execute this command, log on as a UniVerse administrator, then entering the following command from the shell prompt:

```
# /usr/ibm/uv/bin/uv -admin -c
```

This command tells UniVerse that catalog shared memory is not in use, and lets users log in to the UniVerse system.

Using Programs Stored in Catalog Shared Memory

UniVerse invokes programs in the same way, whether they are stored in catalog shared memory or in a disk file. UniVerse attaches to the catalog shared memory only when it starts. Therefore, anyone who starts UniVerse before catalog shared memory is loaded always uses programs from disk. To take advantage of catalog shared memory, those users must leave UniVerse (by entering **Q** at the prompt), and start it again. Even the system administrator must leave UniVerse and restart it to take advantage of the programs in shared memory.

Changing a BASIC program and running the BASIC command changes the copy of the object code stored in the system catalog, but does not change the copy stored in catalog shared memory.

Modifying Programs in Catalog Shared Memory

Once programs are installed in catalog shared memory, they are not affected by any changes made to the BASIC program on disk. If you modify a program, you must install it again.

You can add, remove, and update programs running in shared memory without having to reload it. However, to do this you must have additional (spare) shared memory loaded. When you modify programs in shared memory, it acts only as a temporary measure, and no changes are made to the SHM.TO.LOAD files.

You can perform the following modification tasks from the **Catalog Shared Memory** dialog box:

- Add a program to shared memory
- Update a program in shared memory
- Remove a program from shared memory

Updating a Program in Shared Memory

If a program has been edited and recompiled, you can load the latest version of the program into shared memory. The memory used for the “old” version of the program is not made available, so the updated program will be loaded only if enough spare memory is available.

Complete the following steps to update a program in shared memory:

1. Select the program to update from the **Catalog Shared Memory** dialog box.
2. Click **Update**. A message window appears.
3. Click **Yes**. UniVerse checks that there is enough memory available to load the updated program. If there is not, a message window appears. You must acknowledge the message. If there is enough space, UniVerse loads the program into shared memory and updates the **Catalog Shared Memory** dialog box.

Removing a Program from Shared Memory

You can remove programs from shared memory without reloading. However, the memory used by this program is not made available for reuse.

Complete the following steps to remove a program from shared memory:

1. Select the program to remove from the **Catalog Shared Memory** dialog box.
2. Click **Delete**. A message window appears.
3. Click **Yes**. UniVerse updates the **Catalog Shared Memory** dialog box.



Removing the Catalog Shared Memory Segment

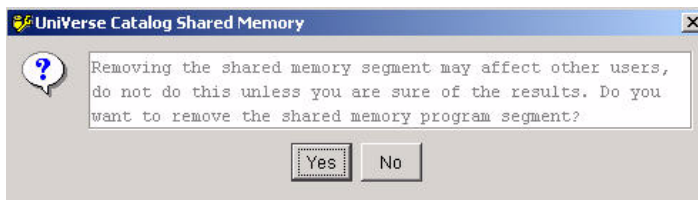
You can remove the catalog shared memory segment without reloading.

Note: *You must be sure you want to remove the catalog shared memory segment, because it can have an impact on other UniVerse users.*

If any users are running programs in shared memory when you delete the catalog shared memory segment, they can continue to run the program from the standard UniVerse catalog.

Complete the following steps to delete the catalog shared memory segment:

1. Click **Remove Catalog Shared Memory Segment** from the **Catalog Shared Memory** dialog box. The following dialog box appears:



2. Click **Yes** to delete the shared memory segment, or click **No** to exit.

Adding Commands to the VOC File

You can make standard operating system utilities and additional system and UniVerse BASIC applications available to UniVerse users through the VOC files in their accounts. Capabilities you want to make available to all accounts can be added to the NEWACC file of the appropriate flavor.

Using either ReVisé or the UniVerse Editor, add a verb to the VOC. (See *UniVerse System Description* for a discussion of how to use ReVisé and *Guide to the UniVerse Editor* for information on using the UniVerse Editor.) The first field of the VOC entry (defined in the VOC dictionary as F1) defines the entry type and gives an optional description. The first one or two characters in F1 define the type of VOC entry. For an executable program, the type must be V for verb.

The rest of F1 is available for an optional description of the record. F1 might look like the following example:

V VI, the UNIX text editor

The second field (F2) contains the *processor name*. This is the name of the program to be executed.

For an operating system command, this field would contain the full path of the program to be executed. For a UniVerse BASIC program, this field would contain the path of the object code, unless the name begins with *, \$, –, or !, which indicates that the object module can be found in catalog space. You can use the MAP command to find the name of an entry in catalog space. Globally cataloged entries are available to all users of the system and have names in the form:

*NAME

The user who catalogs the program can use \$, –, or ! in place of *. Standard normal cataloged entries have names in the form:

*ACCOUNT_NAME*PROGRAM_NAME

Note: *Locally cataloged programs are not entered in the system catalog space and are not listed by either the MAP or the MAKE.MAP.FILE command.*



The third field (F3) specifies the *dispatch type*. This field should normally contain a B for a BASIC program or a U for an operating system command. Standard UniVerse programs in `/uvhome/bin` are type E. For these programs, field 2 contains the name of the executable file in `/uvhome/bin`. The following is a full list of dispatch types:

Code	Dispatch Type	Use
B	Cataloged BASIC program	Used with the BASIC CALL statement, or from the UniVerse prompt.
C	C shell script	Used to call C shell scripts from the operating system. Available by default on UNIX; can be used on Windows platforms where a suitable processor exists.
D	DOS batch files	Used to call DOS batch files, available on Windows platforms only.
E	External	Reserved for internal UniVerse use only.
I	Internal	Reserved for internal UniVerse use only.
P	Primitive command	Provided for compatibility with the PI/open PR command.
Q	Query command	Reserved for internal UniVerse use only.
S	Bourne shell script	Used to call Bourne shell scripts from the operating system. Available by default on UNIX; can be used on Windows platforms where a suitable processor exists.
U	Operating system command	Makes a call to an operating system command.

Dispatch Types

The fourth field (F4) specifies additional information that may be needed by the program. This is referred to as the *processor mode*. It can be one or more of the following:

Code	Processor Mode
A	Use alternative query syntax. Reserved for internal UniVerse use only.
B	<i>Not used.</i>
C	Allow COMO files to be used with external programs and operating system commands.
D	Pass DATA to subprocess. Reserved for internal UniVerse use only.
E	Use Win32 expansion routine (Windows platforms only).
F	Pass format via environment variable. Reserved for internal UniVerse use only.
G	Allowed in an SQL CALL statement.
H	EXECUTE and PERFORM can use in a transaction.
I	Interrupt control.
K	Keep the select list. Reserved for internal UniVerse use only.
M	(NLS) Map output from EXECUTE CAPTURING to internal character set.
N	Do not set @SYSTEM.RETURN.CODE.
P	Allow parenthetical options on the command line.
Q	Use SQL mode. Reserved for internal UniVerse use only.
R	Backslashes (\) can be used to quote strings.
S	Use the active select list. Reserved for internal UniVerse use only.
T	Change the terminal mode.
U	Add path of the UV account directory before the processor name.
V	Function specified by field 5. Reserved for internal UniVerse use only.
X	Read DATA stack if there is an active select list. Reserved for internal UniVerse use only.

Processor Modes

Here is an example that demonstrates how to use the CATALOG command to add a BASIC program called PAYROLL to the VOC file:

```
>CATALOG BP PAYROLL PAYROLL LOCAL
```

You could also use the UniVerse Editor to create the VOC entry:

```
>ED VOC PAYROLL
---: I
001: V BASIC program to do payroll processing
002: BP.O/PAYROLL
003: B
004: <Return>
---: FILE
```

A corresponding entry for a UNIX program might look like this:

```
001: V Vi, the UNIX text editor
002: /bin/vi
003: U
004: TI
```

The TI processor mode indicates that there is a terminal mode change and that interrupt control is enabled.

Managing Network Services

Administering the UniRPC on UNIX Systems	19-3
How the UniRPC Works	19-3
System Requirements	19-3
Defining the UniRPC Port Number and Maintaining the hosts File. .	19-4
Starting and Stopping the UniRPC Daemon	19-7
About the unirpcservices File	19-8
Managing Windows Telnet Sessions.	19-10
Modifying the Telnet Session Parameters.	19-11
Administering Users.	19-14
Adding a New User	19-15
Starting Services on Windows Platforms	19-18

Beta Beta

This chapter describes the following:

- On UNIX systems, how to administer the remote procedure call utility (UniRPC)
- On Windows platforms, how to administer telnet sessions

Administering the UniRPC on UNIX Systems

The UniRPC lets UniVerse communicate with remote systems. The communicating systems must use TCP/IP networking software to make connections.

On UNIX systems the UniRPC comprises the following UniVerse components:

- The UniRPC daemon *unirpcd*. It receives requests from remote machines for services and starts those services.
- UniVerse BASIC programs for administering the UniRPC.

***Note:** In this chapter the terms local and remote refer to client and server programs or systems. However, because client programs can connect to server programs running on the same computer, remote does not necessarily imply that the server is on another physical computer system.*

How the UniRPC Works

When a client program requests a service on a server, the UniRPC daemon on the server checks the *unirpcservices* file to verify that the client system can request the service. If the UniRPC daemon finds the client system in the *unirpcservices* file, it executes the service requested by the client. Each client process connects to its own server process. Each server process uses the same amount of system resources as a local UniVerse user.

System Requirements

Before installing layered or third-party products that use the UniRPC, such as UV/Net, UniAdmin, or the Uni Call Interface (UCI), you must install and configure TCP/IP using the instructions supplied by the TCP/IP facility vendor. You should then identify the systems to be networked with UniVerse by defining them in the */etc/hosts* file. See [“Maintaining the hosts File”](#) on page 6 for information about how to do this.

You must also modify the configurable UniVerse parameter MFILES. MFILES specifies the size of the UniVerse rotating file pool, which is normally at least eight less than the kernel’s limit for open files per process. You should decrease the value of MFILES by one:



- For each host system you want to connect to through UV/Net
- For each UniVerse server you want to connect to via the BASIC SQL Client Interface (BCI)

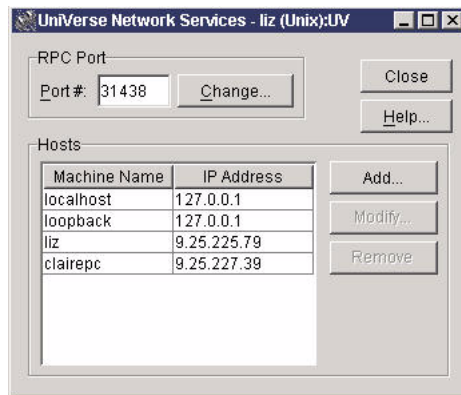
For information about configurable UniVerse parameters, see Chapter 4, [“Configurable UniVerse Parameters.”](#)

Defining the UniRPC Port Number and Maintaining the *hosts* File

Choose **Network Services**, then **UniRPC** from UniAdmin to:

- Define the UniRPC port number
- Maintain the *hosts* file on a UNIX server

The **UniVerse Network Services** window appears, as shown in the following example:



This window has the following components:

- **Port #** field. The current port number for the UniRPC daemon.
- **Hosts** list. Displays the machine name and IP address for each node in the */etc/hosts* file.



***Note:** If you are using the Network Information Services (NIS, also known as Yellow Pages), you do not need to use the `/etc/hosts` file to define, change, and delete network nodes. See the UNIX networking documentation provided with your system for more information.*

Defining the UniRPC Port Number

Before you can use the UniRPC, you must specify the number of the port that the UniRPC is to use. You specify the port number on the client and the server systems.



***Note:** If you specify a port number other than the default, it must be the same on all systems that communicate via the UniRPC.*

The current UniRPC daemon port number is displayed in the **Port #** field in the Network Services window. To change the number, do the following:

1. Click **Change**. The **Change Port Number** dialog box appears. Enter a new number in the **Enter new Port number** field.
2. Click **OK**. The new port number is saved and the Network Services window is updated with the new setting.



***Note:** To use the new port number, you must restart the UniRPC daemon (see [“Starting the UniRPC Daemon”](#) on page 8).*

Maintaining the hosts File

Use the **Network Services** option of UniAdmin to add, modify, and remove nodes in the *hosts* file. These tasks are performed from the **UniVerse Network Services** dialog box.

Adding a Node

Complete the following steps to add a new node to the *hosts* file:

1. Click **Add** from the **UniVerse Network Services** dialog box. The **Add Node** dialog box appears.
2. Enter the node name in the **Machine Name** field.
3. Enter the node address in the **IP Address** field.

4. Click **OK**. The new node's machine name and IP address are checked against existing entries in the *hosts* file. If the new node matches an existing entry, a message box appears. You must acknowledge the message before you can enter alternative values. If the new node details are unique, the new node definition is added to the *hosts* file and the Network Services window is updated.

Modifying a Node

To modify the name or IP address of an existing entry in the *hosts* file:

1. Choose the node to modify by doing one of the following:
 - Double-click the node in the **Hosts** list.
 - Choose the node and click **Modify**.
The Modify Node dialog box appears.
2. Edit the entries in the **Machine Name** and **IP Address** fields.
3. Click **OK**. The node's machine name and IP address are checked against existing entries in the *hosts* file. If the node details match an existing entry, a message box appears. You must acknowledge the message before you can enter alternative values. If the node details are unique, the node definition is added to the *hosts* file and the **UniVerse Network Services** dialog box is updated.

Removing a Node

To remove a node definition from the *hosts* file:

1. Select the node from the Hosts list.
2. Click **Remove**. A message box appears.
3. Click **Yes**. The node definition is removed from the *hosts* file and the **UniVerse Network Services** dialog box is updated.

Starting and Stopping the UniRPC Daemon

You cannot use Unito to start or stop the UniRPC daemon because it uses the UniRPC daemon to connect to the UniVerse server. You must use the UniVerse System Administration menus on the UniVerse server itself to start and stop the UniRPC daemon.

Starting the UniRPC Daemon

Complete the following steps to start the UniRPC daemon:

1. Choose **Rpc administration** from the Package menu, then choose **Start the rpc daemon**.
2. At the prompt, do one of the following to handle any error messages:
 - Enter the name of the file to send all error and system messages to.
 - Enter a space to display messages on your screen.
 - Press **ENTER** if you do not want to display or save messages.
3. At the next prompt, click **Yes** to start the UniRPC daemon or **No** to return to the Rpc administration menu.

***Note:** The file that receives all error and system messages can grow unchecked unless you monitor it periodically.*

Once you start the UniRPC daemon, it automatically restarts whenever you boot UniVerse.

Stopping the UniRPC Daemon

To stop the UniRPC daemon:

1. Choose **Rpc administration** from the Package menu, then choose **Halt the rpc daemon**.
2. At the prompt, click **Yes** to stop the UniRPC daemon or **No** to return to the Rpc administration menu.

***Note:** Stopping the UniRPC daemon does not interrupt active UniRPC processes.*

About the `unirpcservices` File

Each process that uses the UniRPC automatically configures the `unirpcservices` file when it first starts up. If no `unirpcservices` file exists, it is created.

- On UNIX systems the default location of this file is `/usr/ibm/unishared/unirpc`.
- On Windows platforms the default location is `<drive>:\IBM\unishared\unirpc`.



When a client system requests a connection to a service on a server system, the UniRPC daemon (*unirpcd*) on the server uses the *unirpcservices* file to verify that the client system can start the requested service.

The UniRPC software uses field 3 of the *unirpcservices* file to verify that a machine making a request for a service is allowed to do so. The following table lists the fields in the *unirpcservices* file:

Attribute	Description
1	The name of the UniRPC service (for example, <i>uvserver</i>).
2	The full path of the service engine executed by the UniRPC daemon.
3	The names of nodes allowed to execute this service. This field is multi-valued, with values separated by commas (no spaces). If the field contains * (asterisk), all hosts defined in <i>/etc/hosts</i> can execute this service.
4	The network transport mechanism for the service (TCP/IP).
5	Reserved for future use.
6	The value (in tenths of a second) specifying how long an open connection can be idle before automatic closure from the remote connection. The default is 3600, or 6 minutes.

unirpcservices Attributes20

A *unirpcservices* file might contain entries such as the following:

```
uvnet /usr/ibm/uv/bin/uvnetd host1,host2,host3 TCP/IP 3 3600
uvdrsrv /usr/ibm/uv/bin/uvdrsrvd * TCP/IP 0 3600
uvcs /usr/ibm/uv/bin/uvapi_server * TCP/IP 0 3600
uvfilefix /usr/ibm/uv/bin/uvfilefix_server * TCP/IP 0 3600
uvserver /usr/ibm/uv/bin/uvsrvd * TCP/IP 0 3600
```

The version of *uv.rc* shipped with your system (*/usr/ibm/uv/sample/uv.rc*) contains commands that:

- Check for the existence of the *unirpcservices* file
- Verify that services are defined in it
- Start the UniRPC daemon if the file contains services.

The UniRPC daemon is executed as part of the reboot procedure.

Managing Windows Telnet Sessions

To manage telnet sessions on a Windows server from UniAdmin, double-click **Network Services**, then double-click **Telnet**. The **UniVerse Network Services** dialog box appears, as shown in the following example:

UniVerse Network Services - localhost (NT):UV

Parameters | Users

Telnet Port
Port #: 23

SSL Port
Port #: 992

Start

Stop

Pause

Resume

User Policy

☐ Home Account

☐ Any Directory

☒ Home Directory

☐ UV Account

☐ Any Account

☐ UV Directory

Connection Parameters

Max. Logon Attempts: 4

Logon Pause: 4

Logon Timeout: 30

Termination Pause: 4

Keep Alive Parameters

Keep Alive: ☒

Keep Alive Interval: 1000

Keep Alive Time: 7200000

Max. Data Retransmissions: 5

Miscellaneous

Backlog Queue: 14

Detach Process ☒

Create Desktop ☐

Logon Banner: Welcome to the IBM UniVerse Telnet Server.

SSL Logon Banner: Welcome to the IBM UniVerse Secure Telnet Server.

Close Save Help...

The **Network Services** dialog box contains the following fields and options:

- **Telnet Port #** – This field displays the TCP port that the telnet session uses. This is taken from the *services* file. If a *uvtelnet* entry exists in the *services* file, this is the number UniAdmin displays. If these entries do not exist in the *services* file, UniAdmin displays the default port number, 23.

- **User Policy** – The **User Policy** setting determines how the telnet session is used when a user makes a telnet connection.
- **Connection Parameters** – **Connection Parameters** are the current connection values for the telnet service. UniVerse stores these parameters in the Windows Registry on the Server.
- **Keep Alive Parameters** – **The Keep Alive** parameters determine intervals when UniVerse checks the viability of a network connection between the client and server.

Modifying the Telnet Session Parameters

You can modify any of the telnet session parameters from the **Network Services** dialog box.



***Note:** To use the new settings, you must stop and restart the uvtelnet service.*

Changing the Telnet Session Port Number

To change the port number for the telnet session, enter the new port number in the **Port #** box. UniVerse stores the new port number as a *uvtelnet* entry in the *services* file when you click **Save**.

Defining the User Policy

As a UniVerse administrator, you can specify how all users use the telnet session. Valid user policies are:

- **Home Account** – On connection, users attach to their home directory. The home directory must be a valid UniVerse account.
- **Home Directory** – This is the default setting. Users connect to their home directory, but if the home directory is not a UniVerse account, UniVerse prompts users to set up the account.
- **Any Account** – Users can connect to any valid UniVerse account.
- **Any Directory** – Users can connect to any directory, but if the directory is not a UniVerse account, UniVerse prompts to set up the account.
- **UV Account** – Specifies that the user connects to an existing UniVerse account defined in the UV.LOGINS file.



- **UV Directory** – Specifies that the user connects to a directory defined in the UV.LOGINS file, and can create a UniVerse account in that directory if the directory is not already configured for UniVerse.

***Note:** Administrators are prompted for the account to which they want to connect regardless of the User Policy setting.*

Setting the Telnet Connection Parameters

The four valid telnet connection parameters are:

- **Max. Logon Attempts** – Defines the number of failed log in attempts a user is allowed before the telnet connection is dropped. The default setting is 4.
- **Logon Pause** – If a logon attempt fails, the pause between logon attempts (in seconds). The default setting is 4 seconds.
- **Logon Timeout** – The time (in seconds) the system waits for a response to a logon prompt. As soon as this time limit is reached, the telnet connection is dropped. The default value is 30 seconds.
- **Termination Pause** – The amount of time UniVerse pauses after the final failed logon attempt before dropping the telnet connection. The default value is 4 seconds.

Setting Keep Alive Parameters

The Keep Alive feature determines when inactive connections can be disconnected. When a connection becomes inactive, keep-alive packets are periodically exchanged. When a number of consecutive packets remain unanswered, by default 20, the connection is broken.

- **Keep Alive Interval** – The interval, in milliseconds, separating keep alive retransmissions until a response is received. Once a response is received, the delay until the next keep alive transmission is controlled by the value of **Keep Alive Time**. After the number of retransmissions specified by **Max. Data Retransmissions** are unanswered, the connection aborts. The default value is 1000 (one second).
- **Keep Alive Time** – This parameter specifies how often TCP attempts to verify that an idle connection is still valid by sending a keep alive packet. If the connection is still valid, the remote system will acknowledge the keep alive transmission. The default value is 7,200,000 milliseconds (two hours).

- **Max. Data Retransmissions** – This parameter specifies the number of times TCP retransmits an individual data segment before aborting the connection. The retransmission timeout is doubled with each successive retransmission on a connection. It is reset when responses resume.

Setting Event Logging Level

Use the Event Logging Level options to specify the amount of information you want to log. If you do not select an option, UniVerse does not log events. Valid logging levels are:

- **Error** – Logs all errors messages.
- **Warning** – Logs all warning messages.
- **Information** – Logs both errors and warnings.

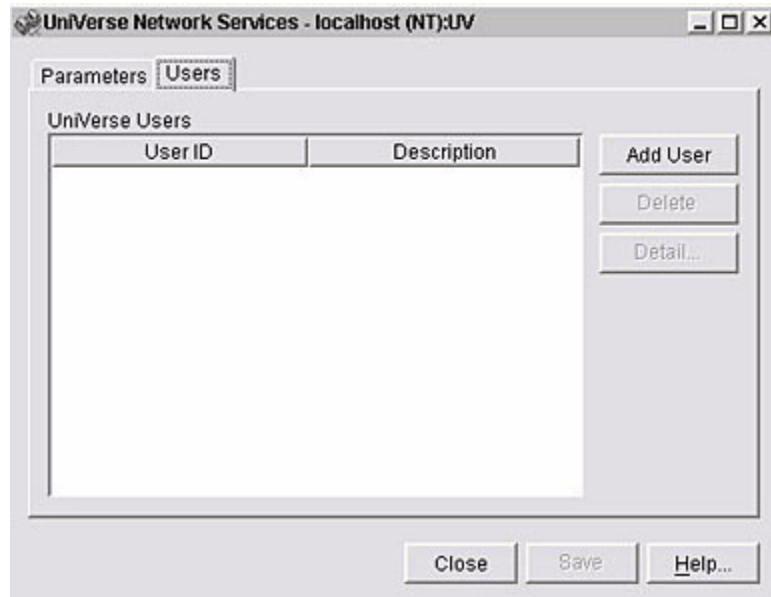
Specify Logon Banner

You can specify the banner that users will see when they telnet to a host in the **Logon Banner** box.

Administering Users

The UV.LOGINS file resides in the UV account. It contains a list of users and the directories or UniVerse accounts they log on to when they first invoke UniVerse from a telnet session.

To maintain entries in the UV.LOGINS file, click the **Users** tab. The **UniVerse Users** dialog box appears, as shown in the following example.

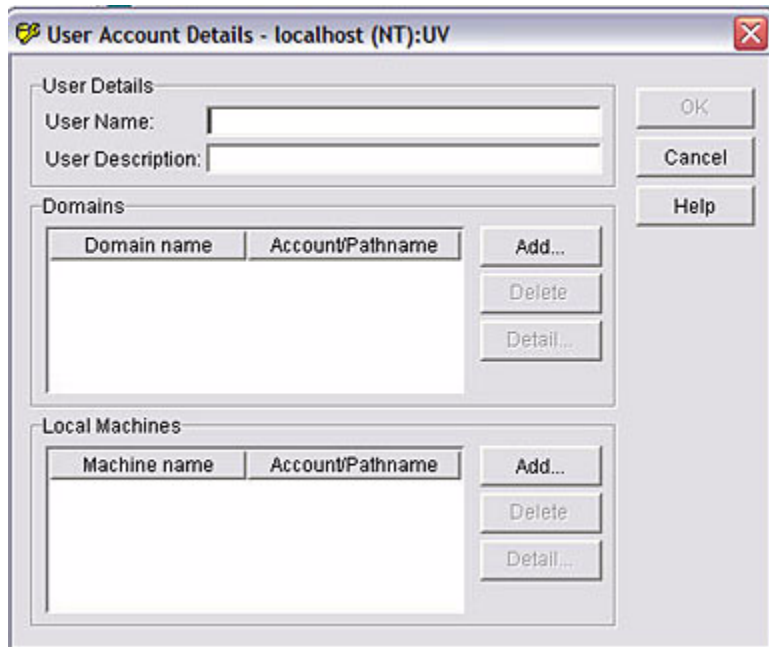


You can enter users logging on to the system both from the local machine and from domains. You can also maintain entries for users who have accounts on multiple domains with access to this system. You can specify the user's account either as a case-sensitive entry in the UV.ACCOUNTS file, or as a fully qualified path.

If the user logs on to the system using a local machine login ID, UniVerse uses the Local Machine entry. If the user logs on to the system via a domain, UniVerse uses the entry for the domain. If the user enters a login ID without a machine or domain name, UniVerse first uses a local machine login ID if it exists, and then checks domain login IDs.

Adding a New User

To add a new user, click **Add User** from the **UniVerse Users** dialog box. A dialog box similar to the following example appears:



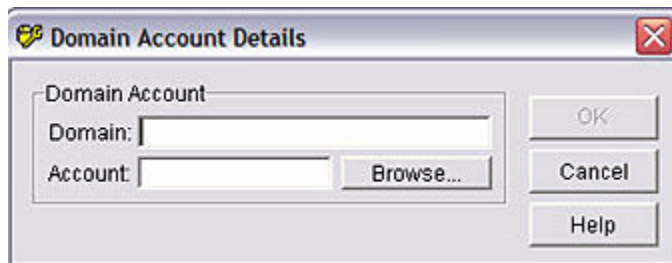
The **User Account Details - localhost (NT):UV** dialog box is used for adding or managing users. It features a title bar with a yellow key icon and a red close button. The main area is divided into three sections: **User Details**, **Domains**, and **Local Machines**. The **User Details** section contains text boxes for **User Name:** and **User Description:**. The **Domains** section has a table with columns **Domain name** and **Account/Pathname**, and buttons for **Add...**, **Delete**, and **Detail...**. The **Local Machines** section has a similar table and buttons. On the right side, there are three buttons: **OK**, **Cancel**, and **Help**.

Domain name	Account/Pathname
-------------	------------------

Machine name	Account/Pathname
--------------	------------------

Add a Domain User

To add a domain user, in the **Domain** area, click **Add**. The **Domain Account Details** dialog box appears, as shown in the following example:

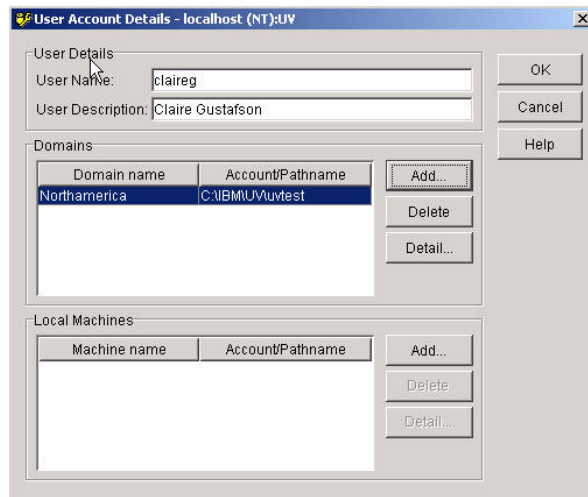


The **Domain Account Details** dialog box is used for adding domain users. It has a title bar with a yellow key icon and a red close button. The main area contains a **Domain Account** section with text boxes for **Domain:** and **Account:**, and a **Browse...** button. On the right side, there are three buttons: **OK**, **Cancel**, and **Help**.

Enter the name of the domain to which you want the user to connect in the **Domain** box.

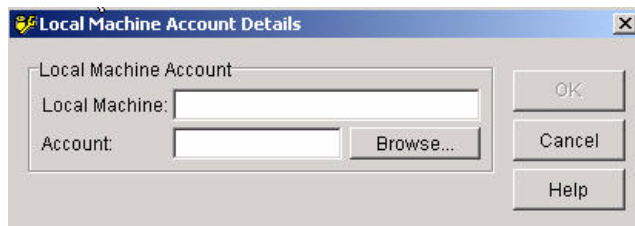
Enter the full path to the account to which the user is to connect, or click **Browse** to search for the account.

Click **OK** to save the information, or click **Cancel** to exit without saving changes. The user appears in the **Domain** area of the **User Account Details** dialog box, as shown in the following example:



Adding a Local Machine User

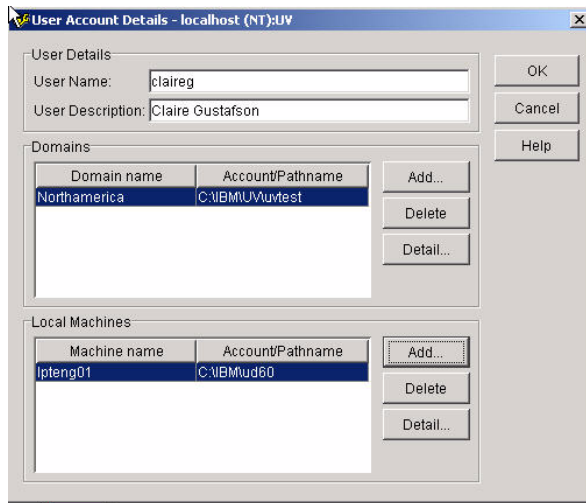
To add a user to a local machine, in the **Local Machines** area of the **User Account Details** dialog box, click **Add**. The **Local Machine Account Details** dialog box appears, as shown in the following example:



Enter the name of the local machine in the **Local Machine** box.

Enter the full path to the account to which the user is to connect, or click **Browse** to search for the account.

Click **OK** to save the information, or click **Cancel** to exit without saving changes. The user appears in the **Local Machines** area of the **User Account Details** dialog box, as shown in the following example:



Starting Services on Windows Platforms

On Windows platforms, you can start and stop the following services from the **UniVerse Services** dialog box:

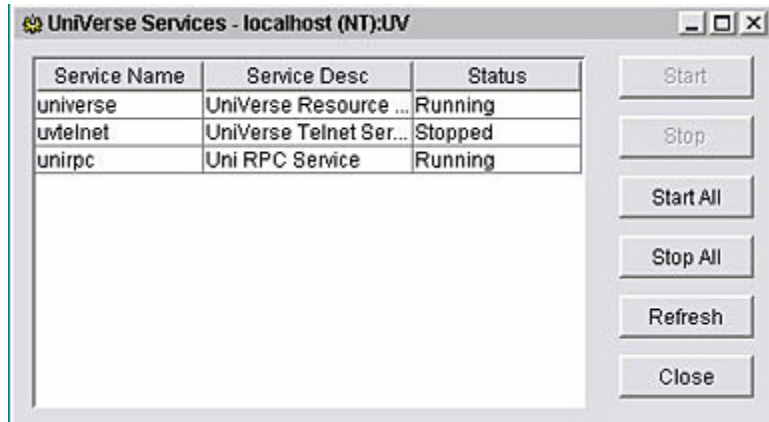
- universe
- uvtelnet
- unirpc
- hsrexec

Select one of the following methods to access the **UniVerse Services** dialog box:

- From the **UniAdmin** window, double-click **Network Services**, then click **Services**.
- From the **UniAdmin** menu, select *Network Services*, then click **Services**.
- From the **UniAdmin** toolbar, click the **Network Services** icon, as shown in the following example.



The **UniVerse Services** dialog box appears, as shown in the following example:



You can perform the following tasks from the **UniVerse Services** dialog box:

- To start a service, click the service you want to start, then click **Start**.
- To stop a service, click the service you want to stop, then click **Stop**.
- If you want to start all services, click **Start All**.
- If you want to stop all services, click **Stop All**.
- If you want to refresh the information displayed in the dialog box, click **Refresh**.

Click **Close** to exit the **UniVerse Services** dialog box.

Device Licensing

UniVerse Licensing Modes	20-4
Why Do I Need Device Licensing?	20-6
Device Licensing Requirements	20-6
Connection Types	20-7
Direct Connections.	20-7
Two-Tier Connections.	20-7
Enabling Telnet Device Licensing on UNIX Servers	20-8
Using Device Subkeys	20-8
Using the License Tool <i>uvlictool</i>	20-9

This chapter describes how to use UniVerse's device licensing system.

UniVerse Licensing Modes

UniVerse provides two licensing modes:

- Session licensing
- Device licensing

Session Licensing

Session licensing is like the UniVerse licensing system used before Release 9.5. Every connection from telnet or a UniVerse API, even from the same PC, consumes one database license. Session licensing has been enhanced to include a new licensing tool, *uvlictool*, that reports on the current licensing state and cleans up current licensing.

Device Licensing

Device licensing, sometimes called client-side licensing, tries to combine all remote connections from a single device to a UniVerse server at both the UniVerse license level and the package level.

Device licensing is available only with the Enterprise Edition of UniVerse. To determine whether your version of UniVerse supports device licensing:

- Use the UniVerse command `CONFIG` with no options
- Use the UNIX shell command *uvlictool* with no options

Device licensing currently works with the following connection types:

- UniAdmin
- UniVerse ODBC
- UCI
- UniObjects
- UniObjects for Java
- InterCall



- Telnet connections such as:
 - SB Client
 - wIntegrate
 - Dynamic Connect

***Note:** Device licensing does not work with UV/Term.*

Why Do I Need Device Licensing?

Users accessing a UniVerse server through one or more client application programs may want to put their licensing scheme on a one-license-per-device basis. Such applications often open multiple connections to a UniVerse server. For example, an application might use one connection to browse, another connection to check data, yet another connection to update the database, and so forth.

Before UniVerse Release 9.5, each connection to the server consumed its own separate license, even though only one user was using all those connections from one PC. UniVerse's device licensing lets such users consume one UniVerse license and one license for up to 10 connections to the server from a single PC.

Device Licensing Requirements

Device licensing has the following requirements:

- Clients must run on a Windows platform.
- Clients must run on a LAN or TCP/IP with an Ethernet card.
- For telnet connections to UNIX servers, device licensing must be enabled.

Connection Types

There are three ways to connect to a UniVerse server:

- Direct connection. This is not a client/server connection.
- Two-tier client/server connection.
- Multiple-tier client/server connection.

Each PC can have up to ten connections to the server, but not all connections from a PC can be combined.

Direct Connections

Direct connections are not really client/server connections because there is no real client. Examples of direct connections are:

- Directly invoking UniVerse on a system
- TTY serial line

Two-Tier Connections

Two-tier connections are typical client/server connections where a client application connects to a UniVerse server either on the same machine or on a different machine. Telnet connections to UniVerse are an example of a two-tier connection.

Client applications running on PCs different from the UniVerse server appear to the server with unique identifiers.

Multiple-Tier Connections

Multiple-tier connections are client applications that connect from a PC to a UniVerse server either through one or more different PCs, or through an application server component. Examples of multiple-tier connections are:

- An HTTP server running scripts that use UniObjects or UniObjects for Java.

- An application that connects first to an application server either on a different PC or on the server system. The application server connects to the UniVerse server.

Enabling Telnet Device Licensing on UNIX Servers

Use the shell command *uvdls* to enable device licensing for telnet connections to a UNIX server. When you telnet to a UNIX server, you can:

- Enter the *uvdls* command instead of the *uv* command to log in to UniVerse
- Configure your UniVerse initialization script (*.profile*, *.cshrc*, and so forth) or your terminal emulator to use *uvdls* instead of *uv* to log in directly to UniVerse
- Some telnet client programs let you enable device licensing from the client. See your telnet client documentation for details.

Using Device Subkeys

Each PC that connects immediately to the UniVerse server can have up to ten connections per license.

Using multiple-tier connections, each PC that connects to an intermediate application component consumes a separate license. But each of these PCs, at one or more removes from the server, can have up to ten connections.

In order for a PC to have multiple connections to the UniVerse server and still consume only one license, users must ensure that each PC connecting to the server through another system specify a unique *device subkey* before requesting a connection to the server. This subkey is a string of up to 24 characters. All client applications on a given device that connect to one UniVerse server must use the same unique subkey.

Using the License Tool *uvlctool*

UniVerse Release 10.2 provides a utility called *uvlctool*. This license tool does the following:

- Lists a report on license use at both the UniVerse and the package level.
- Identifies the process that owns the license, and lists package licenses it holds.
- Identifies the remote device holding the license.
- On UNIX systems, *uvlctool* cleans up the current licenses based on shared memory segments associated with dead processes.
- On Windows platforms, *uvlctool* cleans up the current licenses based on dead entries in the process table.
- Recomputes license counts at the UniVerse, package, and seat levels.

The syntax of *uvlctool* is as follows:

uvlctool [report_lic] [clean_lic [-a]]

Parameter	Description
report_lic	Lists the current licensing state.
clean_lic	Cleans up the current licensing state.
-a	Recomputes license counts at the UniVerse, package, and seat level.

uvlctool Parameters

UniVerse System Administration Menus

On UNIX systems you can use the UniVerse System Administration menus in addition to or instead of the UniAdmin client program to administer UniVerse. These menus let you do normal UniVerse maintenance and some UNIX system administration without having to use the UNIX shell commands or file formats.

The UniVerse System Administration menus are available only to UniVerse Administrators working in the UV account. To have access to all files used for system administration, you must log in as a UniVerse Administrator.

Once you are logged on, activate the System Administration menus from the UNIX shell by changing to the UV account directory (usually */usr/ibm/uv*):

```
# cd /usr/ibm/uv
```

Then use the command *bin/uv*:

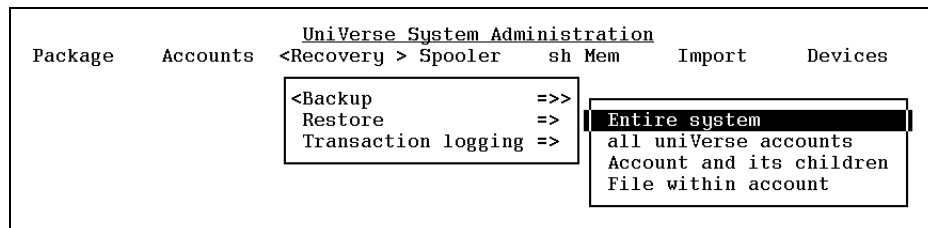
```
# bin/uv
```

From an account in the UniVerse environment, you can use the LOGTO command to log to the UV account.

Overview of Menus and Data Entry Screens

The System Administration menus and data entry screens look and work the way Motif menus do. The main menu bar appears at the top of the screen. Each option in the menu bar stands for a pull-down menu that scrolls down from the main menu bar when the user chooses an option.

Because the menu bar is the primary way to access the system administration functions, browsing through the menus gives a good overview of system administration functions.



The System Administration Menus

Menus list actions that can be performed and options that display submenus (*cascading* menus). Options followed by an arrow (=>) display a submenu. To display a submenu, highlight the option, then choose it.

To start an action, simply choose it. If a data entry screen appears, instructions in the lower part of the screen prompt you to enter the appropriate data. System messages also appear at the lower part of the screen.

The options listed on the submenus are names or descriptions of UniVerse sentences that are stored as part of the menu. Unlike using sentences that are stored in the VOC file, you need not remember sentence names to use an option on a menu. You need only examine the options listed on the menu, and then choose the appropriate one for the task you want to perform.

Moving Around the Menus

Certain responses work at all levels of the menu system. At any menu bar, use the **Right Arrow** key (→) and the **Left Arrow** key (←) to move the cursor to the option that you want.

Note that → does one of two things. If the highlighted menu option calls a submenu, pressing → displays the submenu. If the option does not call a submenu, pressing → displays the submenu belonging to the next option on the main menu bar.

Choosing an Option

To choose the highlighted option, press **ENTER** or **Space**. Or to move to an option and choose it with one keystroke, press the capitalized letter of the option you want. (The capitalized letter is called a *mnemonic*.) The mnemonic may not always be the first letter of an option. If your keyboard does not have arrow keys, you can move around the menu system by using the mnemonic letters to make selections.

Moving Around the Submenus

To move the cursor up and down in the submenus, use ↑ and ↓. Note that ↑ does not work on menu bars, and ↓ works only if there is a submenu. To move from a submenu to the one immediately above it, press ←. To return directly to the main menu level from any submenu, press **F10**.

To exit any System Administration menu or submenu and return to the UniVerse prompt, press **Esc**.

Summary of Standard Keys

The following table lists standard responses that you can use at all data entry screens.

Key	Action
Esc	<p>Pressing the Esc key has two effects. At any menu or submenu, Esc returns you to the UniVerse prompt.</p> <p>If you are entering data in a data entry screen, pressing Esc quits the record, clears the screen of any data you entered without making any changes, and returns to the previous level. If you were adding a new record, it is not added. If you were changing existing data, the record remains as it was before you made any changes to it.</p>
␣ ␣ ␣	Moves the cursor to the next or previous field on the screen.
F10	Always moves the cursor to the menu bar at the top of the screen, if one exists. (If your terminal does not have an F10 key, you can activate the menu bar by pressing Ctrl-T .)
F1	Displays a more detailed help message. (If your terminal does not have an F1 key, you can display the longer help message by typing a question mark (?).
ENTER	If the cursor is highlighting an option in the menu bar or some other option (such as YES or NO), pressing ENTER selects the highlighted option (same as pressing Space).
Space	Same as ENTER .
F4	<p>Pressing F4 at certain data entry prompts displays a list and lets you select one of the items in the list. If your terminal does not have an F4 key, you can display a list by typing an asterisk (*).</p> <p>In most cases the list box displays a sorted list with an entry box below it. To move directly to an item in the sorted list, enter the item you want at the prompt in the entry box. The highlight bar moves to the item you enter. You can enter any number of characters in the entry box to move to a desired item in the sorted list.</p>

Summary of Standard Keys



Key	Action
	For example, if you enter the character j , the highlight bar moves to the first item in the list that begins with j. If you enter the characters sta , the highlight bar moves to the first item in the list that begins with sta.
Page Down	When a list appears, use the Page Down key (also called the Page key or the Pg Dn key) to display the next page of the list.
Page Up	When a list appears, use the Page Up key (also called the Page key or the Pg Up key) to display the previous page of the list.

Summary of Standard Keys (Continued)

***Note:** Pressing some of these keys in rapid succession can create the effect that the keys are not working.*

To improve performance, the program MTF.INPUT.B in the BP directory of the UV account has a variable, ESC.DELAY.TIME, to fine tune the delay. An example follows:

```
>ED BP MTF.INPUT.B
----: 8
0008: * Module MTF.INPUT.B Version 3.3.1.1 Date 4/16/96
----: 95
0095: equ ESC.DELAY.TIME to 100;* Delay to determine if singleton
Escape
```

You should also make changes to the BASIC program CINPUT.B in the APP.PROGS directory. The third argument in the call to GET.TA.BUF is the delay (25 in the following example).

```
>ED APP.PROGS CINPUT.B
----: 6
0006: * Module CINPUT.B Version 3.2.1.1 Date 2/18/96
----: 153
0153: CALL *GET.TA.BUF.B(0,LEN(MO.KEYS<I,1>)-
1,25,100,INPUT.CHARACTER
```

The UniVerse System Administration Menu

The administrative functions of the UniVerse System Administration menu system are performed from the UV account. The functions are implemented as a hierarchy of menus and data entry screens. Some menu selections invoke submenus, other menu selections invoke BASIC routines, UniVerse commands, and UNIX shell command scripts.

The following pages describe the options of the main menu bar and all its submenus.

Invoking the System Administration Menu

The main menu is called the UniVerse System Administration menu. Its record name is **SYSTEM.ADMIN**, which is located in the file **UNIVERSE.MENU.FILE** in the UV account.

The UniVerse System Administration menu is automatically invoked by the UV account's **LOGIN** entry when you enter the UniVerse environment from the UV account directory (by using the command *bin/uv*).

You can also invoke the menu in two other ways:

- Enter **LOGIN** at the UniVerse prompt while in the UV account.
- Enter the command **SYSTEM.ADMIN** at the UniVerse prompt.

The following example shows the UniVerse System Administration menu.

UniVerse System Administration						
Package	Accounts	Recovery	Spooler	sh Mem	Import	Devices
<Space> or mnemonics to select, <F1> for help, arrows, <esc> to exit, or <F10>						

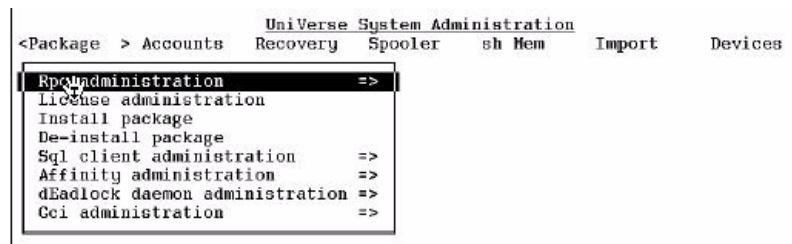
The UniVerse System Administration Menu

Package Option

Use the **Package** option to do the following:

- Install and deinstall additional software packages
- Administer the UniRPC (remote procedure call)
- Administer UniVerse licenses
- Administer the deadlock daemon
- Administer the BASIC SQL Client interface (BCI)
- Administer the General Calling Interface (GCI)

When you choose **Package** from the UniVerse System Administration menu, the Package menu appears as shown in the following example.



Installing and Deinstalling a Software Package

To install a software package, choose **Install package** from the Package menu, then enter the name of the software package and the name of the device from which you are installing it.

To deinstall a software package, choose **De-install package** from the Package menu, then enter the name of the software package. To see a list of all currently installed software packages, press **F4** or enter an asterisk (*) at the prompt.

For information about specific software packages, refer to the documentation provided with the package you want to install.

To display the license number, the current user limit, and the current license expiration date, exit the menus and, at the command prompt, use the **CONFIG** command with no options.

Administering the UniRPC

The Rpc administration menu lets you do the following:

- Define or change the UniRPC port number
- Start and stop the UniRPC daemon
- Add, change, or delete network nodes

Defining the UniRPC Port Number

To define the UniRPC port number:

1. Choose **Rpc administration** from the Package menu, then choose **Change the rpc port**. The following message appears:
2. The well-known port number defined for unirpc is '31438'.
Do you wish to continue using this well-known port number for the unirpc well-known port?
3. At the prompt, choose **Yes** to accept the default UniRPC port number. Choose **No** to specify another port number.
4. If you choose **No**, the system prompts you to enter a new UniRPC port number. When this number is accepted, the following message appears:
5. `/etc/services` successfully written.

Starting the UniRPC Daemon

To start the UniRPC daemon:

1. Choose **Rpc administration** from the Package menu, then choose **Start the rpc daemon**.
2. At the prompt, enter the name of the file to send all error and system messages to (this can be useful for debugging). Enter a space to display messages on your screen. Press **ENTER** if you don't want to display or save messages.
3. At the next prompt, choose **Yes** to start the UniRPC daemon. Choose **No** to return to the Rpc administration menu.

Adding, Changing, or Deleting Network Nodes

Use the Maintain the RPC screen to add nodes to the network, modify the definitions for nodes on the network, and delete nodes from the network. You can also list the nodes that are defined in the */etc/hosts* file.

To display the Maintain the RPC screen, choose **Rpc administration** from the System Administration menu, then choose **Modify /etc/hosts**. The following example shows the Maintain the RPC screen.

File		Maintain the RPC	Action	Help
Machine Name	?			
Node ID	:			
<div>Help Region</div> <div>Enter a name for this machine. Press F4 for a list of currently defined machines.</div>				

The Maintain the RPC Screen

You can add, modify, or delete nodes by entering information in this screen. This changes the TCP/IP hosts file (*/etc/hosts*). The Maintain the RPC screen prompts you for the following information:

Machine Name

Enter the node name for a machine (node names are case-sensitive).

Node ID

Enter the TCP/IP Internet address for a machine.

Adding Nodes to the Network

Use the Maintain the RPC screen to add machines to the TCP/IP hosts file (*/etc/hosts*). Enter the machine name and node ID in the fields provided.

If you enter the name of a machine already defined in */etc/hosts*, the screen displays the name of the machine and its node ID. If you enter a node ID for an existing machine, the screen displays the following message:

A record with that Node id already exists. Is this a Synonym?

To define the new machine's name as a synonym for the existing machine, choose **Yes**, otherwise choose **No**.

Changing an Entry in the TCP/IP Hosts File

Use the Maintain the RPC screen to change a node's name or ID. For example, to change the machine name, specify the new name with the old node ID. To change the node ID, specify the new node ID with the old machine name.

To display the machines currently on the network, enter an asterisk (*****) at the Machine Name prompt, or press **F4**. Use the arrow keys to move through the list, highlighting the entry you want to modify. To choose the highlighted entry, press **ENTER**.

Deleting Nodes from the Network

Use the Maintain the RPC screen to delete a node definition from the TCP/IP hosts file. You can either enter the machine name of the node you want to delete, or you can enter an asterisk (*****) or press **F4** to display a list of names in the TCP/IP hosts file. Use the arrow keys to move through the list and to highlight the entry you want to delete. To choose the highlighted entry, press **ENTER**.

Press **F10** to move to the menu bar, then choose **Action**, then choose **Delete entry** to delete the node from the network. The system asks you to confirm the deletion. Choose **Yes** to delete the node from */etc/hosts*. Choose **No** to cancel the deletion. You can then specify a different node to delete, or you can return to the Rpc administration menu.

Listing Nodes on the Network

To list nodes on the network, press **F10** to move to the menu bar of the Maintain the RPC screen, then choose **Action**, then choose **List entries**.

Changing the UniRPC Port Number

If you change the UniRPC port number, you must change it on all other systems that communicate via the UniRPC. Do the following:

1. Choose **Rpc administration** from the Package menu, then choose **Change the rpc port**. The current UniRPC port number is displayed.
2. At the prompt, choose **Yes** to accept the current UniRPC port number. Choose **No** to change the port number.
3. If you choose **No**, the system prompts you to enter a new UniRPC port number. When this number is accepted, the following message appears:

```
/etc/services successfully written.
```

Stopping the UniRPC Daemon

1. From the main System Administration menu, choose **Package**, then choose **Rpc administration**, then choose **Halt the rpc daemon**.
2. At the prompt, choose **Yes** to stop the UniRPC daemon. Choose **No** to return to the Rpc administration menu.

Stopping the UniRPC does not interrupt active UniRPC processes.

***Note:** On some systems, after you stop the UniRPC daemon and all active services terminate, it can take five minutes for UNIX or TCP/IP to recognize that the network connections used by the UniRPC are available again. Wait at least five minutes after halting the UniRPC daemon before you restart it. Use the UNIX netstat command to see what services are still active.*



UniVerse License Administration

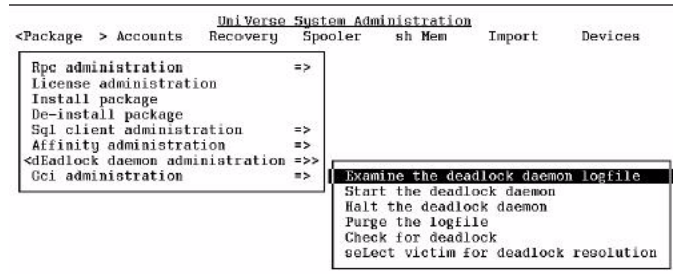
The Package menu also includes the UniVerse **License administration** option. Use this option to authorize a licensed number of users on your system.

When you choose **License administration** from the Package menu, UniVerse displays the Upgrade UniVerse License data entry screen. To upgrade your license, enter the authorization code obtained from your vendor.

Administering the Deadlock Daemon

You can use the System Administration menus or the *uvdlockd* command to do the following:

- Start and stop the deadlock daemon
 - Set the restart time, resolution strategy, and deadlock log file location
 - Examine the deadlock log files
 - Delete deadlock log files
 - Display the status of the deadlock daemon
1. To display the Deadlock Daemon Administration menu, choose **Package** from the System Administration menu, then choose **dEadlock daemon administration**.



2. Choose the administrative task you want to perform from the following options:

Examine the deadlock daemon logfile. This option displays the contents of the deadlock log file. This log file records the date and time the deadlock daemon is started up and shut down, any detected deadlocks, and the resolutions applied to them.

Start the deadlock daemon. This option starts the deadlock daemon (*uvdlockd*) if it is not already running. The following data entry screen appears:

Deadlock Daemon Administration		
File	Action	Help
Enter interval timer	? 60	
Enter the resolution strategy	: 0	
Enter the location of the log file	: /u1/uv/uvdlockd.log	
Should the daemon be started at boot time?	: Y	
Should this information be stored in the configuration file?	: Y	
<div>Help Region</div> Press <F1> for longer help about any particular entry, or <ESCAPE> to exit program. Enter the time interval, in seconds, that the daemon should wait before rechecking for deadlock conditions		

Enter the interval time in seconds. For the resolution strategy, enter **0** (random), **1** (newest), or **2** (fewest locks). Press **Return** to accept the default log file location, or enter another pathname. Enter **Y** if you want the deadlock daemon to start up each time you restart UniVerse, otherwise enter **N**. Enter **Y** to store the current setting in the deadlock daemon configuration file (*uvd-lockd.config*), otherwise enter **N**.

Halt the deadlock daemon. This option shuts down the deadlock daemon.

Purge the logfile. This option clears the deadlock log file.

Check for deadlock. This option generates a report based on a one-shot analysis of the lock-waiter tables and any detected deadlocks.

seLect victim for deadlock resolution. This option lets you select the user number of a process to abort, thus resolving the deadlock.

Resolving Deadlocks Manually

You can resolve deadlocks manually in two ways:

- Use the **seLect victim for deadlock resolution** option from the Deadlock Daemon Administration menu
- Use the `uvdlockd` command

Complete the following steps to resolve a deadlock manually:

1. Analyze the lock-waiter tables to see if there are any current deadlocks. Choose the **Check for deadlock** option from the Deadlock Daemon Administration menu.
2. If there is a deadlock, determine which deadlocked user process you want to abort.
3. Abort the user process by choosing **seLect victim for deadlock resolution** from the Deadlock Daemon Administration menu.

Administering SQL Client and GCI

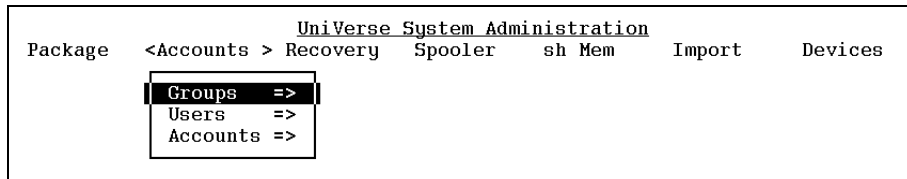
For information about administering the BASIC SQL Client Interface (BCI) and the General Calling Interface (GCI), see *UniVerse BASIC SQL Client Interface Guide* and *UniVerse GCI Guide*.

Accounts Option

Use the **Accounts** option to create and maintain user login accounts, UniVerse accounts, and user groups. The **Accounts** option updates and maintains the following files:

- */etc/group*
- */etc/passwd*
- UV.ACCOUNT

When you choose **Accounts** from the UniVerse System Administration menu, the Accounts menu appears, as shown in the following example.



The Accounts Menu

The following sections tell how to use the System Administration menus to add, change, and delete user login accounts, user groups, and UniVerse accounts.

Using the Accounts Menu

To display the Accounts menu, choose **Accounts** from the UniVerse System Administration menu. You can then choose to add, change, or delete user groups, user login accounts, or UniVerse accounts.

The data entry screens for adding a new user group, adding a user, and adding a new UniVerse account include a menu bar at the top of the screen that provides the following three options:

File Action Help

The **File** option lets you exit the data entry screen. The **Action** option lets you modify or delete another group, user, or account. The **Help** option provides further information on each of these menu bar options.

You can also list all groups, users, or accounts currently defined in the */etc/group*, */etc/passwd*, or *UV.ACCOUNT* files. Display a list either by choosing **List**, or by pressing **F4** or ***** at certain prompts.

Inside a list, use the **Page Down** key to display the next page of the list. Use the **Page Up** key to display the previous page. To choose an item in the list, use the **Down** and **Up Arrow** keys to move the highlight bar to the item you want then press **Return** or **Space**. Or, at the prompt in the entry box just below the list box, enter the item you want (or the first unique characters of it) to move the highlight bar directly to that item.

Some fields on the System Administration menus have a limited amount of space, such as, you can only enter 44 characters for the user's home directory. If you need to specify a longer path, edit the */etc/passwd* file.

Maintaining Users and User Groups

You can do any of the following tasks to maintain user groups and user login accounts:

- Add a new user group or user login account.
- Change information about an existing user group or user login account.
- Delete a user group or user login account from the system.

You use UNIX commands to perform these tasks. In UniVerse you can register SQL users and create SQL schemas with the **GRANT** and **CREATE SCHEMA** statements.

Adding a New User Group

Complete the following steps to add a new user group:

1. Choose **Add a group** from the Accounts menu.
2. Enter the name of a new user group. At the **Group ID** prompt, enter the group ID number, or if you want to assign the next default group ID number, press **ENTER** at the prompt.
3. Choose **Yes** to add the new group. UniVerse updates the */etc/group* file and clears the data entry screen. You can now enter another group name.

When you are finished adding user groups, press **Esc** to return to the System Administration menu.

Changing a User Group

To change the name of an existing user group:

1. Choose **Modify a group** from the Accounts menu. A list of all groups currently defined in the */etc/group* file appears.
2. Choose the group whose name you want to change.
3. Enter the new name of the group, then choose **Yes** to save the new information.

When you save the revised user group information, UniVerse updates the */etc/group* file. You are automatically returned to the System Administration menu.

Only the name of a group can be changed using the **Modify a group** option. The group ID number cannot be changed. To change a group ID number while retaining the group's name:

1. Delete the existing group.
2. Add a new group, specifying the old group name and the new group ID number.

Deleting a User Group

To delete an existing user group:

1. Choose **Delete a group** from the Accounts menu. A list of all groups currently defined in the */etc/group* file appears.
2. Choose the name of the group you want to delete.
3. Choose **Yes** to delete the group.

When you are finished deleting groups, UniVerse updates the */etc/group* file. You are returned to the System Administration menu.

Adding, Changing, and Deleting Individual Users

You can use the UniVerse System Administration menus to add individual users, change information about existing users, and delete users from the system. You can create new accounts with the SQL CREATE SCHEMA statement.

When you add a new user, change information about a user, or delete a user from the system, UniVerse does the following:

- Edits the `/etc/passwd` file
- Adds, changes, or deletes the user's home directory
- Sets or changes file access permissions, user ownership, and group ownership for the files and subdirectories contained in the user's home directory

***Note:** Your system may require users to be added using UNIX shell commands. See the UniVerse release notes for your system before you try to change the `/etc/passwd` file.*



Adding a New User

To add a new user to the system:

1. Choose **Accounts** from the System Administration menu, then choose **Users**, then choose **Add a User**. UniVerse displays the **Add a User** screen.
2. Enter the required information at the prompts.
3. With your entries specified, choose **Yes** to add a new user to the system. UniVerse adds a line containing the specified user information to the `/etc/passwd` file.
4. You are prompted to set file permissions for the user's home directory and all its dependent files and directories.

You can either set the default permissions, which give all users permission to read, write, or execute all files and directories, or you can specify a more limited set of permissions. If the directory specified as the user's home directory does not yet exist on the system, UniVerse creates it.

The procedure for setting file permissions is described later in [“Adding a New UniVerse Account”](#) on page 22.

What happens next depends on whether you specified UniVerse or a UNIX shell as the user's environment:

- If you specified UNIX for `Login Shell`, a UniVerse account is not created for this user (you are not prompted to enter any information pertaining to a UniVerse account). Proceed to the next prompt (go to step 2).
 - If you specified UniVerse for `Login Shell`, you are prompted to enter information about the UniVerse account. Proceed to the next prompt (go to step 5).
1. Perform this step only if you specified UniVerse for `Login Shell`. You must create a UniVerse account for the user's home directory:
 - You are prompted to specify which compatibility flavor the UniVerse account is to have. UniVerse accounts can be one of several standard flavors: `INFORMATION`, `PICK`, `REALITY`, `IN2`, `PIOPEN`, or `IDEAL` UniVerse. `IDEAL` is the default.
 - Choose the compatibility flavor for the account you are creating.
 - The UniVerse account is put in the user's home directory, that is, the special UniVerse files are copied into the directory. This can take a few minutes.
 - You are asked if you want to add the name of the UniVerse account to the Accounts file, `UV.ACCOUNT`. If you choose **Yes**, you are prompted to enter the name of the UniVerse account. (Normally, UniVerse account names are all uppercase letters.)
 2. The system asks if you want to use the default login file. The login file is a file used by the system to initialize the user's working environment when the user logs in. Login files differ depending on which login shell the user will use.

If the user's login shell is the Bourne shell, the login file will be the `.profile` file. If the user's login shell is UniVerse, the login file will be the `LOGIN` entry. The default `.profile` and `LOGIN` files are in the directory `/usr/ibm/uv/sample`.
 3. Once you have entered all the relevant user information, the cursor returns to the `Logname` prompt at the top of the screen. You can now add another user to the system. To return to the System Administration menu, press **Esc**.

Note: Your system may require new users to be added through UNIX. Please see the UniVerse release notes for your system before trying to change the `/etc/passwd` file.



Changing User Information

To modify an existing user definition:

1. Choose **Accounts** from the System Administration menu, then choose **Users**, then choose **Modify a User**. UniVerse lists all users currently defined in the */etc/passwd* file.
2. From the list, choose the name of the user whose data you want to modify. UniVerse displays information about that user.
3. To change the information in any field, use the down and up arrows to move the cursor to the field you want, then enter the new data at the prompt. Continue in the same way to make as many changes as you want.
4. When the data is as you want it, choose **Yes** to save your changes. UniVerse updates the */etc/passwd* file and makes any required changes to the files in the user's home directory such as any ownership, group ownership, or file permission changes.
5. You are asked if you want to modify another user. Choose **Yes** to make more changes, or choose **No** to return to the System Administration menu.

Deleting a User

To delete a user from the system:

1. Choose **Accounts** from the System Administration menu, then choose **Users**, then choose **Delete a User**. UniVerse displays a list of all users currently defined in the */etc/passwd* file.
2. From the list, choose the name of the user to remove from the system. UniVerse displays information about that user and asks if you want to delete the user.
3. If you choose **Yes**, UniVerse deletes the user by removing a line from the */etc/passwd* file, displays the list of current users again, and asks if you want to delete another user.
4. Choose **Yes** to delete another user, or **No** to return to the System Administration menu.

Maintaining UniVerse Accounts

You can do any of the following tasks to maintain UniVerse accounts:

- Add a new account.
- Change information about an existing account.
- Delete an account from the system.

You can perform these tasks using the System Administration menus. To maintain accounts in this way, choose **Accounts** from the System Administration menu, then choose **Accounts** from the submenu. The following sections tell how to use the System Administration menus to add, change, and delete UniVerse accounts.

Adding a New UniVerse Account

To add a new UniVerse account:

1. Choose **Accounts** from the System Administration menu, then choose **Accounts** from the submenu, then choose **Add an Account**. UniVerse displays the Add an Account screen.
2. At the **Account Name** prompt, enter the name of the new UniVerse account. (Typically, UniVerse account names are all uppercase letters, whereas UNIX user names are all lowercase letters.)
3. Before specifying the new name, you can view the list of existing accounts. To list the current contents of the UV.ACCOUNT file, press **F4** or enter an asterisk (*****) at the prompt.
4. At the **Directory** prompt, enter one of the following:
 - The full path of the UNIX directory where the UniVerse account is to be located
 - An existing user's login name
 - A user ID number
 - The name of an existing UniVerse account

One of the following results occurs depending on what you enter at the **Directory** prompt:

- If you enter an existing directory pathname, the UniVerse account is created in that directory. If you enter the path of a directory that does not exist, the directory is created.
- If you enter an existing user login name or user ID number, that user's home directory becomes the directory for the UniVerse account.
- If you enter an existing UniVerse account name, the new account name becomes a synonym for the existing UniVerse account.



5. UniVerse asks if you want to set default file permissions for the directory where the account is located. To accept the default permissions, press **ENTER** at the prompt.

If you answer **No**, you are prompted to enter the file permissions you want to assign. Valid values are in either of the following formats: a 9-character string, or a 3-digit octal value.

***Note:** The file permissions assigned here will be applied to all files and subdirectories contained in the directory where the UniVerse account is located, regardless of whether or not they are UniVerse files and directories.*

6. UniVerse prompts you to enter the name of a user to be the owner of the account and its files. You can enter the name of either an existing user or a new one. To see a list of all users currently defined in the `/etc/passwd` file, press **F4** or enter an asterisk (*) at the prompt.

If you make an existing user the owner, that user's group becomes the group owner of the account.

If you create a new user to be the account's owner, you are prompted to specify a group owner for the account. Enter the group name at the prompt. To see a list of all groups currently defined in the `/etc/group` file, press **F4** or enter an asterisk (*) at the prompt.

7. UniVerse displays a list of compatibility flavors. UniVerse accounts can be one of several standard flavors: INFORMATION, PICK, REALITY, IN2, PIOPEN, or IDEAL UniVerse (IDEAL is the default).
8. Choose the compatibility flavor for the account you are creating.
9. UniVerse asks if you want to use the default LOGIN entry for the account. If you choose **No**, item c is skipped. If you choose **Yes**, UniVerse does the following:
 - Creates a directory for the account if it does not exist
 - Places the special UniVerse directories and files in the directory where the account is located
 - Copies the LOGIN entry located in the *sample* subdirectory of the UV account directory to the VOC file of the new UniVerse account
 - Edits the *.profile* file in the account's directory
 - Updates the UV.ACCOUNT file
 - Edits the `/etc/passwd` file if necessary

10. The account definition is complete. UniVerse redisplay the Add an Account screen. You can now create another UniVerse account. Or, to return to the System Administration menu, press **Esc**.

Changing an Existing UniVerse Account

To modify an existing UniVerse account definition:

1. Choose **Accounts** from the System Administration menu, then choose **Accounts** from the submenu, then choose **Modify an account**. A list of all accounts currently defined in the UV.ACCOUNT file appears.
2. Choose the name of the account that you want to change. The name of the account and the path of the directory where the account is located appear.
3. To change either the account name or the directory path, use the **Down** or **Up Arrow** key to move to the field you want to change, then enter the new account name or directory path.
4. You can change the owner, group, and file permissions on all files and subdirectories in the account directory. (This is in addition to changing the account name and directory pathname of a UniVerse account as described in step 2.)

To change permissions, press **ENTER** at the Directory prompt.

5. When you save the revised account information, UniVerse updates the UV.ACCOUNT file and edits the */etc/passwd* and *.profile* files if necessary. UniVerse asks if you want to modify another account. Choose **Yes** to make more changes, or choose **No** to return to the System Administration menu.

Deleting a UniVerse Account

To delete a UniVerse account:

1. Choose **Accounts** from the System Administration menu, then choose **Accounts** from the submenu, then choose **Delete an account**. A list of all UniVerse accounts currently defined the UV.ACCOUNT file appears.
2. Choose the name of the account you want to delete. The account name and the path of its associated directory appear.

3. The system checks to see if the account's directory is the home directory for any other users. You can choose any one of the following options:
 - Transfer ownership of the directory and its files by making some other user the owner of the directory.
 - Leave the directory as it is.
 - Delete the directory and all of its dependent files and subdirectories from the system.
4. You are asked to confirm that you want to delete the directory and all of its files. Be sure you want to do this before you choose **Yes**.

If the owner of the directory has already been deleted from the system, the directory and all its files, including the UniVerse files, remains on the system as an *orphan* directory, that is, the owner a user ID number without a corresponding entry in the */etc/passwd* file.
5. On deleting a UniVerse account from the system, UniVerse deletes its record in the UV.ACCOUNT file. You can then remove the UNIX directory associated with the account and all of its files.
6. UniVerse asks if you want to delete another account. Choose **Yes** to delete another account, or **No** to return to the System Administration menu.

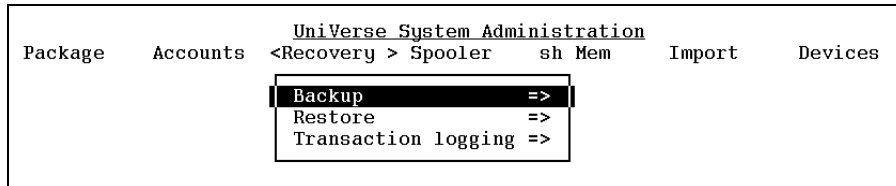
Recovery Option

Use the **Recovery** option when you want to back up and restore the following:

- The entire system
- All files in all UniVerse accounts
- The files in any specified account, with all of its children (files in dependent subdirectories)
- A specific file

The **Recovery** option also includes the transaction logging facility. The options on the Transaction logging menu are documented in this Appendix. For detailed information about transaction logging, see *UniVerse Transaction Logging and Recovery*.

When you choose **Recovery** from the UniVerse System Administration menu, the Recovery menu appears as shown in Figure .



The Recovery Menu

Backing Up and Restoring Files

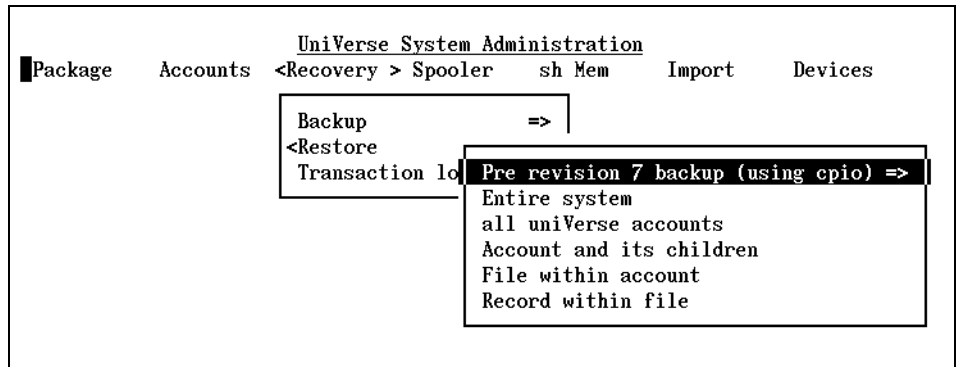
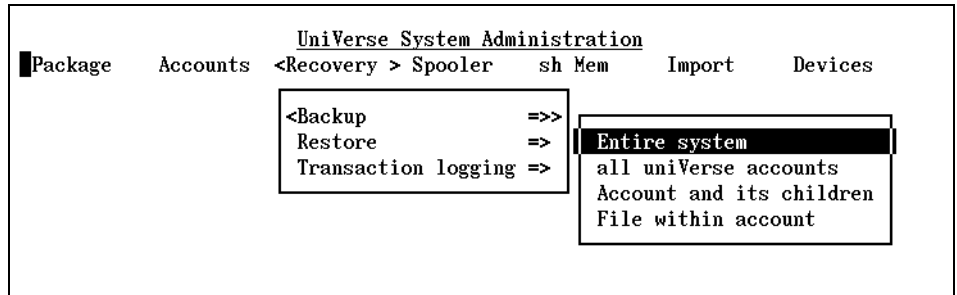
This section describes how to use the System Administration menus to back up and restore the following:

- The entire system
- The contents of all UniVerse account directories on the system
- The contents of one UniVerse account directory
- Any specified UniVerse or UNIX file

If you are using a peripheral device such as a tape or disk drive, you can mount your streamer tape or diskette in the drive before you invoke the System Administration menu. The rest of this chapter refers to streamer tape and diskette as a *backup volume*.

To back up or restore the system:

1. Choose **Recovery** from the UniVerse System Administration menu. The Recovery menu appears.
2. Choose either **Backup** or **Restore**. A menu of backup or restore options appears:



3. Specify the kind of backup or restoration you want. Choose from the following options:

Entire system. This option backs up every directory and file on the system, from *root* on down. When restoring, use this option to restore every directory and file on the backup volume.

all uniVerse accounts. This option backs up the contents of all directories defined in the UV.ACCOUNT file. All files in the directories and their dependencies are backed up, whether or not they are UniVerse files and directories.

When restoring, use this option to restore all directories on the backup volume. UVRESTORE restores only directories defined in the current UV.ACCOUNT file.

Account and its children. This option backs up the directory containing the specified UniVerse account and all other dependent files and subdirectories. When restoring, use this option to restore the account you specify. You can also use this option to back up the contents of directories that are not set up as UniVerse accounts.

File within account. This option backs up a UniVerse file, a UNIX file, or a directory. When restoring, use this option to restore the file you specify.

The Restoration menu has two additional options:

Pre revision 7 backup (using cpio). This option restores backups made with the *cpio* command.

Record within file. This option restores a record from a UniVerse file.

4. Depending on whether you chose **Backup** or **Restore** in step 2, UniVerse displays the UVBACKUP or UVRESTORE screen.
5. After you finish entering data in the UVBACKUP or UVRESTORE screen, the system asks you if you are ready to start the backup or restoration. Choose **Yes** to start the backup or restoration.



***Note:** When you restore UniVerse files that have secondary indexes, rebuild the indexes with the BUILD.INDEX command.*

Using the UVBACKUP Screen

When you choose an option from the Backup menu, UniVerse displays the UVBACKUP screen. Look for one of the following UVBACKUP screen titles:

- UVBACKUP Entire System
- UVBACKUP All uniVerse Accounts
- UVBACKUP A uniVerse Account & Children
- UVBACKUP A File within a uniVerse Account

The following example corresponds to the **Account and its children** option on the Backup menu.

UVBACKUP A uniVerse Account & Children

Backup Device: /tmp/doc.example

Block size: 8192

Type: FULL Compression: NO Verbose: ON Level: FILE

Image Label: Backup of DOCEX Account

To Backup: DOCEX (/rd2/qa/test/UVback)

File: ----

Help Region

Enter an entry from the &DEVICE& file, or a unix path. Press F4 to get a listing of &DEVICE& entries. Press F10 for more help.

To exit this program without changes, press the <ESCAPE> key.

A Sample UVBACKUP Screen

The basic format is the same for all UVBACKUP screens. Certain fields may or may not be available for data entry depending on which backup option you specified.

The default values for UVBACKUP screen data entry fields are stored in the UniVerse SYS.MESSAGE file. The record ID is 085619. As a UniVerse Administrator you can edit this record to customize the UVBACKUP screen defaults for your site. Upgrading or reinstalling UniVerse will overwrite these records.

Use the arrow keys to move forward and backward inside the UVBACKUP screen. The Help Region at the bottom of the screen gives information about valid entries for the current field.

Specifying the Backup Device

The **Backup Device** field tells where to write the formatted output from UVBACKUP. Specify either of the following:

- A full UNIX path (that is, the path begins with /, which is the root directory).
- The name of a device defined in the &DEVICE& file. UVBACKUP uses the path defined in field 6 of the &DEVICE& record. The pathname should specify a rewind device. See Chapter 10, "[Configuring Peripheral Devices](#)" for information about entries in the &DEVICE& file.

If you are backing up either the entire system or all UniVerse accounts on the system and you have not mounted the backup volume in the tape drive, you are prompted to do so.

Specifying the Block Size

Specify the block size in increments of 512 bytes. The default is 8192. The minimum is 512. The maximum is defined by the configurable parameter BLKMAX in the *uv.config* file.

Specifying a Full, Weekly, or Daily Backup

Specify one of the following backup procedures at the **Type** prompt. You can enter the word or just the first letter of the word.

- **FULL** – (Default) Saves all specified directories and files.
- **WEEKLY** – Saves only those records in UniVerse hashed files (types 2 through 18 and type 30) that have been changed since the last full or weekly backup, including records previously saved by a **DAILY** backup.
- **DAILY** – Saves only those records in UniVerse hashed files that have been changed since the last full, daily, or weekly backup.

The **Compression** field is not available.

Specifying What to Display on the Terminal Screen

Specify one of the following options at the **Verbose** prompt. You can enter the word or just its first two letters.

- ON – (Default) UVBACKUP displays the image label and the pathnames of the files as they are backed up. If there is an error, you get a message describing the problem. (UVBACKUP output is like UNIX *cpio* command output.)
- OFF – Turns off all terminal output, including error messages.

If you specified ON at the `Verbose` prompt, specify at the `Level` prompt whether you want to display record IDs as well as paths. You can enter the word or just its first letter.

- FILE – (Default) Displays pathnames of files only.
- ITEM – Displays pathnames of files, and, for UniVerse hashed files, displays the names of records.

Specifying a Label to Identify the Backup

Enter a label (up to 60 characters) at the `Image Label` prompt. All characters are valid. (Use the *uvbackup* command from a UNIX shell to enter a longer label.)

When you restore anything from this backup, UVRESTORE displays the image label so you can verify that it is the backup you want.

Specifying the Account You Want to Back Up

If you are backing up the entire system or all UniVerse accounts, UniVerse fills in the **To Backup** field for you.

If you are backing up a UniVerse account and its children, or if you are backing up a file in an account, enter either of the following at the `To Backup` prompt:

- The name of an account defined in the UV.ACCOUNT file. (Press **F4** to list the entries in the UV.ACCOUNT file.) UVBACKUP uses the path defined in field 11 of the UV.ACCOUNT record.
- The full path of a UNIX directory, beginning with a / character. If the directory contains multiple accounts, UVBACKUP backs up all accounts in the directory.

When you back up a UniVerse account, UVBACKUP saves only those files stored in the specified directory and its subdirectories. It does not back up files in remote accounts to which Q-pointers and remote F-pointers refer, nor does it back up multiple data files of a UniVerse file if the data files are stored in remote accounts.

Specifying the File You Want to Back Up

If you are backing up the entire system or one or more UniVerse accounts, the **File** field is void.

If you are backing up a file in an account, enter the name of the file at the **File** prompt. You can specify either the UNIX file name (not the full path) or the name of the file as defined in the account's VOC file. Press **F4** to list the UniVerse files defined in the VOC file.

When you back up a UniVerse file, UVBACKUP saves only the data file. You must back up the file dictionary separately (by specifying *DICT filename*). You must also back up a file's secondary indexes separately (by specifying *I_filename*).

When you back up a distributed file, UVBACKUP saves only the file header. You must specify each part file you want to back up.

To specify a type 1 or type 19 file, use the name of the file as it is defined in the VOC file.

Starting the Backup

After you finish entering information, the system asks you if you are ready to start the backup. Choose **No** to return to the Backup Device prompt. Choose **Yes** to start the backup.

How UVBACKUP Displays File Pathnames

Here is the output generated by the menu selections shown in Figure :

```
Backup Date      : Fri Aug 23 17:06:46 1996
Reel Number     : 1
Compression     : False
Image Type      : Full Backup
Block Size      : 8192 bytes
Label           : Backup of DOCEX Account

Backing up /rd2/qa/test/UVback/errlog
Backing up /rd2/qa/test/UVback/BACK1
Backing up /rd2/qa/test/UVback/BACK1/.Type1
Backing up /rd2/qa/test/UVback/BACK1/1
Backing up /rd2/qa/test/UVback/BACK1/2
Backing up /rd2/qa/test/UVback/BACK1/3
Backing up /rd2/qa/test/UVback/BACK1/4
Backing up /rd2/qa/test/UVback/BACK1/5
Backing up /rd2/qa/test/UVback/BACK1/6
```

```

Backing up /rd2/qa/test/UVback/BACK1/7
Backing up /rd2/qa/test/UVback/BACK1/8
Backing up /rd2/qa/test/UVback/BACK1/9
Backing up /rd2/qa/test/UVback/BACK1/10
Backing up /rd2/qa/test/UVback/D_BACK1
Backing up /rd2/qa/test/UVback/VOC
Backing up /rd2/qa/test/UVback/D_ORIG
Backing up /rd2/qa/test/UVback/D_VOC
Backing up /rd2/qa/test/UVback/ORIG
Backing up /rd2/qa/test/UVback/VOCLIB
.
.
.
Backing up /rd2/qa/test/UVback/BP
Backing up /rd2/qa/test/UVback/BP/CLEARALL
Backing up /rd2/qa/test/UVback/BP/MAKE.FILES
Backing up /rd2/qa/test/UVback/BP/RM.FILES
Backing up /rd2/qa/test/UVback/BP/.Type1
Backing up /rd2/qa/test/UVback/BP.O
Backing up /rd2/qa/test/UVback/BP.O/CLEARALL
Backing up /rd2/qa/test/UVback/BP.O/MAKE.FILES
Backing up /rd2/qa/test/UVback/BP.O/RM.FILES
Backing up /rd2/qa/test/UVback/BP.O/.Type1
Backing up /rd2/qa/test/UVback/&SAVEDLISTS&
Backing up /rd2/qa/test/UVback/&SAVEDLISTS&/.Type1

EndOfUvbackup

```

When the backup is complete, UniVerse asks if you want to return to the UVBACKUP screen. Choose **Yes** to redisplay the screen. Choose **No** to return to the main System Administration menu.

Backing Up on Multiple Tapes

If your backup does not fit on a single tape, UniVerse rewinds the tape and prompts you to enter the name of a backup device. Do either of the following:

- If using the same backup device, remove the first tape and load the next tape, then enter the name of the backup device at the prompt.
- If using a different backup device, make sure the tape is loaded, then enter the backup device name at the prompt.

If you have not mounted the tape in the drive, UniVerse prompts you to do so.

Using the UVRESTORE Screen

When you choose an option from the Restore menu, UniVerse displays the UVRESTORE screen corresponding to the option chosen. Look for one of the following UVRESTORE screen titles:

- UVRESTORE Entire Image
- UVRESTORE All UniVerse Accounts from Image
- UVRESTORE A UniVerse Account & Children from Image
- UVRESTORE A File within a UniVerse Account from Image
- UVRESTORE A Record within a UniVerse File from Image

The screen in the following example corresponds to the **Account and its children** option on the Restoration menu.

UVRESTORE A uniVerse Account & Children from Image

Restore Device:

Type: FULL Compression: NO Verbose: ON Level: FILE

Image Label: Backup of DOCEX Account

Restore: /rd2/qa/test/UVback=/tmp/UVback

File : ----

Record : ----

Options:

Help Region

Enter an entry from the &DEVICE& file, or a unix path. Press F4 to get a listing of &DEVICE& entries. Press F10 for more help.

To exit this program without changes, press the <ESCAPE> key.

A Sample UVRESTORE Screen

The basic format is the same for all UVRESTORE screens. Certain fields may or may not be available for data entry depending on which restore option you specified.

The default values for UVRESTORE screen data entry fields are stored in the UniVerse SYS.MESSAGE file. The record ID is 085720. As a UniVerse Administrator you can edit this record to customize the UVRESTORE screen defaults for your system. Upgrading or reinstalling UniVerse overwrites any changes to this record.

Use the arrow keys to move forward and backward inside the UVRESTORE screen. The Help Region at the bottom of the screen lists valid entries for the current field.

Specifying the Restore Device

The **Restore Device** field tells the location of the backup image being restored. Specify either of the following:

- A full UNIX path (that is, the path begins with /, which is the root directory).
- The name of a device defined in the &DEVICE& file. UVRESTORE uses the path defined in field 6 of the &DEVICE& record. See Chapter 10, ["Configuring Peripheral Devices"](#) for information about creating entries in the &DEVICE& file.

If you are restoring either the entire system or all UniVerse accounts on the system and you have not mounted the backup volume in the tape drive, you are prompted to do so.

Ensuring the Backup Image Is the One You Want

After you enter the restore device name, the following screen appears.

UVRESTORE A uniVerse Account & Children from Image	
Restore	
Type:	Backup Date : Fri Aug 27 17:06:46 1993
	Reel Number : 1
	Compression : False
Image La	Image Type : Full Backup
	Block Size : 8192 bytes
Restore:	Label : Backup of DOCEX Account
File :	
Record :	Is this the correct image to restore from?
Options:	<input checked="" type="radio"/> Yes <input type="radio"/> No
<div>Help Region</div> <p>Enter an entry from the &DEVICE& file, or a unix path. Press F4 to get a listing of &DEVICE& entries. Press F10 for more help.</p> <p>To exit this program without changes, press the <ESCAPE> key.</p>	

The UVRESTORE Image Label Screen

Choose **No** to return to the **Restore Device** field. Choose **Yes** to continue.

UniVerse reads the backup type (full, weekly, daily) from the backup image and displays it in the **Type** field.

The **Compression** field is not available.

Specifying What to Display on the Terminal Screen

Specify one of the following options at the *Verbose* prompt. You can enter the word or just its first two letters.

- **ON** – (Default) UVRESTORE displays the image label and the paths of files as they are restored. If there is an error, you get a message describing the problem. (UVRESTORE output is similar to UNIX *cpio* command output.)
- **OFF** – Turns off all terminal output, including error messages.

If you specified ON at the `Verbose` prompt, specify at the `Level` prompt whether you want to display record IDs as well as paths. You can enter the word or just its first letter.

- `FILE` – (Default) Displays pathnames of files only.
- `ITEM` – Displays pathnames of files, and, for UniVerse hashed files, the names of records.

UniVerse reads the image label from the backup image and displays it in the **Image Label** field.

Specifying the Accounts You Want to Restore

If you are restoring the entire system or all UniVerse accounts, UniVerse fills in the **Restore** field for you.

If you are restoring one or more UniVerse accounts and their children, a file in an account, or a record in a file, enter one of the following at the `Restore` prompt:

- The names of one or more accounts defined in the `UV.ACCOUNT` file. Separate account names with commas. (Press **F4** to list the entries in the `UV.ACCOUNT` file.) `UVRESTORE` uses the path defined in field 11 of the `UV.ACCOUNT` record.
- The full paths of one or more UNIX directories. Paths should begin and end with a `/` character, and they must be identical to the path stored on the backup volume. Separate paths with commas.
- You can specify multiple accounts if they are all in the same directory. Use an asterisk (`*`) as the last character in the UNIX path. For example, `/usr/*` restores all accounts in the `/usr` directory, and `/usr/SAL*` restores all accounts beginning with `SAL` (such as `SALES.EAST` and `SALES.WEST`) in the `/usr` directory.

If you want to restore the account (or file or record) to a different account on disk, enter the name of the account on the backup volume, then enter an equal sign (`=`), then enter the name of the account on disk to which you want to restore the account, file, or record. For example, to restore the backed-up account `UVback` to the `/tmp` directory on disk, enter `/rd2/qa/test/UVback=/tmp/UVback`.

UVRESTORE A uniVerse Account & Children from Image			
Restore Device: /tmp/doc.example			
Type: FULL	Compression: NO	Verbose: ON	Level: FILE
Image Label: Backup of DOCEX Account			
Restore: /rd2/qa/test/UVback=/tmp/UVback			
File	:	----	
Record	:	----	
Options:			
Help Region			
Enter an entry from the &DEVICE& file, or a unix path. Press F4 to get a listing of &DEVICE& entries. Press F10 for more help.			
To exit this program without changes, press the <ESCAPE> key.			

The *UVback* Account Restored to the */tmp* Directory on Disk

Specifying the Files You Want to Restore

If you are restoring the entire system or one or more UniVerse accounts, the **File** field is void.

If you want to restore one or more files from an account, enter the names of the UNIX files or directories (not full pathnames) at the **File** prompt exactly as they are stored on the backup volume. If you want to restore one or more records from a file, enter the name of the file at the **File** prompt.

Use an asterisk (*****) as the last character in the UNIX filename to specify multiple files. If you enter just an asterisk at the **File** prompt, all files in the account directory are restored. If you enter **ORDERS***, all files beginning with ORDERS (such as ORDERS.JAN and ORDERS.FEB) are restored.

If you want to restore a file or record to a different file on disk, enter the name of the file on the backup volume, then an equal sign (**=**), then the name of the file on the disk to which you want to restore the file or record. For example, to restore the backed-up file BACK1 to a disk file called BACK1.A, enter **BACK1=BACK1.A**.

UVRESTORE A File within a uniVerse Account from Image			
Restore Device: /tmp/doc.example			
Type: FULL	Compression: NO	Verbose: ON	Level: FILE
Image Label: Backup of DOCEX Account			
Restore: /rd2/qa/test/UVback			
File : BACK1=BACK1.A			
Record : ----			
Options: OVERWRITE			
Help Region			
Enter an entry from the &DEVICE& file, or a unix path. Press F4 to get a listing of &DEVICE& entries. Press F10 for more help.			
To exit this program without changes, press the <ESCAPE> key.			

BACK1 Restored to New File BACK1.A in Account *UVback*

Specifying the Record You Want to Restore

If you are restoring the entire system, one or more UniVerse accounts, or a file in an account, the **Record** field is void.

If you want to restore a record from a file, enter the record ID at the **Record** prompt exactly as it is stored on the backup volume. If you want to restore the record to a different record on disk, enter the record ID on the backup volume, then enter an equal sign (=), then enter the record ID to which you want to restore the record on disk. For example, to restore the backed-up record 1 to a new record on disk called 11, enter **1=11**.

Use an asterisk (*) at the **Record** prompt to specify all records in the file. Use an asterisk as the last character of the record ID to specify multiple records. If you enter **11***, all records beginning with 11 (such as 1100, 1101, 1102, and so on) are restored.

UVRESTORE A Record within a uniVerse File from Image			
Restore Device: /tmp/doc.example			
Type: FULL	Compression: NO	Verbose: ON	Level: FILE
Image Label: Backup of DOCEX Account			
Restore: /rd2/qa/test/UVback			
File : BACK2			
Record : 1=11			
Options: OVERWRITE			
Help Region			
Enter an entry from the &DEVICE& file, or a unix path. Press F4 to get a listing of &DEVICE& entries. Press F10 for more help.			
To exit this program without changes, press the <ESCAPE> key.			

Record 1 Restored as 11 in File BACK2

Specifying Restoration Options

Enter any of the following options at the Options prompt, or press **ENTER**. You can enter the word or just its first two letters.

- INDEX – Generates a contents listing from the image. No data is restored.
- PAGING – Displays the contents listing one page at a time. The current TERM settings determine the page size.
- OVERWRITE – For each restored file, removes the disk file with the same name, then restores the file from the backup image. If you do not specify OVERWRITE, *uvrestore* restores only files from the backup image that do not exist on disk.

If you are restoring a record in a file, OVERWRITE has no effect. UVRESTORE always overwrites a record.

- **PROMPT** – Interactively restores files and records. Before a file or record is restored, you are prompted to verify restoration. Enter one of the following at the prompt:

y Restore the file or record.

n Do not restore the file or record.

d Disable prompting after the current prompt.

q Quit. At the prompt, choose **Yes** to return to the UVRESTORE screen. Choose **No** to return to the main System Administration menu.

Starting the Restoration

When you finish entering information, the system asks if you want to start the restoration. Choose **No** to return to the Restore Device prompt. Choose **Yes** to start the restoration.

If you entered **ON** at the Verbose prompt, the output looks like this:

```
Backup Date      : Fri Aug 23 17:06:46 1996
Reel Number     : 1
Compression      : False
Image Type       : Full Backup
Block Size       : 8192 bytes
Label            : Backup of DOCEX Account

Restoring /rd2/qa/test/UVback/errlog (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/.Type1 (17:06:46 1996,
000)
Restoring /rd2/qa/test/UVback/BACK1/1 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/2 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/3 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/4 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/5 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/6 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/7 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/8 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/9 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/BACK1/10 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/D_BACK1 (17:06:46 1996, 000)
Restoring /rd2/qa/test/UVback/VOC (17:06:47 1996, 000)
Restoring /rd2/qa/test/UVback/D_ORIG (17:06:47 1996, 000)
Restoring /rd2/qa/test/UVback/D_VOC (17:06:47 1996, 000)
Restoring /rd2/qa/test/UVback/ORIG (17:06:47 1996, 000)
Restoring /rd2/qa/test/UVback/VOCLIB (17:06:48 1996, 000)
.
.
.
```

```

Restoring /rd2/qa/test/UVback/BP (17:06:54 1996, 000)
Restoring /rd2/qa/test/UVback/BP/CLEARALL (17:06:54 1996, 000)
Restoring /rd2/qa/test/UVback/BP/MAKE.FILES (17:06:54 1996,
000)
Restoring /rd2/qa/test/UVback/BP/RM.FILES (17:06:55 1996, 000)
Restoring /rd2/qa/test/UVback/BP/.Type1 (17:06:55 1996, 000)
Restoring /rd2/qa/test/UVback/BP.O (17:06:55 1996, 000)
Restoring /rd2/qa/test/UVback/BP.O/CLEARALL (17:06:55 1996,
000)
Restoring /rd2/qa/test/UVback/BP.O/MAKE.FILES (17:06:55 1996,
000)
Restoring /rd2/qa/test/UVback/BP.O/RM.FILES (17:06:55 1996,
000)
Restoring /rd2/qa/test/UVback/BP.O/.Type1 (17:06:55 1996, 000)
Restoring /rd2/qa/test/UVback/&SAVEDLISTS& (17:06:55 1996,
000)
Restoring /rd2/qa/test/UVback/&SAVEDLISTS&/.Type1 (17:06:55
1996, 000)

EndOfUvrestore

```

When the restoration is complete, UniVerse asks if you want to return to the UVRESTORE screen. Choose **Yes** to redisplay the screen. Choose **No** to return to the main System Administration menu.

Restoring from Multiple Tapes

If your backup comprises several tapes, UniVerse rewinds the first tape and prompts you to enter the name of a backup device. Do either of the following:

- If you use the same restoration device, remove the first tape and load the next tape, then enter the name of the restoration device at the prompt.
- If you use a different restoration device, make sure the tape is loaded, then enter the name of the restoration device at the prompt.

If you have not mounted the tape in the drive, UniVerse prompts you to do so.

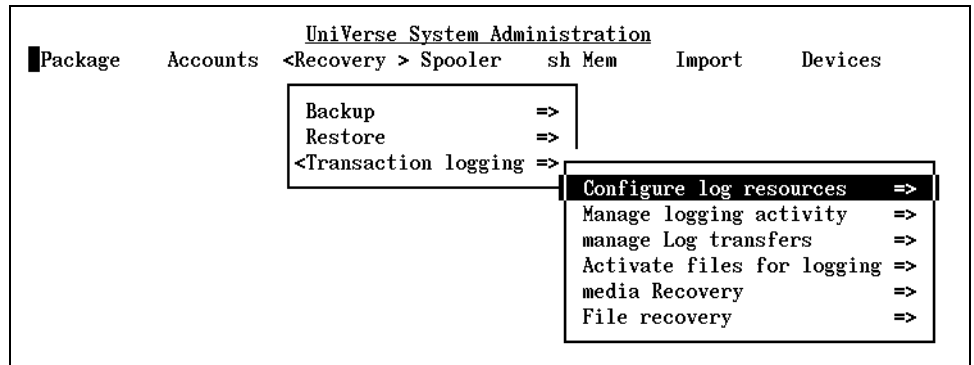
Transaction Logging

The Transaction Logging menu gives access to further menus that enable you to perform the following transaction logging and related activities:

- Setting up for transaction logging
- Setting the systemwide state of transaction logging
- Activating UniVerse files for logging

- Managing logging activity
- Transferring log files to tape
- Releasing log files
- Recovering by rolling forward updates from log files

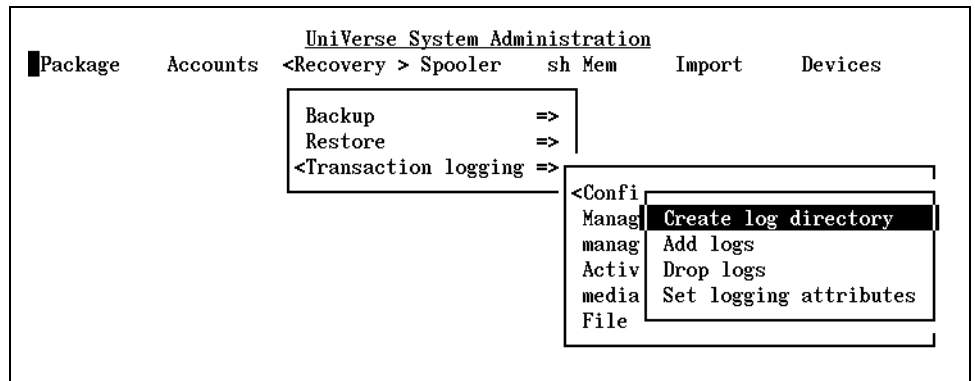
Figure shows the Transaction Logging menu. The options on this menu take you to submenus that are described on the following pages.



The Transaction Logging Menu

Configure Log Resources Menu

The **Configure log resources** option on the Transaction Logging menu displays the Configure Log Resources menu. This menu outlines the initial requirements for setting up transaction logging.



The Configure Log Resources Menu

The Configure Log Resources menu options are as follows:

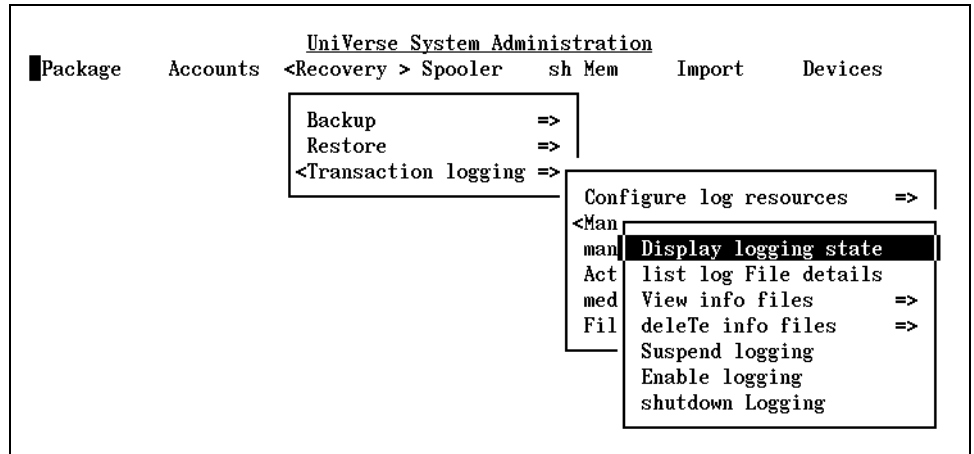
Option	Description
Create log directory	Creates the log directory (on a different disk from your UniVerse files), using the CREATE.LDIR command. You are prompted for the path.
Add logs	Creates log files in the log directory, using the CREATE.LFILE command. You are prompted for the size in bytes (the default is 512) and for the number of files.
Drop logs	Deletes Available log files in the log directory, using the DELETE.LFILE command. The most recently created log file is deleted first, then the next most recently created, and so on. The UV_LOGS file is also updated.
Set logging attributes	Sets the transaction logging modes and the archive type, using the SET.LOG.ATTR command. If you are logging file updates to disk, you can set transaction logging to run in archive mode, checkpoint mode, or both. If you are logging file updates to tape, you can set transaction logging to run only in archive mode. If you set the archive type to TAPE, you must specify one or more tape devices to which to log updates. The tape devices must be defined as DC or DT types in the &DEVICE& file.

Configure Log Resources Menu Options

Manage Logging Activity Menu

The **Manage logging activity** option on the Transaction Logging menu displays the Manage Logging Activity menu.

Use this menu to display or change the systemwide state of transaction logging, to display information about log files, and to display or delete information about the log, checkpoint, and roll-forward daemons.



The Manage Logging Activity Menu

The Manage Logging Activity menu options are as follows:

Option	Description
Display logging state	Lists the systemwide state of transaction logging.
list log File details	Displays information about the log files, using the command LIST UV_LOGS BY.DSND @ID.
View info files	Displays the View Info Files menu.
deleTe info files	Displays the Delete Info Files menu.

Manage Logging Activity Menu Options

Option	Description
Suspend logging	Suspends transaction logging systemwide, using the SUSPEND.RECOVERY command.
Enable logging	Enables transaction logging systemwide, using the ENABLE.RECOVERY command.
shutdown Logging	Disables transaction logging systemwide, using the SHUTDOWN.RECOVERY command.

Manage Logging Activity Menu Options (Continued)

View Info Files Menu

Use this menu to display any of the three information files: *uvlogd.info*, *uvchkd.info*, or *uvrolf.info*. The View Info Files menu options are as follows:

Option	Description
dispLay logging info file	Displays the logging information file (<i>uvlogd.info</i>).
display Checkpoint info file	Displays the checkpoint information file (<i>uvchkd.info</i>).
display Rollforward info file	Displays the roll-forward information file (<i>uvrolf.info</i>).

View Info Files Menu Options

Delete Info Files Menu

Use this menu to delete any of the three information files. The Delete Info Files menu options are as follows:

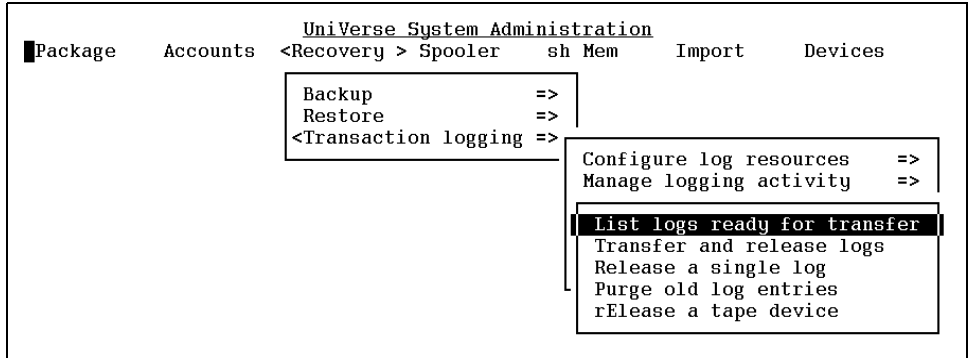
Option	Description
deLeTe logging info file	Deletes the logging information file (<i>uvlogd.info</i>). Do not delete this file while transaction logging is in the initializing, warmstart, or enabled states.
delete Checkpoint info file	Deletes the checkpoint information file (<i>uvchkd.info</i>). Do not delete this file while the checkpoint daemon is active.
delete Rollforward info file	Deletes the roll-forward information file (<i>uvrolf.info</i>). Do not delete this file while a roll-forward is in progress, because you will lose all further output from the roll-forward.

Delete Info Files Menu Options

Manage Log Transfers Menu

The **manage Log transfers** option on the Transaction Logging menu displays the Manage Log Transfers menu.

Use this menu to manage log files on the system. You should use this menu to ensure that there is always enough space in log files to record any updates to recoverable files. If you are running out of space in the log files, you can use this menu to save the log files to tape and release the disk space they occupy.



The Manage Log Transfers Menu

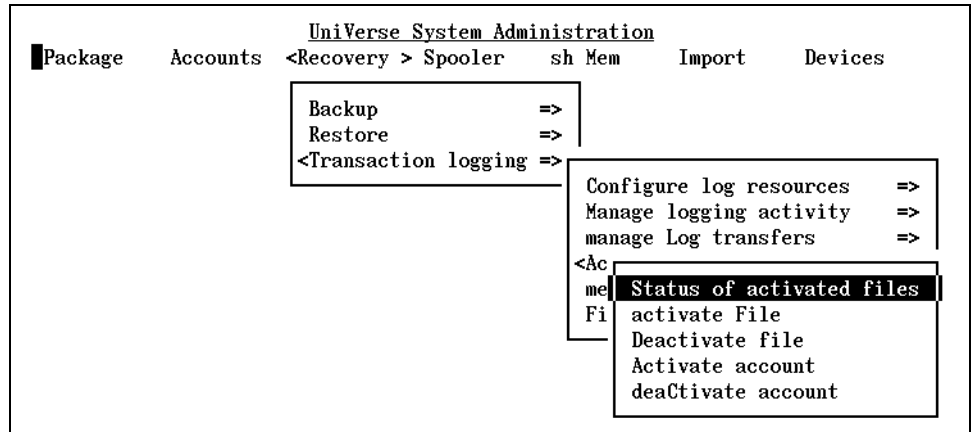
The Manage Log Transfers menu options are as follows:

Option	Description
List logs ready for transfer	Lists the log files that are Full and ready to be backed up, using the command LIST UV_LOGS WITH STATUS = FULL BY @ID.
Transfer and release logs	Backs up and releases Full log files, using the <i>tlsave</i> shell script.
Release a single log	Prompts for the number of a Full log file, then releases that log file, using the RELEASE.LFILE command. It prompts for the log file number.
Purge old log entries	Prompts you to enter a date, then removes out-of-date records from the UV_LOGS file that refer to log files released earlier than the specified date.
rElease a tape device	Prompts you to enter a tape device name, then releases the tape device, making it available again for logging. Use this option only after you mount a new tape in the device.

Manage Log Transfers Menu Options

Activate Files for Logging Menu

The **Activate files for logging** option on the Transaction Logging menu displays the Activate Files for Logging menu. This menu lets you activate files for transaction logging, making them recoverable.



The Activate Files for Logging Menu

The Activate Files for Logging menu options are as follows:

Option	Description
Status of activated files	Lists the current status of all recoverable files, using the command LIST UV.TRANS.
activate File	Prompts for an account name and a file name, then activates the file for transaction logging, making it recoverable. You cannot activate type 1 and type 19 files for logging.

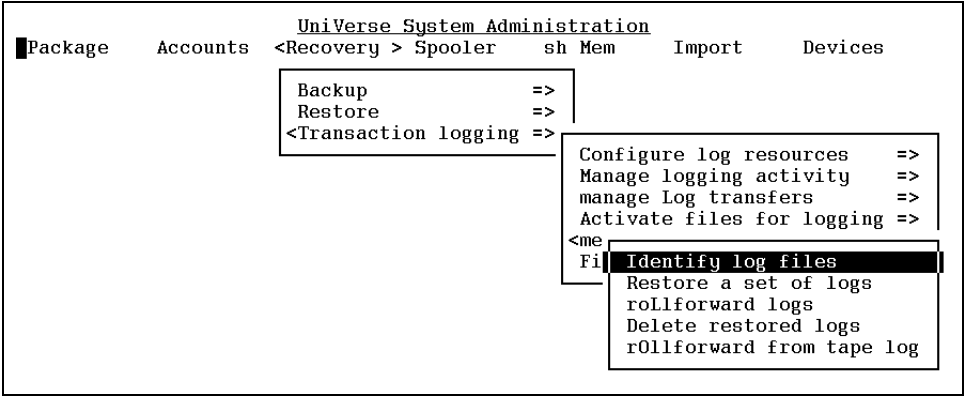
Activate Files for Logging Menu Options

Option	Description
Deactivate file	Prompts for an account name and a file name, then deactivates the file for transaction logging. The name of the deactivated file is not removed from UV.TRANS, but its status is set to OFF.
Activate account	Prompts for an account name, then activates all files in that account for transaction logging.
deaCtivate account	Prompts for an account name, then deactivates all files in that account for transaction logging.

Activate Files for Logging Menu Options (Continued)

Media Recovery Menu

The **media Recovery** option on the Transaction Logging menu displays the Media Recovery menu. This menu lets you step through the process of roll-forward recovery.



The Media Recovery Menu

The Media Recovery menu options are as follows:

Option	Description
Identify log files	Prompts for the name of a select list containing full paths of UniVerse files that require roll-forward recovery. It then uses the RECOVERY.CHECKPOINT command to search the selected files for the file containing the earliest log file checkpoint. This log file should be rolled forward first. You can then determine how many log files to restore in view of the disk space available.
Restore a set of logs	Prompts for the range of log files to use for roll-forward recovery, and for the directory to which to restore them. It then restores the log files from tape using the LOG.RESTORE command. Restoring log files from tape does not change the UV_LOGS file.
roLlforward logs	Prompts for the name of a select list containing full paths of UniVerse files that require roll-forward recovery, the name of the directory containing the log files, and the range of log files to roll forward. It then rolls forward updates from the restored log files in date-time sequence.
Delete restored logs	Prompts for the range of log files to delete, and for the name of the directory containing the log files, then uses the DEL.RFILE command to delete the log files just restored from tape and rolled forward. Deleting log files that have been rolled forward frees up disk space for subsequent restorations. It does not change the UV_LOGS file.
rOllforward from tape log	Prompts for the name of a select list containing full paths of UniVerse files that require roll-forward recovery, a list of tape devices containing the log files, and the first and last tape log files to roll forward. It then rolls forward updates from the tape log file in date-time sequence.

Media Recovery Menu Options

***Note:** We recommend that you restore your log files into a directory other than the log directory. Delete only the restored log files.*

File Recovery Menu

The **File recovery** option on the Transaction Logging menu displays the File Recovery menu. This menu rolls forward updates to one particular file.



UniVerse System Administration			
Package	Accounts	<Recovery > Spooler	sh Mem Import Devices
		Backup =>	
		Restore =>	
		<Transaction logging =>	
			Configure log resources =>
			Manage logging activity =>
			manage Log transfers =>
			Activate files for logging =>
			media Recovery =>
			Rollforward file
			Clear file inconsistency flag
			rollforward file from tape

The File Recovery Menu

The File Recovery menu options are as follows:

Option	Description
Rollforward file	Prompts for the range of log files to use for roll-forward recovery, the name of the file you want to roll forward, and the name of the directory where the log files are located. It then rolls forwards updates to the specified files.
Clear file inconsistency flag	Prompts for the path of the UniVerse file, then clears the flag marking the file as inconsistent, using the RECOVERY.CONSISTENT command.
rollforward file from tape	Prompts for the full path of the file you want to roll forward, a list of tape devices containing the log files, and the number of the log file to use for roll-forward recovery. It then rolls forward updates to the specified files.

File Recovery Menu Options

Spooler Option

Use the **Spooler** option to define and configure system printers and to perform routine spooler maintenance.

When you choose **Spooler** from the UniVerse System Administration menu, the Spooler menu appears as shown in the following example.

UniVerse System Administration					
Package	Accounts	Recovery	<Spooler > sh Mem	Import	Devices
			Status	=>	
			Job control	=>	
			Modify job characteristics	=>	
			Queue management	=>	
			Device management	=>	
			sPooler management	=>	
			printer Group management	=>	
			printer groUp queue management	=>	
			error and activity Logs	=>	

The Spooler Menu

Spooler Status Report

To display a spooler status report, choose **Spooler** from the System Administration menu, then choose **Status**. Choose one of the following:

Quick status – All spooler queues except the empty ones

Empty queues too – All spooler queues whether empty or not

Active jobs – Only print jobs that are currently active

You can also display a spooler status report by using the SPOOL –LIST command at the UniVerse prompt, or by executing a usa command with no options from a UNIX shell.

Here is a sample spooler status report:

```

Printer: lp          Q: on      P: on      Form:
Job # Job description User name Pri Forms Size Cps Status
Delay
00001 test          julie    32      1437    1 active
0:14
00003 portrait.file kira     65 LW   12342    1 wait
00004 hold.file     judy     32 HOLD 1589    1 hold&

Printer: lp2         Q: on      P: on      Form:
no entries.

Printer: lw          Q: on      P: on      Form: LW
no entries.

Printer: lwlscape    Q: on      P: on      Form: LSCAPE
Job # Job description User name Pri Forms Size Cps Status
Delay
00002 lscape.file   larry    86 LSCAPE 213764    1 active

Printer: remote      Q: on      P: on      Form: REMOTE
no entries.

```

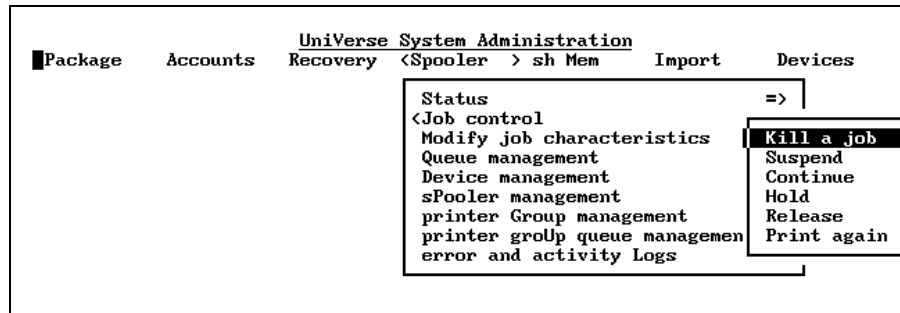
The first line for each queue shows the printer, indicating if queuing is enabled, if printing is enabled, and what form, if any, is mounted. Subsequent lines give a job-by-job summary of the queue contents. In the previous example jobs 1 and 2 are printing. Job 3 is waiting in the queue for printer *lp* although it requested the form that is currently mounted on printer *lw*, it must wait for the following reasons:

1. Job 2 is active on printer *lwlscape*, which is the same physical device as printer *lw*.
2. Queued jobs are always shown in the queue of the first defined printer, unless they are spooled to a specific printer either with the AT keyword on a SETPTR (UNIX) or SPOOL command, or with the *-p* option of the *usp* command.
3. The ampersand (&) after the word hold on job 4 indicates that a copy of this file has already been printed. An asterisk (*) would indicate a file spooled as a hold file.

Managing Print Jobs

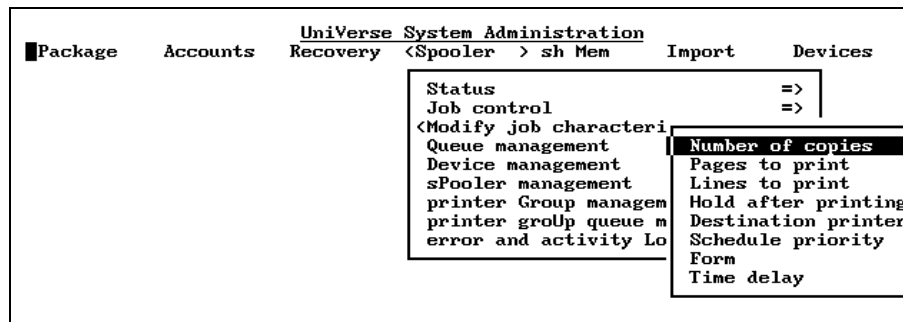
You can manage most print jobs from the **Job control** and **Modify job characteristics** options of the Spooler menu. These two options display submenus when they are chosen. The Job Control menu lets you make changes to the print jobs themselves—for example, cancelling or temporarily suspending print jobs, or printing them again. The Modify Job Characteristics menu lets you change various attributes of a print job, such as the number of pages to print, which pages to print, when to print, and so on.

The following example shows the Job Control menu.



The Spooler Job Control Menu

The next example shows the Modify Job Characteristics menu.



The Spooler Modify Job Characteristics Menu

Changing Print Job Characteristics

The following subsections describe how to change the characteristics of print jobs waiting in the queue. All these changes can be made from the Modify Job Characteristics menu.

Changing the Number of Copies You Want Printed

Choose **Modify job characteristics** from the Spooler menu, then choose **Number of copies**. At the prompts enter the print job number, then the number of copies you want.

Specifying Which Pages to Print

Choose **Modify job characteristics** from the Spooler menu, then choose **Pages to print**. At the prompts enter the print job number, then specify the page where you want printing to begin and the page where you want printing to end.

Specifying Which Lines to Print

Choose **Modify job characteristics** from the Spooler menu, then choose **Lines to print**. At the prompts enter the print job number, then specify the line where you want printing to begin and the line where you want printing to end.

Changing the Priority of a Print Job

Choose **Modify job characteristics** from the Spooler menu, then choose **Schedule priority**. At the prompts enter the print job number, then the level of priority you want to assign. 1 is the highest priority, 255 is the lowest.

Specifying When to Print a Job

Choose **Modify job characteristics** from the Spooler menu, then choose **Time delay**. You can then specify either the absolute time you want the spooler to print the job, for example, 3:30 p.m. or the relative time, such as 4 hours from now.

Controlling Print Jobs

The following subsections describe how to manipulate print jobs waiting in the queue. These changes can be made from the Job Control menu.

Holding a Print Job

If a print job is not printing, choose **Job control** from the Spooler menu, then choose **Hold**. At the prompt enter the print job number. This makes the print file a *hold file*—that is, it holds the job until you release it for printing.

To release a hold file for printing, choose **Job control** from the Spooler menu, then choose **Release**. The shell command is `usm -r print.job`.

If you want to retain a print file as a hold file after it is printed, choose **Modify job characteristics** from the Spooler menu, then choose **Hold after printing**.

Suspending a Print Job

If the job is actively printing, choose **Job control** from the Spooler menu, then choose **Suspend**. At the prompt enter the printer number. This suspends a currently printing job until you are ready to continue printing.

To restart a suspended print job, choose **Job control** from the Spooler menu, then choose **Continue**.

Killing a Print Job

Choose **Job control** from the Spooler menu, then choose **Kill a job**. At the prompt enter the print job number.

Managing the Spooler

Use the **Queue management** or the **printer groUp queue management** option from the Spooler menu to do the following:

- Enable and disable printing on a printer
- Enable and disable queuing for a printer
- Mount a form on a printer

Queue management lets you do these things for any printer. **printer groUp queue management** lets you do them for printers belonging to a printer group. For information about mounting a form on a printer, see [“Mounting a Form on a Printer”](#) on page 65.

Disabling Printing

To stop printing on a printer:

1. Choose **Queue management** or **printer groUp queue management** from the Spooler menu.
2. Choose **Halt printing**.
3. Enter the name of the printer where you want printing to stop.
4. Choose **Yes** to stop printing.

Enabling Printing

To restart printing on a printer:

1. Choose **Queue management** or **printer groUp queue management** from the Spooler menu.
2. Choose **Begin printing**.
3. Enter the name of the printer where you want to restart printing.
4. Choose **Yes** to start printing.

Disabling Queuing

To disable queuing on a printer queue:

1. Choose **Queue management** or **printer groUp queue management** from the Spooler menu.
2. Choose **Deny queuing**.
3. Enter the name of the printer where you want queuing to stop.
4. Choose **Yes** to disable queuing.

Enabling Queuing

To enable queuing on a printer queue:

1. Choose **Queue management** or **printer groUp queue management** from the Spooler menu.
2. Choose **Allow queuing**.
3. Enter the name of the printer where you want to enable queuing.

4. Choose **Yes** to enable queuing.

Defining Printers

To define printers, choose **Spooler** from the System Administration menu, choose **Device management**, then choose **Maintain devices**. The data entry screen for the *sp.config* file prompts you to enter the following information:

Printer Name	:	Baud Rate	:
Unix Pathname	:	Parity	:
Driver	:	CR Mode	:
Form	:	Tab Expansion	:
Flow Control	:	FF Delay	:
Enable Printing	:	LF Delay	:
Enable Queuing	:	Word Length	:
Lock file 1	:	Map Name	:
Lock file 2	:		
Other Options	:		

The data you enter adds a line to the *sp.config* file for each printer. An entry is also added to the &DEVICE& file. There is one logical line in *sp.config* for each printer. (If you are editing the file directly with *vi* or the UniVerse Editor, you can continue logical lines across physical lines by putting a backslash (\) or an underscore (_) at the end of all physical lines but the last that make up the logical line.)

Here is an explanation of the parameters you can specify:

Parameter	Description
Printer Name	The logical printer name; a unique name that identifies the printer. This name is used in various UniVerse commands to refer to the printer.
Unix Pathname	The UNIX path for the printer. This might be a path such as <i>/dev/lp0</i> . Be sure to assign the correct access permissions to the printer. You can set permissions for a printer with the UNIX <i>chmod(1)</i> command.
Driver	The printer driver is a device-specific post-processor for spooled output. The path of the driver can be up to 22 characters in length. If you need to specify a longer pathname, edit the <i>sp.config</i> file. The driver can be a UNIX executable or a shell script. When the driver is invoked, standard input is taken from the print file and standard output is sent to the printer device with characteristics set according to the PTERM (UNIX) options specified for that printer. In addition five command line arguments can be specified. Typical uses of the driver are to execute device-specific formatting filters such as a PostScript filter, or to redirect printing using a <i>uux</i> command.
Form	The name of the default form to be mounted on the printer. The name of the form can be up to 32 characters long. To print the file, you must specify the exact form name when queuing a spool file or when changing the mounted form or the form requested. Form names are case-sensitive. You can mount another form on the printer at any time, or you can use the usa command to mount multiple forms on the printer.
Flow Control	<p>The following types of flow control are available:</p> <p>none – Cancels any DTR or XON/XOFF flow control handshaking that may be set.</p> <p>DTR – Enables Data Terminal Ready handshaking. Loss of DTR is treated the same as the input of the stop character. If DTR is not enabled, loss of DTR is treated as a hangup, and foreground jobs are terminated.</p> <p>XON/XOFF any – Enables XON/XOFF handshaking. Any character acts the same as the XON character.</p>

sp.config Parameters

Parameter	Description
	XON/XOFF XON starts – Enables XON/XOFF handshaking. The XON character is required to <i>start</i> transmission. On Berkeley systems the XON character is the start character; on System V, XON is always CTRL-Q.
Enable Printing	Enter Y to instruct the spooler daemon to start up with printing enabled for that printer. Enter N to start the spooler with printing disabled. At any time you can enable or disable printing from the Spooler menu.
Enable Queuing	Enter Y to instruct the spooler daemon to start up with queuing enabled for that printer. Enter N to start the spooler with queuing disabled. At any time you can enable or disable queuing from the Spooler menu.
Baud Rate	The following baud rates are supported: 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200
Parity	Parity can be one of the following: NONE EVEN ODD
CR Mode	Carriage return mode, can be any of the following: no conversion – Resets all CR modes: carriage returns and newlines are not converted. convert LF to CR/LF – Converts newline to newline, carriage return on output. convert CR to LF – Converts carriage return to newline on output. no CR@(0,0) – Does not output a carriage return when the cursor is at line 0, column 0.
sp.config Parameters (Continued)	

Parameter	Description
Tab Expansion	Enter ON or OFF . If tab expansion is ON, a tab character is expanded to the proper number of spaces on output. Tab stops are set every eight columns. If tab expansion is OFF, a tab character is unchanged on output. Some terminals (such as ADDS Viewpoint) use a tab character as part of the cursor movement function. On these terminals tab expansion must be OFF for cursor movement to work properly.
FF Delay	<p>Enter * to display a list of possible formfeed delays, then choose one of the following:</p> <p>no form feeds – Clear-screens are sent to the terminal, but no formfeeds are sent to the line printer.</p> <p>no delay – Clear-screens are sent to the terminal, and formfeeds are sent to the line printer. Output of a formfeed causes no delay.</p> <p>2-second delay – Clear-screens are sent to the terminal, and formfeeds are sent to the line printer. Output of a formfeed causes a two-second delay.</p>
LF Delay	<p>Enter * to display a list of possible newline delays, then choose one of the following:</p> <p>None – No delay for each newline.</p> <p>.08sec. – A delay of about .08 second occurs after each newline.</p> <p>.10sec. – A delay of about .10 second occurs after each newline.</p> <p>.16sec. – A delay of about .16 second occurs after each newline.</p> <p>.18sec. – A delay of about .18 second occurs after each newline.</p> <p>.26sec. – A delay of about .26 second occurs after each newline.</p> <p>Teletype37 – A delay dependent on the column position occurs after each newline. This mode has been configured for Teletype model 37s.</p>

sp.config Parameters (Continued)

Parameter	Description
	<p>col+.08sec. – A delay dependent on the column position plus about .08 second occurs after each newline.</p> <p>col+.16sec. – A delay dependent on the column position plus about .16 second occurs after each newline.</p>
Word Length	The number of data bits that make up a word, not including the parity bit. Can be 5, 6, 7, or 8.
Map Name	The name of a map you want to assign to the device. For information about maps, see <i>UniVerse NLS Guide</i> .
Lock files	When a device is shared by UniVerse and UNIX system processes, it needs a special lock file created for it that coordinates access to the device when more than one process tries to access it. Field 5 of the &DEVICE& file contains the UNIX paths used to implement the locking protocol used by the UniVerse spooler and several UNIX facilities such as the spooler and <i>uucp</i> . For information about the form of the lock filename for a system, see the UNIX reference manual for the process that is sharing the device.
Other Options	You can specify any of the UniVerse PTERM options to control the port used to drive the printer. See Appendix B for a complete list of PTERM (UNIX). In addition to the PTERM options, you can also specify NORESET. The NORESET option instructs the spooler daemon not to restore printer device characteristics upon completion of a print job.

sp.config Parameters (Continued)

Mounting a Form on a Printer

To mount and align a form:

1. Choose **Queue management** from the Spooler menu to mount a form on all printers. Choose **printer groUp queue management** to mount a form on a printer belonging to a printer group.
2. Choose **Mount form**.
3. Enter the name of the printer.
4. Enter the name of the form you want to mount, or press **ENTER** to retain the form that is currently mounted on the printer.

You are asked if you want to check the vertical and horizontal alignment. If you enter **Y**, you are prompted to enter the path of the UNIX file you want to print to test the alignment of the printer. If you press **ENTER**, the next queued job is printed as an alignment test. You can also specify the number of lines you want printed.

To display the current spooler status report, press **F10** to activate the menu bar, then choose **Action**, then choose **List**.

Configuring the Spooler

To display the information contained in the *sp.config* file, choose **Spooler** from the System Administration menu, then choose **Device management**, then choose **List printers**. Select the printer whose configuration you want to display by highlighting it and pressing **ENTER**. Choose **Yes** when asked if you want to modify the printer. The printer details are listed, and you can modify them if you want. To exit without making changes, press **Esc**.

To make changes to the *sp.config* file, use either of the following menu options:

- Choose **Spooler** from the System Administration menu, then choose **Device management**, then choose **Maintain devices**.
- Choose **Devices** from the System Administration menu, then choose **maintain Devices**.

Changes you make will not take effect until you do one of the following:

- Reread the spooler configuration
- Reset the spooler daemon

Rereading the Spooler Configuration

To reread the configuration without resetting the spooler daemon, choose **Spooler** from the System Administration menu, then choose **Device management**, then choose **Reread configuration**. Rereading the configuration does not reread the *usplog* file. This means that any temporary changes you made to the spooler are lost.

Resetting the Spooler Daemon

To reset the spooler daemon, choose **Spooler** from the System Administration menu, then choose **sPooler management**, then choose **Reset spooler**. Resetting the spooler daemon rereads both the *sp.config* file and the *usplog* file; any temporary changes you made to the spooler are restored.

If a job is actively printing when you reset the daemon, it is put in the suspended state. To reset or terminate the spooler gracefully, first disable printing on all printers, then wait until all active jobs have finished printing. Disable printing on all printers by choosing **Spooler** from the System Administration menu, then choose **Queue management**, then choose **Halt printing**. To disable printing on printers in a printer group, choose **Spooler**, then choose **printer groUp queue management**, then choose **Halt printing**.

Changing the Spooler Configuration

You can change information the spooler daemon uses to communicate with the system. You do this from the System Administration menus. You can move the spooler home directory to a partition with more space, change the path or file name of the logging files, change the order in which jobs are printed, or change the amount of time the spooler daemon waits for the system to respond. You can also specify these options with the *usd* command. For more information about *usd*, see *UniVerse User Reference*.

Choose **Spooler** from the System Administration menu, choose **sPooler management**, then choose **Configure spooler**. The following example shows the Spooler Configuration screen.

Spooler Configuration	
File	Help
Spool Directory	? /usr/spool/uv
Error Log File	: /usr/spool/uv
Activity Log File	: /usr/spool/uv
Logging	: OFF
Chronological Order	: OFF
Timer Value	: 10

Help Region

Press <F1> for longer help about any particular entry, or <ESCAPE> to exit program.
Enter the pathname where the spooler is to be started from.

The following is an explanation of the parameters:

Spool Directory. Type the path of the directory in which spooled files are created. The spooler configuration files, *sp.config* and *usplog*, are in this directory. The default directory is the value of the configurable parameter UVSPool.

Error Log File. Type the name of the file used by the spooler daemon to report errors. If the path is a directory, the spooler uses the default file name, *err.log*. The default path is the spool directory.

Activity Log File. Type the name of the file in which activity messages are stored. If the path is a directory, (such as, */usr/spool/uv*) the spooler uses the default filename, *act.log*. The default path is the spool directory.

Logging. Type **ON** to enable error and activity logging. The default is **OFF**.

Chronological Order. Type **ON** to print jobs in first in first out order. The default is **OFF**.

Timer Value. Type the number of seconds the spooler waits for the system to respond to commands. The default is 10 seconds.

After you enter the parameters, press **F10** to activate the menu bar. Type **F** to choose the File menu. Type **S** to save changes to the *uv.rc* script. You are prompted to restart the spooler. If you want the changes to take effect immediately, choose **YES**.

Type **E** to exit the Spooler Configuration screen without any changes.

Maintaining Printer Groups

To define a printer group, choose **Spooler** from the System Administration menu, then choose **printer Group management**. When you add, change, or delete a printer group definition, UniVerse updates the *print_group* file.

Adding a Printer Group

To add a printer group, follow these steps:

1. Choose **Spooler** from the System Administration menu.
2. Choose **printer Group management**.
3. Choose **printer Groups**, then choose **Add a group**.
4. At the prompts, enter the name of a new printer group.
5. Enter the login names of users who are to have access to this printer group. Enter **a11** to specify that all users can access the printer group. To see a list of all users currently defined in the */etc/passwd* file, press **F4** or enter an asterisk (*) at the prompt.
6. Enter the names of the printers that are to be included in the printer group, or enter **a11** to include all printers in the printer group. To see a list of all currently defined printers on the system, press **F4** or enter an asterisk (*) at the prompt. Choose **Yes** to add the new printer group.

When you save the new printer group information, UniVerse updates the *print_group* file and clears the data entry screen. If you do not define another printer group, you are returned to the System Administration menu.

Changing a Printer Group

To change an existing printer group, follow these steps:

1. Choose **printer Groups** from the Printer Group menu.
2. Choose **Add a group**. To make changes, choose the printer group you want to change, use the arrow keys to move to the fields you want to change, then make the changes you want.

Deleting a Printer Group

To delete a printer group from the *print_group* file, follow these steps:

1. Choose **printer Groups** from the Printer Group menu.
2. Choose **Delete a group**. Choose the printer group you want to delete.

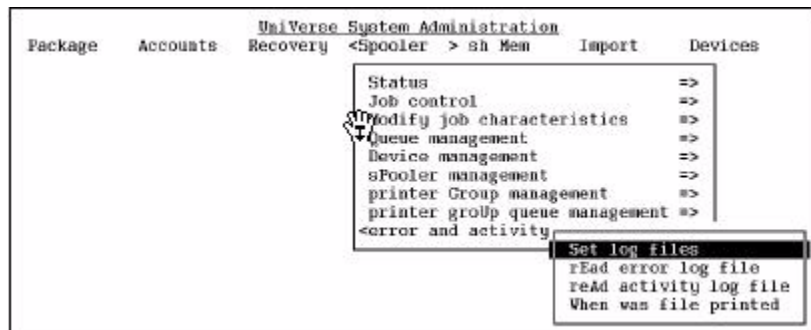
When you save the revised printer group information, or when you are finished deleting printer groups, UniVerse updates the *print_group* file. You are then returned to the System Administration menu.

Maintaining Printer Group Users and Printers

You can also add, change, and delete users and printers from the *print_group* file. To add, change, or delete users, choose **Users** from the Printer Group menu. To add, change, or delete printers, choose **Printers** from the Printer Group menu.

Spooler Log Files

Use the Spooler menu to create and use the spooler log files. The Error and Activity Logs menu looks like the following example.



Creating Spooler Log Files

To create spooler log files, choose **Spooler** from the System Administration menu, then choose **error and activity Logs**, then choose **Set log files**. At the prompts, enter the UNIX pathnames for the files you want to create or press **ENTER** to accept the default log files.

Displaying Spooler Log Files

To display the contents of the spooler log files, choose **Spooler** from the System Administration menu, then choose **error and activity Logs**, then choose either **rEad error log file** or **reAd activity log file**.

The activity log file displays information about the spooler (S), the spooler queues (Q), printers (P), and print jobs (J). Each piece of information is displayed on a separate screen (or series of screens). The subject of the information is indicated by the letter in the first column of the report.

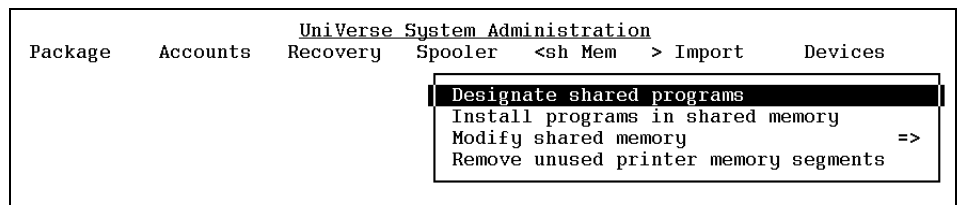
When you choose **reAd activity log file**, spooler information is displayed (indicated by the letter S). Press **ENTER** to display queue information, printer information, and print job information. Each report lists the action performed and the date and time it was logged.

You can use the Spooler Log menu to determine when a specified print job was printed. Choose **Spooler** from the System Administration menu, then choose **error and activity Logs**, then choose **When was file printed**. At the prompt, enter either the UNIX file name that was printed, the print job ID number (the number that was displayed when the print job was generated), or the name of the user who sent the job to the printer. In the latter case, all jobs sent to the printer by that user will be displayed.

Shared Memory (sh Mem) Option

Use the Shared Memory (**sh Mem**) option to load BASIC programs into catalog shared memory. Catalog shared memory reduces the amount of memory required for multiple users to run the same program concurrently.

When you choose **sh Mem** from the UniVerse System Administration menu, the Shared Memory menu appears as shown in the following example.



The Shared Memory Menu

Designating Programs for Catalog Shared Memory

To designate BASIC programs to run in catalog shared memory, choose **Designate shared programs** from the Shared Memory menu. The Program Selection screen appears.

This screen displays 22 programs at a time from the *catdir* file. The following example shows an example of programs listed on the Program Selection screen. If there are more programs than can fit on the screen, a plus sign (+) is displayed next to the last program on the screen. This screen also displays the total size of programs designated to run in catalog shared memory and the amount of space available.

Select Shared BASIC Programs			
File	Action	Help	
1 IADDS	12 ICOUNTS		
2 IAMLC	13 IDISLEN		
3 IANDS	14 IDIVH		
4 IASYN	15 IDIVS		
5 IBINARY.CONVERT	16 IEDIT.INPUT		
6 IBPLOC	17 IEQS		
7 ICATS	18 IERRNO		
8 ICHARS	19 IEXIST		
9 ICHECK.TYPE1.ID	20 IFADD		
10 ICLEAR.PROMPTS	21 IFCMP		
11 ICOMO	+22 IFDIV		
Bytes used: 1,016,665	Bytes available:	9,469,095	
Enter Choice ?	Current File	catdir	
Help Region			
Enter tag number, <A> (tag all), <C> (change file), <L> (list chosen programs), <S> (save and exit), or Page Up/Page Down to scroll screen.			

The system prompts you to designate programs by entering the following:

- The number appearing in front of a program name.
- **A** to designate all the programs in the file. This may take several minutes if there are many programs in the file.
- **L** to list the programs currently designated to run.
- **C** to change to another program file either in the UV account or in some other account. Programs in the file you specify are listed.

The Program Selection screen first displays normally and globally cataloged programs, but any BASIC program can be loaded into catalog shared memory.

To see a list of the programs already selected, type **L**. This displays the account, file, record name, and the number of bytes this program will take up in catalog shared memory, as shown in the following example.

List Chosen BASIC Programs				
File		Action		Help
ACCOUNT	FILE	RECORD	SIZE	
1 UV	: catdir	: !ADDS	: 508	
2 UV	: catdir	: !AMLC	: 3299	
3 UV	: catdir	: !ANDS	: 508	
4 UV	: catdir	: !ASVNC	: 3299	
5 UV	: catdir	: !BINARY.CONVERT	: 5666	
6 UV	: catdir	: !BPIOCP	: 600	
7 UV	: catdir	: !CATS	: 508	
8 UV	: catdir	: !CHARS	: 506	
9 UV	: catdir	: !CHECK.TYPE1.ID	: 1536	
+10 UV	: catdir	: !CLEAR.PROMPTS	: 502	
Bytes used: 1,016,665		Bytes available: 9,469,095		
Enter Choice ? █		Current File		catdir
Help Region				
Enter field number to delete, <D> (delete all), <C> (change file), <E> (exit), <S> to save results and exit, or <X> to return to previous menu, or Page Up/Page Down to scroll screen.				

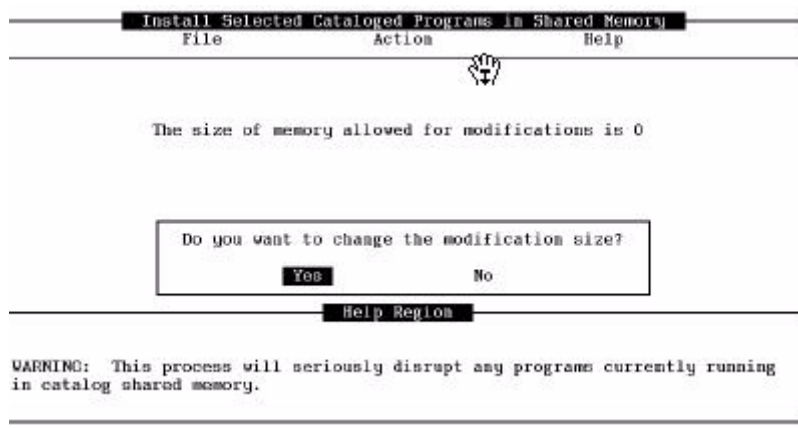
To change to a different file of BASIC programs, type **C**. You are prompted to enter the account name and file you want displayed. The account name can be the name of a UniVerse account (for example, UV) or a UNIX path (such as /u1/uv/BP). The records in the file are displayed 22 at a time.

To remove a program from the list, enter the number of the program and answer **N** to the question Should this program run in shared memory?.

To remove all programs from the list, type **A**.

Installing Programs into Catalog Shared Memory

To load programs into shared memory, choose **Install programs in shared memory** from the Shared Memory menu. The screen shown in the following example appears.



You are asked if you want to change the amount of memory allocated for modifying shared memory. You might want to allocate extra space if all you want to do is change a few programs in catalog shared memory without completely reloading it. Most users will not need extra space and should answer **N**.

You are then prompted with Continue to load shared memory (Y/N)?. Answer **Y** to load your designated programs into catalog shared memory. Any other response stops the load. UniVerse lists the name of each program as it is loads and gives an error message for any program it cannot find and for any records that are not compiled BASIC programs.

You can allocate extra space in the catalog memory segment at startup time by modifying the *uv.rc* script. To increase the segment size 8192 bytes, change the following line in *uv.rc*:

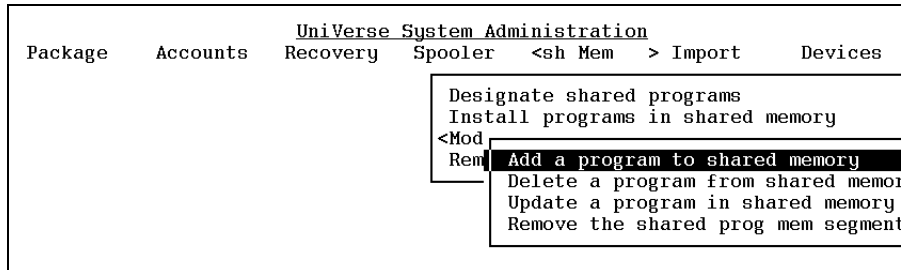
```
bin/load_shm_cat 2>&1 > /dev/null
```

to:

```
bin/load_shm_cat 8192 2>&1 > /dev/null
```

Modifying Catalog Shared Memory

When you change and recompile a BASIC program, the changes are not applied to the copy of the program in shared memory. To update the copy in shared memory, choose **Modify shared memory** from the Shared Memory menu. This choice displays four modification options as shown in the following example.



The Modify Shared Memory Menu

Add a program to shared memory lets you add a new program to catalog shared memory. Anyone using the program continues to use it, but everyone who starts the program after it is loaded uses the copy in catalog shared memory.

Delete a program from shared memory deletes a program from catalog shared memory. This option prompts you to choose a program by asking for an account, file, and record. If you respond **Y** to the Should program be deleted from shared memory prompt, the program is removed. Anyone using the program when it is deleted continues to use it. Anyone who starts using the program after it is deleted gets a copy from disk.

Update a program in shared memory updates a program already loaded into catalog shared memory. Again, anyone using the program continues to use the old copy, but anyone who starts the program after shared memory is modified gets the new version.

Remove the shared prog mem segment removes the shared memory segment. Anyone attached to the shared memory segment continues to use it (they must leave UniVerse and reenter it to detach). Anyone who enters UniVerse after the shared memory segment is removed gets all programs from disk files.

If you try to add or update a program in catalog shared memory when there is not sufficient space, you get a message that you have run out of space. The add or update is aborted. To do the add or update, you have to reload the entire segment.

Removing Printer Memory Segments

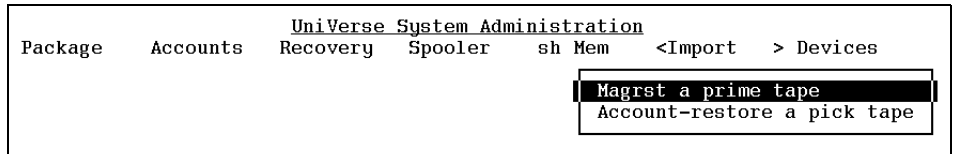
Occasionally after an abnormal termination of UniVerse, printer shared memory segments are not automatically removed. To remove such obsolete printer memory segments:

1. Choose **Remove the shared prog mem segment** from the Shared Memory menu.
2. Choose **Yes** at the Are you sure you want to continue? prompt.

Import Option

Use the **Import** option to import a Pick or Prime INFORMATION account from a remote system or to restore a Pick or Prime account from tape.

When you choose **Import** from the UniVerse System Administration menu, the Import menu appears as shown in the following example.



The Import Menu

Restoring Non-UniVerse Accounts from Tape

To restore or transfer an account, a UniVerse account or a UNIX directory must exist to receive the restored or transferred account. The UniVerse account or UNIX directory will be the *parent directory* of the restored or transferred account. If the parent directory does not exist, you must create it before you restore or transfer an account to UniVerse. You can create a UniVerse account using the **Accounts** option from the System Administration menu.

The steps are as follows:

1. Choose **Import** from the System Administration menu. The Import menu appears.
2. Choose **Magrst a prime tape** to restore Prime INFORMATION tapes saved with the MAGSAV command. Choose **Account-restore a pick tape** to restore Pick account tapes saved with the ACCOUNT-SAVE command.

3. At the MAGRST screen or ACCOUNT-RESTORE screen prompt, enter either the name of the UniVerse account or the path of the UNIX directory under which you want to restore the account from the tape.

The account is restored in a subdirectory of the existing UniVerse account's directory. If you specify a UniVerse account name, it must exist in the UV.ACCOUNT file. To see a list of valid UniVerse account names, enter an asterisk (*) at the prompt. Be sure to enter the UniVerse account name in uppercase letters, where applicable.

You can also enter a user name or user ID number. In this case the account is restored in a subdirectory of that user's home directory.

4. UniVerse prompts you to mount your MAGSAV or ACCOUNT-SAVE tape on the specified device.
5. Press **ENTER**. If the tape is properly mounted on the drive, a subdirectory is created with the name of the account on tape, it is made a UniVerse account, and all the files in the account on tape are restored to it. A list of the file names appears on your screen while the restoration is in progress. If the directory under which you want to restore the account on tape does not exist, an error message appears.

Restoring a Prime INFORMATION Account

This section describes how to restore Prime INFORMATION accounts that have been saved on tape using the MAGSAV command. The example shows how to use the System Administration menus to restore a Prime account, named INFOACCT, to */u1/accts/SALES*.

1. Choose **Import** from the System Administration menu. The Import menu appears.
2. Choose **Magrst a prime tape**. The MAGRST screen appears:

MAGRST a Prime Information Tape		
File	Action	Help
<p>Parent Directory : /u1/accts/SALES No rewind device name : MT0 /dev/rmt0,5 Rewind device name : /dev/rmt0,4 Device type : DC Input block size : 512 Use type 19 files ? <input type="checkbox"/></p>		
		Help Region
Use type 19 files		

3. The following text describes information you can enter in the fields on this screen:

Parent Directory. The directory that is to contain the restored account. You can specify the parent directory either as a UNIX path or as a UniVerse account defined in the UV.ACCOUNT file. You can also specify a user login name or ID number. The account is restored in the home directory of that user.

If you specify a UniVerse account, the account on tape is restored as a *sub-directory* in the specified UniVerse account's *parent directory*. Do not specify the name of the account you are restoring as the parent directory.

In the previous example, the INFOACCT account is restored as a subdirectory under the parent directory */u1/accts*; the restored account's directory is */u1/accts/SALES*.

No rewind device name. A valid entry in the &DEVICE& file, (such as MT0) or a valid UNIX device or file name (such as */dev/rmt12*). It specifies the path to use for a tape device that does not rewind when finished. To display a list of all devices defined in the &DEVICE& file, enter an * at the prompt.

Rewind device name. A valid entry in the &DEVICE& file (such as MT0) or a valid UNIX device or file name (such as */dev/rmt8*). It specifies the path to use for a tape device that rewinds when finished. To display a list of all devices defined in the &DEVICE& file, enter an * at the prompt.

Device type. One of the following:

F Diskette

DT Default nine-track tape (1/2-inch tape)

DC Default cartridge tape (1/4-inch tape)

Input block size. If the device name is found in the &DEVICE& file, the input block size is taken from field 16 of the &DEVICE& file. If field 16 of the &DEVICE& file is blank, a default of 8192 is used for 1/2-inch tape devices, and a default of 512 is used for cartridge tape devices. If the device name is not found in the &DEVICE& file, a default of 512 is used. This field is not used for diskettes.

Use type 19 files. The default for this is **N**, which creates type 1 files. A **Y** specifies type 19 files instead of type 1 files. For more information about type 1 and type 19 files, see *UniVerse System Description*.

If the device name is not found in the &DEVICE& file, a warning appears and you are prompted to verify that the defaults for input block size and

header files to skip are correct.

Restoring a Pick or REALITY Account

This section shows how to use the ACCOUNT-SAVE command to restore accounts that have been saved on tape using the ACCOUNT-SAVE command. The example shows how to use the System Administration menu to restore a Pick account named SALES to */u1/accts*.

The steps are as follows:

1. Choose **Import** from the System Administration menu. The Import menu appears.
2. Choose **Account-restore a pick tape**. The ACCT.RESTORE screen appears:

ACCT.RESTORE a Pick Tape

File	Action	Help
Parent Directory : /u1/accts/SALES		
No rewind device name : MT0 /dev/rmt0.5		
Rewind device name : /dev/rmt0.4		
Device type : DC		
Input block size : 512		
Pick or Reality tape : P		
Use type 19 files : N		

Help Region

Do you wish to go back and change any of your answers?

YesNo

3. The following text describes the information you can enter in the fields on this screen:

Parent Directory. The directory that is to contain the restored account. You can specify the parent directory either as a UNIX path or as a UniVerse account defined in the UV.ACCOUNT file. You can also specify a user login name or ID number. The account will be restored in the home directory of that user.

If you specify a UniVerse account, it is restored as a *subdirectory* in the specified UniVerse account's *parent directory*. Do not specify the name of the account you are restoring as the parent directory.

In the previous example, the PICKACCT account is restored as a subdirectory under the parent directory */u1/accts*. The restored account's directory is */u1/accts/SALES*.

No rewind device name. A valid entry in the &DEVICE& file, (such as MT0) or a valid UNIX device or file name (such as */dev/rmt12*). It specifies the path to use for a tape device that does not rewind when closed. To display a list of all devices defined in the &DEVICE& file, enter an * at the prompt.

Rewind device name. A valid entry in the &DEVICE& file (such as MT0) or a valid UNIX device or file name (such as */dev/rmt8*). It specifies the path to use for a tape device that rewinds when closed. To display a list of all devices defined in the &DEVICE& file, enter an * at the prompt.

Device type. One of the following:

F Diskette

DT Default nine-track tape (1/2-inch tape)

DC Default cartridge tape (1/4-inch tape)

Input block size. If the device name is found in the &DEVICE& file, the input block size will be taken from field 16 of the &DEVICE& file. If field 16 of the &DEVICE& is blank, a default of 8192 is used for 1/2-inch tape devices, and a default of 512 is used for cartridge tape devices. If the device name is not found in the &DEVICE& file, a default of 512 is used. This field is not used for diskettes.

For 1/2-inch tape drives, which read physical tape records, it is important that the input block size be large enough to accommodate the largest physical record on the tape, otherwise data will be lost. Note that some tapes may require a block size larger than 8192.

Pick or Reality data tape. Specify an **M** (Microdata) for an ACCOUNT-

SAVE created on a REALITY machine, and a **P** for all others.

Use type 19 files. The default for this is **N** which creates type 1 files. A **Y** specifies type 19 files instead of type 1 files.

If the device name is not found in the **&DEVICE&** file, the system displays a warning and prompts you to verify that the defaults for input block size, label records, and header files to skip are correct.

The system administration routines execute the UniVerse program *tapein* to restore multireel ACCOUNT-SAVE and MAGSAV tapes.

Running T.LOAD from a File. Before running T.LOAD, you must set up an **&DEVICE&** entry to access the *pqic.results* file.

1. Choose **Devices** from the System Administration menu, then choose **maintain Devices**.
2. Enter a new name in the **Device Name** field, for example, MTFILE. You can specify any user-defined name as a temporary name for the device.
3. Press **ENTER** to skip to **Device Type** and enter **O** for Other.
4. Enter the full path of the *pqic.results* file for **Device Pathname (no rewind)** and for **Device Pathname (rewind)**.
5. Press **ENTER** to skip the remaining fields on the screen.
6. Save the device entry when prompted.
7. Exit the System Administration menu using the **Esc** key.

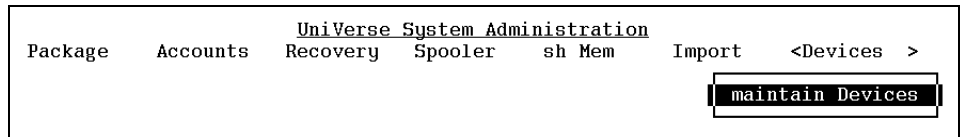
LONGNAMES Mode

If you restore a file through the System Administration menus, make sure that LONGNAMES has the same value on both systems or filenames may get truncated. For more information, see [Support for Long Filenames on UNIX Systems](#) in Chapter 7, “[Transferring Accounts](#).”

Devices Option

Use the **Devices** option to update and maintain the UniVerse file &DEVICE& and the UNIX file *sp.config*.

When you choose **Devices** from the UniVerse System Administration menu, the Devices menu appears, as shown in the following example. When you choose **maintain Devices**, a data entry screen (called Maintain Devices) for the &DEVICE& file appears.



The Devices Menu

Updating the &DEVICE& File

To update the &DEVICE& file:

1. Choose the **Devices** option from the System Administration menu. The Devices menu appears.
2. Choose **maintain Devices**. The Maintain Devices screen appears:

Maintain Devices		
File	Action	Help
Device Name : <input type="text"/> Description : <input type="text"/> Device Type : <input type="text"/> Block Size : <input type="text"/> Device Pathname (norewind) : <input type="text"/> Device Pathname (rewind) : <input type="text"/> Lock File : <input type="text"/> Backup Shell Command : <input type="text"/> Restore Shell Command : <input type="text"/> Skip Shell Command : <input type="text"/> Rewind Shell Command : <input type="text"/> Offline Shell Command : <input type="text"/> Account Transfer Block Size : 8192		
Help Region field, Down arrow to go down a field, and <F1> for longer help about each prompt. Enter the UniVerse device name that the entry is to be known by, or <F4> for a list of all currently configured devices. Press <F10> to activate the Menu Bar, <ESC> to Abort, Up Arrow to go up a		

- Enter the logical device name of either a tape device or a printer at the Device Name prompt. The logical device name is the name used in various UniVerse commands, such as ASSIGN, to refer to the device. For example, you might enter **MT0** for a tape drive, or you might enter **LW** for a printer.
- Enter a brief description of the device.
- Enter the type of peripheral device. For tape drives you can enter any of the following:

Code	Description
F	Diskette.
DC	Default cartridge tape. Enter DC if you want to run the tape device testing program.
DT	Default nine-track tape. Enter DT if you want to run the tape device testing program.
C	Cartridge tape.
T	Nine-track tape.
P	Printer.
O	A device other than a printer or tape drive.

Tape Drive Codes



Later sections describe the remaining information you need to enter to define a tape device, and how to define a printer.

Note: Multireel tape handling for the UniVerse *T.DUMP* and *T.LOAD* commands is supported only for device types *DC*, *DT*, and *F*.

Defining a Tape Device

If you enter either **DC** or **DT** as the device type, you are asked if you want to run the tape device testing program to determine the following:

- Where to allow the tape mode to change from read to write mode
- What action to take when a tape file that is opened for read is closed
- If a second read call at the end-of-file should return the end-of-file condition again

If you run the tests, their results are automatically filled in for you. If you do not run the tests, you can fill in your own values for these fields.

At the prompts, enter the following information:

Block Size (*Field 3*) This is needed only if the device is for cartridge tape (types *DC* and *DT*) or diskette (type *F*).

For diskettes the default block size is 500. Do not change this setting; any other block size can cause problems.

For cartridge tape the block size should be a multiple of 512.

For nine-track tape (types *T* and *DT*) there is no default block size for *IDEAL* and *INFORMATION* flavor accounts: tape records are read or written with variable length. If field 3 of the *&DEVICE&* file is empty, the default block size for *PICK* or *REALITY* flavor accounts is 8192.

If the device was assigned using the *ASSIGN* command, the default block size is taken from field 3 of the *&DEVICE&* file.

If the device was assigned using the *T.ATT* command, the default block size is taken from field 3 of the *&DEVICE&* file. If that is empty, the default block size is taken from the *VOC* entry for the *T.ATT* command.

Device Pathname (*Fields 2, 6, 7*) For tapes, the path can be different depending on whether the tape drive is a rewind or a no-rewind device. A no-rewind tape drive that does not rewind when closed. A rewind tape drive rewinds when closed. For example, you might enter a path such as `/dev/rmt0` (a rewind device), or `/dev/rmt0n` (a no-rewind device). It is important that you assign the correct access permissions to the device. You can set permissions for a device with the UNIX `chmod(1)` command.

Lock files (*Field 5*) When UniVerse and UNIX system processes share a device, it needs a special lock file that coordinates access to the device when more than one process tries to access it. Field 5 of the `&DEVICE&` file contains the UNIX paths used to implement the locking protocol used by the UniVerse spooler and UNIX facilities such as the spooler and `uucp`. This field is usually empty for tape devices, but can be used to display ownership information. For information about the form of the lock file name for a system, see the UNIX reference manual for the process that is sharing the device.

Backup Shell Command (*Field 8*) The shell command sequence used to back up files to the device.

Restore Shell Command (*Field 9*) The shell command sequence used to restore files from the device.

Skip Shell Command (*Field 10*) The shell command sequence used to move forward one logical tape block on the device.

Rewind Shell Command (*Field 11*) The shell command sequence used to rewind the tape.

Offline Shell Command (*Field 12*) The shell command sequence used to take the device offline.

If you entered either **DC** or **DT** as the device type, and you did not run the tape device testing program, fill in the following three fields:

Read-to-Write Mode Position (Field 13) This specifies where on the tape a change from read mode to write mode is allowed. Enter one of the following:

Mode	Description
L	Write must begin at load point. This is the default.
E	Write can begin at load point or after end-of-file.
A	Write can begin anywhere on the tape. This usually works only on 1/2-inch tapes.

Read-to-Write Mode Positions

Close on Read (Field 14) This specifies what action is taken at the close of a tape file which was opened for read. Enter one of the following:

Option	Description
Y	Tape moves forward to the beginning of next file on close. This is the default value. Use Y for most 1/4-inch tape devices.
N	Tape does not move forward on close. Use N for most 1/2-inch tape devices.

Close on Read Options

Multiple Read at End-of-File Status (Field 15) This specifies whether a second read call at the end-of-file returns the end-of-file condition again. Enter one of the following:

Option	Description
Y	The second read call at the end-of-file returns end-of-file indication again. This is the default value. Use Y for most 1/4-inch tape drives.
N	The second read call at the end-of-file returns the first record of the next file. Use N for most 1/2-inch tape drives.

Multiple Read at End-of-File Options

Note: Most Berkeley device drivers work with F13–F15 set at A,N,N or E,N,N. Most System V device drivers work with E,Y,Y or L,Y,Y.

Account Transfer Block Size (Field 16) This specifies the input block size for use with the account transfer functions described in Chapter 7, “[Transferring Accounts](#).” The default is 8192.



Delete Flag (*Field 17*) This specifies if this device is included in the rotating file pool when it is opened. Enter one of the following:

Option	Description
Y	Opened device is not included in the file pool.
N	Opened device is included in the rotating file pool.

Delete Flag Options

NLS Map Name (*Field 19*) This specifies the name of a character set map for the device. For more information about maps , see *UniVerse NLS Guide*.

O_NDELAY (*Field 20*) If this field contains a Y (or y) when used with the BASIC OPENDEV statement, the file is opened with the O_NDELAY flag set at the UNIX-level open. Otherwise, the O_NDELAY flag is not set. For more information, see your UNIX documentation for *open(2)*.

Defining a Printer

When you enter **P** as the Device Type, the data entry screen changes. Figure shows the data entry screen for defining printers.

Maintain Devices			
File	Action		Help
Printer Name	: LPO	Baud Rate	: 9600
Pathname	: ?	Parity	: None
Driver	:	CR Mode	: None
Form	:	Tab Expansion	: ON
Flow control	: XOFF STARTANY OFF	FF Delay	: No formfeed
Enable Printing	: Y	LF Delay	: None
Enable Queuing	: Y	Word Length	: 8
Lock file 1	:		
Lock file 2	:		
Other Options	:		

Help Region
Press <F10> to activate the Menu Bar, <ESC> to Abort, Up Arrow to go up a field, Down arrow to go down a field, and <F1> for longer help about each prompt. Enter the Pathname of the device to print to.

Defining Printers

To define printers, choose **Spooler** from the System Administration menu, choose **Device management**, then choose **Maintain devices**. The data entry screen for the *sp.config* file prompts you to enter the following information:

Printer Name	:	Baud Rate	:
Unix Pathname	:	Parity	:
Driver	:	CR Mode	:
Form	:	Tab Expansion	:
Flow Control	:	FF Delay	:
Enable Printing	:	LF Delay	:
Enable Queuing	:	Word Length	:
Lock file 1	:	Map Name	:
Lock file 2	:		
Other Options	:		

The data you enter adds an entry to the `&DEVICE&` file. It also adds a line to the *sp.config* file for each printer. (If you are editing the *sp.config* file directly with *vi* or the UniVerse Editor, you can continue logical lines across physical lines by putting a backslash (\) or an underscore (_) at the end of all physical lines but the last that make up the logical line.)

Here is an explanation of the parameters you can specify:

Parameter	Description
Printer Name	The logical printer name; a unique name that identifies the printer. This name is used in various UniVerse commands to refer to the printer.
Unix Pathname	The UNIX path for the printer. This might be a path such as <i>/dev/lp0</i> . Be sure to assign the correct access permissions to the printer. You can set permissions for a printer with the UNIX <i>chmod(1)</i> command.
Driver	The printer driver is a device-specific post-processor for spooled output. The path of the driver can be up to 22 characters in length. If you need to specify a longer pathname, edit the <i>sp.config</i> file. The driver can be a UNIX executable or a shell script. When the driver is invoked, standard input is taken from the print file and standard output is sent to the printer device with characteristics set according to the PTERM (UNIX) options specified for that printer. In addition five command line arguments can be specified. Typical uses of the driver are to execute device-specific formatting filters such as a PostScript filter, or to redirect printing using a <i>uux</i> command.
Form	The name of the default form to be mounted on the printer. The name of the form can be up to 32 characters long. To print the file, you must specify the exact form name when queuing a spool file or when changing the mounted form or the form requested. Form names are case-sensitive. You can mount another form on the printer at any time, or you can use the <i>usa</i> command to mount multiple forms on the printer.
Flow Control	<p>The following types of flow control are available:</p> <p>none – Cancels any DTR or XON/XOFF flow control handshaking that may be set.</p> <p>DTR – Enables Data Terminal Ready handshaking. Loss of DTR is treated the same as the input of the stop character. If DTR is not enabled, loss of DTR is treated as a hangup, and foreground jobs are terminated.</p> <p>XON/XOFF any – Enables XON/XOFF handshaking. Any character acts the same as the XON character.</p>

Maintain Devices Parameters

Parameter	Description
	XON/XOFF XON starts – Enables XON/XOFF handshaking. The XON character is required to <i>start</i> transmission. On Berkeley systems the XON character is the start character; on System V, XON is always CTRL-Q.
Enable Printing	Enter Y to instruct the spooler daemon to start up with printing enabled for that printer. Enter N to start the spooler with printing disabled. At any time you can enable or disable printing from the Spooler menu.
Enable Queuing	Enter Y to instruct the spooler daemon to start up with queuing enabled for that printer. Enter N to start the spooler with queuing disabled. At any time you can enable or disable queuing from the Spooler menu.
Baud Rate	The following baud rates are supported: 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200
Parity	Parity can be one of the following: NONE EVEN ODD
CR Mode	Carriage return mode, can be any of the following: no conversion – Resets all CR modes: carriage returns and newlines are not converted. convert LF to CR/LF – Converts newline to newline, carriage return on output. convert CR to LF – Converts carriage return to newline on output. no CR@(0,0) – Does not output a carriage return when the cursor is at line 0, column 0.

Maintain Devices Parameters (Continued)

Parameter	Description
Tab Expansion	Enter ON or OFF . If tab expansion is ON, a tab character is expanded to the proper number of spaces on output. Tab stops are set every eight columns. If tab expansion is OFF, a tab character is unchanged on output. Some terminals (such as ADDS Viewpoint) use a tab character as part of the cursor movement function. On these terminals tab expansion must be OFF for cursor movement to work properly.
FF Delay	<p>Enter * to display a list of possible formfeed delays, then choose one of the following:</p> <p>no form feeds – Clear-screens are sent to the terminal, but no formfeeds are sent to the line printer.</p> <p>no delay – Clear-screens are sent to the terminal, and formfeeds are sent to the line printer. Output of a formfeed causes no delay.</p> <p>2-second delay – Clear-screens are sent to the terminal, and formfeeds are sent to the line printer. Output of a formfeed causes a two-second delay.</p>
LF Delay	<p>Enter * to display a list of possible newline delays, then choose one of the following:</p> <p>None – No delay for each newline.</p> <p>.08sec. – A delay of about .08 second occurs after each newline.</p> <p>.10sec. – A delay of about .10 second occurs after each newline.</p> <p>.16sec. – A delay of about .16 second occurs after each newline.</p> <p>.18sec. – A delay of about .18 second occurs after each newline.</p> <p>.26sec. – A delay of about .26 second occurs after each newline.</p> <p>Teletype37 – A delay dependent on the column position occurs after each newline. This mode has been configured for Teletype model 37s.</p> <p>col+.08sec. – A delay dependent on the column position plus about .08 second occurs after each newline.</p> <p>col+.16sec. – A delay dependent on the column position plus about .16 second occurs after each newline.</p>

Maintain Devices Parameters (Continued)

Parameter	Description
Word Length	The number of data bits that make up a word, not including the parity bit. Can be 5, 6, 7, or 8.
Map Name	The name of a map you want to assign to the device. This field appears only if NLS is enabled. For information about maps,, see <i>UniVerse NLS Guide</i> .
Lock files	When UniVerse and UNIX system processes share a device, it needs a special lock file created for it that coordinates access to the device when more than one process tries to access it. Field 5 of the &DEVICE& file contains the UNIX paths used to implement the locking protocol used by the UniVerse spooler and several UNIX facilities such as the spooler and <i>uucp</i> . For information about the form of the lock file name for a system, see the UNIX reference manual for the process that is sharing the device.
Other Options	You can specify any of the UniVerse PTERM options to control the port used to drive the printer. See Appendix B for a complete list of PTERM (UNIX). In addition to the PTERM options, you can also specify NORESET. The NORESET option instructs the spooler daemon not to restore printer device characteristics upon completion of a print job.

Maintain Devices Parameters (Continued)

User Menus

In addition to the UniVerse System Administration menus, UniVerse also provides user menus to support print job administration and application conversion functions.

Users can invoke the menus that control spooler queue functions with the command `PRINT.ADMIN`.

The application conversion functions are supported only in compatibility flavor accounts: `PICK`, `INFORMATION`, `PIOPEN`, `REALITY`, and `IN2`. Users can invoke the Account Conversion menu with the command `CONVERT.ACCOUNT`.

PTERM and *stty* Options

The PTERM options described in the following table apply to UniVerse systems running Berkeley (BSD) and System V (SysV) implementations of the UNIX operating system. The PTERM options described in Table apply to UniVerse systems running on Windows NT systems. The term *Not Supported* means that the option is not available for the system specified.

Option	Setting	Description	stty Equivalent	
			BSD	SysV
BAUD	0	Hangs up the dataset connection.	0	0
	rate	Sets the baud rate of your terminal to <i>rate</i> . Available baud rates are 0, 50, 75, 110, 134, 150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, EXTA, EXTB.	rate	rate
BGSTOP	ON	Causes any job running in the background to stop if it attempts to output to your terminal.	tostop	Not Supported
	OFF	Lets background terminal output to be multiplexed with foreground output.	–tostop	
BREAK	ON	Causes <i>intr</i> , <i>quit</i> , <i>susp</i> , and <i>dsusp</i> to cause a BREAK condition within UniVerse. On BSD this is supported by setting the above keys to their default values.	Special	isig

Option	Setting	Description	stty Equivalent	
			BSD	SysV
BREAK (continued)	OFF	Treats <i>intr</i> , <i>quit</i> , <i>susp</i> , and <i>dsusp</i> as normal input characters. On BSD this is supported by turning off all the above keys.	Special	–isig
	INTR	The Break key is treated the same as the <i>intr</i> key.	Default condition	–ignbrk brkint
	IGNORE	The Break key is ignored, no interrupt is generated.	Not Supported	ignbrk –brkint
	NUL	The Break key is treated as the input of a NUL (ASCII 0) character.		–ignbrk –brkint
BRK	<i>char</i>	<i>brk</i> can be used by special programs that require input to terminate on a character other than a newline. In UniVerse the <i>brk</i> is treated the same as a newline.	brk	Not Supported
	ON	Sets the <i>brk</i> character to Return (Ctrl-M) .		
	OFF	Turns off the <i>brk</i> character.		
BSDELAY	ON	Specifies a delay of about .05 second when a backspace is output.	bs1	bs1
	OFF	Specifies no delay upon output of a backspace.	bs0	bs0
CASE	INVERT	Inverts character case (uppercase to lowercase and vice versa) on input.	Only in UniVerse	Only in UniVerse
	NOINVERT	Character case is input without any conversion.		
	UC-IN	Translates uppercase input to lowercase (which might be inverted to uppercase).	Not Supported	iucL
	UC-OUT	Translates lowercase output to uppercase.		olcu

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
CASE (continued)	LC-IN	Performs no translation of case on input (even though it still might be inverted).	Not Supported	–iucLc
	LC-OUT	Performs no translation of case on output.		–olcuc
	XCASE	Uppercase output is preceded with a backslash (\) to distinguish it from lowercase. This is useful when UC-OUT is set because in this case uppercase and lowercase are printed in uppercase.		xcase
	NOXCASE	No distinction on output between uppercase and lowercase.		–xcase
	UC	Combines UC-IN and UC-OUT. On BSD this option also sets XCASE.	lcase	iucLc olcuc
	LC	Combines LC-IN and LC-OUT. On BSD this option sets NOXCASE.	–lcase	–iucLc –olcuc
CRMODE	INLCR	Converts newline to carriage return on input.	Not Supported	inlcr
	NOINLCR	Doesn't convert newline to carriage return on input.		–inlcr
	IGNCR	Ignores carriage return on input.		igncr
	NOIGNCR	Doesn't ignore carriage return on input.		–igncr
	ICRNL	Converts carriage return to newline on input.		icrnl
	NOICRNL	Doesn't convert carriage return to newline on input.		–icrnl
	ONLCR	Converts newline to newline, carriage return on output.		onlcr

Option	Setting	Description	stty Equivalent	
			BSD	SysV
CRMODE (continued)	NOONLCR	Doesn't convert newline to newline, carriage return on output.	Not Supported	–onlcr
	OCRNL	Converts carriage return to newline on output.		ocrnl
	NOOCRNL	Prohibits conversion of carriage return to newline on output.		–ocrnl
	ONOCR	Prohibits output of carriage return when cursor is in column 0.		onocr
	NOONOCR	Outputs carriage return when cursor is in column 0.		–onocr
	ONLRET	Newline performs carriage return function.		onlret
	NOONL-RET	Newline doesn't perform carriage return function.		–onlret
	ON	Sets ICRNL and ONLCR, resets all other values.	nl	Special
	OFF	Resets all CRMODE values.	–nl	
DATABITS	5–8	Changes the number of data bits on the terminal line protocol.	Not Supported	cs5 cs6 cs7 cs8
DSUSP	<i>char</i>	The <i>dsusp</i> (delayed-suspend) character acts like the <i>susp</i> character except that no action is taken until the process actually inputs the character. Thus the <i>dsusp</i> character is a way to type-ahead a <i>susp</i> character.	dsusp	Not Supported
	ON	Sets the <i>dsusp</i> character to Ctrl-Y .		
	OFF	Turns off the <i>dsusp</i> character.		
DTR	ON	Turns on DTR (Data Terminal Ready) handshaking. Loss of DTR is treated the same as the input of the <i>stop</i> character.	mdmbuf	Not Supported
	OFF	Turns off DTR handshaking. Loss of DTR is treated as a hang up, and foreground jobs are terminated.	–mdmbuf	

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
ECHO	ON	Turns the terminal echo on.	echo	echo
	OFF	Turns the terminal echo off.	–echo	–echo
	FAST	Echoes <i>erase</i> as backspace-space-backspace, and echoes <i>kill</i> as a series of backspace-space-backspaces.	crterase crtkill	echoe echok
	MEDIUM	Echoes <i>erase</i> as backspace-space-backspace.	crterase –crtkill	echoe –echok
	SLOW	Echoes <i>erase</i> as a backspace.	–crterase –crtkill	–echoe –echok
	PRINTER	<i>erase</i> causes deleted characters to be echoed backwards between \ and /.	prterase	Only in UniVerse
	CTRL	Echoes all control characters (less than ASCII 32) in a printable fashion as ^ followed by the appropriate alphabetic character. Echoes Del (ASCII 127) as Ctrl-? .	ctlecho	
	NOCTRL	Echoes all control characters as nonprintable control characters.	–ctlecho	
	LF	Echoes the newline character even when the echo is turned off. This mode is useful for some half-duplex terminals.	Not Supported	echonl
	NOLF	Doesn't echo the newline character when the echo is turned off.	Default condition	–echonl
EOF	<i>char</i>	The <i>eof</i> (end-of-file) character is used to terminate input to many UNIX commands (<i>mail</i> , <i>dc</i> , and others). In UniVerse the <i>eof</i> is treated the same as a newline.	eof	eof
	ON	Sets the <i>eof</i> character to Ctrl-D .		
	OFF	Turns off the <i>eof</i> character.		

Option	Setting	Description	stty Equivalent	
			BSD	SysV
EOL	<i>char</i>	<i>eol</i> is the SysV equivalent of the BSD <i>brk</i> character, its uses are the same, and in UniVerse it is treated the same as a newline.	Not Supported	eol
	ON	Sets the <i>eol</i> character to Return (Ctrl-M) .		
	OFF	Turns off the <i>eol</i> character.		
EOL2	<i>char</i>	<i>eol2</i> is a second <i>eol</i> character.	Not Supported	eol2 Machine dependent
	ON	Sets the <i>eol2</i> character to Esc (Ctrl-[) .		
	OFF	Turns off the <i>eol2</i> character.		
ERASE	<i>char</i>	<i>erase</i> causes the previous character to be deleted from the input.	erase	erase
	ON	Sets the <i>erase</i> character to Backspace (Ctrl-H) .		
	OFF	Turns off the <i>erase</i> character.		
FFDELAY	0	Output is paged to the terminal and the line printer, but the clear-screen is not printed at the beginning of each page on the terminal, and no formfeeds are sent to the line printer.	Only in UniVerse	Only in UniVerse
	1	Clear-screens are sent to the terminal, but no formfeeds are sent to the line printer.		
	2	Clear-screens are sent to the terminal, and formfeeds are sent to the line printer, output of a formfeed causes no delay.	ff0	ff0
	3	Clear-screens are sent to the terminal, and formfeeds are sent to the line printer, output of a formfeed caused a two-second delay.	ff1	ff1

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
FILL	OFF	Specifies that all delays (FFDELAY, LFDELAY, BSDELAY, TABS, VTDELAY) should pause. They should not use fill characters.	Default Condition	–ofill
	ON	Specifies that all delays should use fill characters, the fill character can either be a NUL or a DEL (see below).	Not Supported	ofill
	NUL	Specifies than when delays are using fill characters, the character NUL should be used.	Not Supported	–ofdel
	DEL	Specifies than when delays are using fill characters, the character DEL should be used.		ofdel
FLUSH	<i>char</i>	The <i>flush</i> character stops all output to the terminal. Unlike the <i>stop</i> character, all output is lost. To resume output, another <i>flush</i> character must be input.	flush	Not Supported
	ON	Sets the <i>flush</i> character to Ctrl-O .		
	OFF	Turns off the <i>flush</i> character.		
FMC	<i>char</i>	A literal field mark (ASCII 254) can be entered using the <i>fmc</i> character.	Only in UniVerse	Only in UniVerse
	ON	Sets the <i>fmc</i> character to Ctrl-^ .		
	OFF	Turns off the <i>fmc</i> character.		
INBUFF	ON	Input characters are not transmitted until a carriage return is received. Same as MODE LINE.	cooked –raw	cooked –raw
	OFF	Input characters are transmitted as they are received (raw mode). The difference from raw mode is that for networking the data is not packetized until a carriage return is received. Same as MODE EMULATE.	Only in UniVerse	Only in UniVerse

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
INPUTCTL	ON	Allows input of control characters.	Only in UniVerse	Only in UniVerse
	OFF	Disallows input of control characters.		
	TCL.RESET	Disallows input of control characters until TCL level is reached.		
INTR	<i>char</i>	<i>intr</i> (interrupt) is used to terminate a currently running job. In UniVerse the <i>intr</i> character is treated as a BREAK condition.	intr	intr
	ON	Sets the <i>intr</i> character to DEL (Ctrl-?).		
	OFF	Turns off the <i>intr</i> character.		
KILL	<i>char</i>	The <i>kill</i> character causes the entire input line to be erased.	kill	kill
	ON	Sets the <i>kill</i> character to Ctrl-X .		
	OFF	Turns off the <i>kill</i> character.		
LCONT	<i>char</i>	<i>lcont</i> (line-continue) is a shorthand way of extending an input line at the Command Language prompt. Typing the <i>lcont</i> character is the same as entering an underscore (_) followed by a newline.	Only in UniVerse	Only in UniVerse
	ON	Sets the <i>lcont</i> character to Ctrl-_ .		
	OFF	Turns off the <i>lcont</i> character.		

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
LFDELAY	0	Specifies no delay for each newline.	cr0 nl0	cr0 nl0
	1	A delay of about .08 second occurs after each newline.	cr1 nl0	cr0 nl1
	2	A delay of about .10 second occurs after each newline.	cr0 nl2	cr2 nl0
	3	A delay of about .16 second occurs after each newline.	cr2 nl0	cr3 nl0
	4	A delay of about .18 second occurs after each newline.	cr1 nl2	cr2 nl1
	5	A delay of about .26 second occurs after each newline.	cr2 nl2	cr3 nl1
	6	A delay dependent on the column position occurs after each newline. This mode has been tuned for Tele-type model 37s.	cr0 nl1	cr1 nl0
	7	A delay dependent on the column position + about .08 second occurs after each newline.	cr1 nl1	cr1 nl1
	8	A delay dependent on the column position + about .16 second occurs after each newline.	cr2 nl1	
LITOUT	ON	Outputs characters with normal post-processing.	–litout	opost
	OFF	Outputs characters without postprocessing.	litout	–opost
LNEXT	<i>char</i>	The <i>lnext</i> (literal-next) character causes the next character typed to be entered literally. No input processing occurs. <i>lnext</i> can be used to enter the <i>erase</i> character literally into text. This option has no effect when used in UniVerse.	lnext	Not Supported
	ON	Sets the <i>lnext</i> character to Ctrl-V .		
	OFF	Turns off the <i>lnext</i> character.		

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
MODE	LINE	Input characters are not transmitted until a carriage return is received. Same as INBUFF ON.	cooked -raw	cooked -raw
	RAW	Input characters are transmitted as they are received.	raw	raw
	CHAR	Input characters are transmitted as they are received, except for special characters.	cbreak	Not Supported
	EMULATE	Input characters are transmitted as they are received (raw mode). The difference from raw mode is that for networking the data is not packetized until a carriage return is received. Same as INBUFF OFF.	Only in UniVerse	Only in UniVerse
NOHANG	ON	Causes the loss of DTR to be ignored. Loss of carrier will not terminate a job.	nohan	Not Supported
	OFF	Causes the loss of DTR to be treated as a hang up, and running foreground jobs are terminated.	-nohan	

Option	Setting	Description	stty Equivalent	
			BSD	SysV
PARITY	NONE	Specifies that no parity generation is done for output, and no parity checking is enforced on input.	even odd	–parenb
	EVEN	Even parity is generated for output, and checked for on input (if enabled).	even –odd	parenb –parodd
	ODD	Odd parity is generated for output, and checked for on input (if enabled).	–even odd	parend parodd
	ENABLE	Parity input checking is enabled, provided that the parity mode is not set to NONE.	Default condition	inpck
	DISABLE	Input parity checking is disabled, characters of any parity are allowed.	Not Supported	–inpck
	ERR-IGN	If input parity checking is enabled, errors (characters of the wrong parity) are ignored.		inpar
	ERR-MRK	When input parity checking is enabled, errors are marked by simulating a special input sequence. This mode cannot be used within UniVerse. If set, it acts the same as ERR-IGN.		–ignpar parmrk
	ERR-NUL	When input parity checking is enabled, errors are input as the NUL character.		–ignpar –parmrk
PENDIN	ON	Automatically retypes input and enters an <i>erase</i> character. This mode has no effect within UniVerse.	pendin	
	OFF	Doesn't automatically retype input.	–pendin	
QUIT	<i>char</i>	<i>quit</i> is used to terminate a currently running job. However, a core dump is also produced. In UniVerse the <i>quit</i> character is treated as a BREAK condition.	quit	quit
	ON	Sets the <i>quit</i> character to Ctrl-^ .		
	OFF	Turns off the <i>quit</i> character.		

Option	Setting	Description	stty Equivalent	
			BSD	SysV
RPRNT	<i>char</i>	The <i>rprnt</i> (reprint) character causes the previous line to be redisplayed. This is useful when transmission errors or background output has disturbed the data on the terminal screen.	rprnt	Only in UniVerse
	ON	Sets the <i>rprnt</i> character to Ctrl-R .		
	OFF	Turns off the <i>rprnt</i> character.		
SMC	<i>char</i>	A literal subvalue mark (ASCII 252) can be entered using the <i>smc</i> character.	Only in UniVerse	Only in UniVerse
	ON	Sets the <i>smc</i> character to Ctrl-\\ .		
	OFF	Turns off the <i>smc</i> character.		
SQLNULL	ON	Sets the null value character to Ctrl-N .	Only in UniVerse	Only in UniVerse
	OFF	Turns off the null value character.		
	<i>char</i>	Sets the null value character to <i>char</i> .		
START	<i>char</i>	The counterpart of the <i>stop</i> character, <i>start</i> resumes output after it has been held. If the XON STARTANY option is set, any input character resumes output, and the <i>start</i> character is the only character not entered as data.	start	See XON
	ON	Sets the <i>start</i> character to Ctrl-Q .		
	OFF	Turns off the <i>start</i> character.		
STOP	<i>char</i>	The <i>stop</i> character is used to temporarily stop output to the terminal. Output is resumed by typing the <i>start</i> character (above).	stop	See XON
	ON	Sets the <i>stop</i> character to Ctrl-S .		
	OFF	Turns off the <i>stop</i> character.		
STOPBITS	1	The terminal line protocol is set for 1 stop bit.	Not Supported	–cstopb
	2	The terminal line protocol is set for 2 stop bits.		cstopb

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
STRIP	ON	Strips the eighth bit off input characters.	Not Supported	istrip
	OFF	Doesn't strip the eighth bit off input characters.		–istrip
SUSP	<i>char</i>	The <i>susp</i> (suspend) character immediately causes the current job to stop. In UniVerse this character is treated as a BREAK condition.	susp	Not Supported
	ON	Sets the <i>susp</i> character to Ctrl-Z .		
	OFF	Turns off the <i>susp</i> character.		
SWTCH	<i>char</i>	<i>switch</i> (switch) is used in conjunction with <i>shl</i> to switch terminal input to the layering program (<i>shl</i>).	Not Supported	switch Machine dependent
	ON	Sets the <i>switch</i> character to Ctrl-Z .		
	OFF	Turns off the <i>switch</i> character.		
TABS	ON	Turns tab expansion on. On output, a tab character is expanded to the proper number of spaces. Tab stops are set every 8 columns.	–tabs	tab3
	OFF	Turns tab expansion off: on output, a tab character is unchanged. Some terminals (like the ADDS View-point) use a tab character as a part of the cursor movement function. On these terminals TABS must be set to OFF for cursor movement to work properly.	tabs	tab0
TILDE	ON	Converts ~ (tilde) to ‘ (accent grave) on output.	tilde	Not Supported
	OFF	Does not convert ~ (tilde).	–tilde	
TMC	ON	Sets the text mark character to Ctrl-T .	Only in UniVerse	Only in UniVerse
	OFF	Turns off the text mark character.		
	<i>char</i>	Sets the text mark character to <i>char</i> .		

Option	Setting	Description	<i>stty</i> Equivalent	
			BSD	SysV
VMC	<i>char</i>	A literal value mark (ASCII 253) can be entered using the <i>vmc</i> character.	Only in UniVerse	Only in UniVerse
	ON	Sets the <i>vmc</i> character to Ctrl-] .		
	OFF	Turns off the <i>vmc</i> character.		
VTDELAY	ON	Specifies a two-second delay each time a vertical tab is output.	Not Supported	vt1
	OFF	Specifies no delay time when a vertical tab is output.		vt0

Option	Setting	Description	stty Equivalent	
			BSD	SysV
WERASE	<i>char</i>	The <i>werase</i> (word erase) character causes the previous word (up to but not including a space) to be deleted.	werase	Only in UniVerse
	ON	Sets the <i>werase</i> character to Ctrl-W .		
	OFF	Turns off the <i>werase</i> character.		
XON	ON	Turns on X-ON/X-OFF protocol. When a X-OFF is received by the computer, all transmission stops until an X-ON is received. On BSD the X-OFF character is the <i>stop</i> character and the X-ON is the <i>start</i> character. This option is implemented by setting <i>stop</i> and <i>start</i> to their default values. On SysV, X-OFF is always Ctrl-S , and X-ON is always Ctrl-Q .	Special	ixon
	OFF	Disables the X-ON/X-OFF protocol. The X-OFF and the X-ON character are treated as normal input. On BSD this option is implemented by turning off the <i>stop</i> and <i>start</i> characters.		–ixon
	STARTANY	Causes the receipt of any character to act the same as X-ON, if X-ON/X-OFF is enabled.	–decctq	ixany
	NOSTART-ANY	Requires receiving an X-ON character to restart transmission.	decctq	–ixany
	NOTANDEM	Turns off the automatic X-OFF, X-ON mode described above.	notandem	–ixoff
	TANDEM	Causes the computer, when its input buffer is almost full, to transmit an X-OFF character to the terminal, and when the buffer is almost empty, to transmit an X-ON. This lets the computer communicate with another device or computer.	tandem	ixoff

The next table illustrates PTERM options that now use a different command.

Historical PTERM Options		
Option	Setting	Equivalent
XOFF		XON ON
NOXOFF		XON OFF
FULL		ECHO ON
HALF		ECHO OFF
TYPE	LF	ECHO OFF ECHO LF
	NOLF	ECHO OFF ECHO NOLF
	PRINTER	ECHO PRINTER
	SLOW	ECHO SLOW
	MEDIUM	ECHO MEDIUM
	FAST	ECHO FAST
CTRLECHO	ON	ECHO CTRL
	OFF	ECHO NOCTRL
STARTANY	ON	XON STARTANY
	OFF	XON NOSTARTANY
CRT		ECHO FAST ECHO CTRL
ECHO	DELAY	ECHO ON
	NODELAY	ECHO ON

The following table lists the PTERM options available on Windows NT systems.

Windows NT PTERM Options		
Option	Setting	Description
BAUD	0	Hangs up the dataset connection.
	<i>rate</i>	Sets the baud rate of your terminal to <i>rate</i> . Available baud rates are 0, 50, 75, 110, 134, 150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, EXTA, EXTB.
CASE	INVERT	Inverts character case (uppercase to lowercase and vice versa) on input.
	NOINVERT	Character case is input without any conversion.

Windows NT PTERM Options (Continued)

Option	Setting	Description
CRMODE	ICRNL	Converts carriage return to newline on input.
	NOICRNL	Doesn't convert carriage return to newline on input.
	CRONLY	When carriage return and newline are sent as a pair, both characters are recognized.
	NOCRONLY	When carriage return and newline are sent as a pair, only the carriage return is recognized.
ECHO	ON	Turns the terminal echo on.
	OFF	Turns the terminal echo off.
	CTRL	Echoes all control characters (less than ASCII 32) in a printable fashion as ^ followed by the appropriate alphabetic character. Echoes Del (ASCII 127) as Ctrl-? .
	NOCTRL	Echoes all control characters as nonprintable control characters.
	LF	Echoes the newline character even when the echo is turned off. This mode is useful for some half-duplex terminals.
	NOLF	Doesn't echo the newline character when the echo is turned off.
ERASE	<i>char</i>	<i>erase</i> causes the previous character to be deleted from the input.
	ON	Sets the <i>erase</i> character to Backspace (Ctrl-H) .
	OFF	Turns off the <i>erase</i> character.
FMC	<i>char</i>	A literal field mark (ASCII 254) can be entered using the <i>fmc</i> character.
	ON	Sets the <i>fmc</i> character to Ctrl-^ .
	OFF	Turns off the <i>fmc</i> character.
INBUFF	ON	Input characters are not transmitted until a carriage return is received. Same as MODE LINE.
	OFF	Input characters are transmitted as they are received (raw mode). The difference from raw mode is that for networking the data is not packetized until a carriage return is received. Same as MODE EMULATE.
INPUTCTL	ON	Allows input of control characters.
	OFF	Disallows input of control characters.
	TCL.RESET	Disallows input of control characters until TCL level is reached.

Windows NT PTERM Options (Continued)

Option	Setting	Description
INTR	<i>char</i>	<i>intr</i> (interrupt) is used to terminate a currently running job. In UniVerse the <i>intr</i> character is treated as a BREAK condition.
	ON	Sets the <i>intr</i> character to DEL (Ctrl-?).
	OFF	Turns off the <i>intr</i> character.
KILL	<i>char</i>	The <i>kill</i> character causes the entire input line to be erased.
	ON	Sets the <i>kill</i> character to Ctrl-X .
	OFF	Turns off the <i>kill</i> character.
LCONT	<i>char</i>	<i>lcont</i> (line-continue) is a shorthand way of extending an input line at the Command Language prompt. Typing the <i>lcont</i> character is the same as entering an underscore (_) followed by a newline.
	ON	Sets the <i>lcont</i> character to Ctrl-__ .
	OFF	Turns off the <i>lcont</i> character.
MODE	LINE	Input characters are not transmitted until a carriage return is received. Same as INBUFF ON.
	RAW	Input characters are transmitted as they are received.
	CHAR	Input characters are transmitted as they are received, except for special characters.
	EMULATE	Input characters are transmitted as they are received (raw mode). The difference from raw mode is that for networking the data is not packetized until a carriage return is received. Same as INBUFF OFF.

Windows NT PTERM Options (Continued)

Option	Setting	Description
PARITY	NONE	Specifies that no parity generation is done for output, and no parity checking is enforced on input.
	EVEN	Even parity is generated for output, and checked for on input (if enabled).
	ODD	Odd parity is generated for output, and checked for on input (if enabled).
	ENABLE	Parity input checking is enabled, provided that the parity mode is not set to NONE.
	DISABLE	Input parity checking is disabled, characters of any parity are allowed.
	ERR-MRK	When input parity checking is enabled, errors are marked by simulating a special input sequence. This mode cannot be used within UniVerse. If set, it acts the same as ERR-IGN.
	ERR-NUL	When input parity checking is enabled, errors are input as the NUL character.
RPRNT	<i>char</i>	The <i>rprnt</i> (reprint) character causes the previous line to be redisplayed. This is useful when transmission errors or background output has disturbed the data on the terminal screen.
	ON	Sets the <i>rprnt</i> character to Ctrl-R .
	OFF	Turns off the <i>rprnt</i> character.
SMC	<i>char</i>	A literal subvalue mark (ASCII 252) can be entered using the <i>smc</i> character.
	ON	Sets the <i>smc</i> character to Ctrl-\.
	OFF	Turns off the <i>smc</i> character.
SQLNULL	ON	Sets the null value character to Ctrl-N .
	OFF	Turns off the null value character.
	<i>char</i>	Sets the null value character to <i>char</i> .
STOPBITS	1	The terminal line protocol is set for 1 stop bit.
	2	The terminal line protocol is set for 2 stop bits.
TMC	ON	Sets the text mark character to Ctrl-T .
	OFF	Turns off the text mark character.
	<i>char</i>	Sets the text mark character to <i>char</i> .

Windows NT PTERM Options (Continued)

Option	Setting	Description
VMC	<i>char</i>	A literal value mark (ASCII 253) can be entered using the <i>vmc</i> character.
	ON	Sets the <i>vmc</i> character to Ctrl-].
	OFF	Turns off the <i>vmc</i> character.
WERASE	<i>char</i>	The <i>werase</i> (word erase) character causes the previous word (up to but not including a space) to be deleted.
	ON	Sets the <i>werase</i> character to Ctrl-W.
	OFF	Turns off the <i>werase</i> character.
XON	ON	Turns on X-ON/X-OFF protocol. When a X-OFF is received by the computer, all transmission stops until an X-ON is received. On BSD the X-OFF character is the <i>stop</i> character and the X-ON is the <i>start</i> character. This option is implemented by setting <i>stop</i> and <i>start</i> to their default values. On SysV, X-OFF is always Ctrl-S , and X-ON is always Ctrl-Q .
	OFF	Disables the X-ON/X-OFF protocol. The X-OFF and the X-ON character are treated as normal input. On BSD this option is implemented by turning off the <i>stop</i> and <i>start</i> characters.
	NOTANDEM	Turns off the automatic X-OFF, X-ON mode described above.
	TANDEM	Causes the computer, when its input buffer is almost full, to transmit an X-OFF character to the terminal, and when the buffer is almost empty, to transmit an X-ON. This lets the computer communicate with another device or computer.

terminfo Terminal Capabilities

This appendix contains:

- A list of terminals that UniVerse adds to *terminfo.src*
- Descriptions of the four kinds of terminal capability
- A table of *terminfo* variables with their associated names in *termcap* and in UniVerse

For information about UNIX *terminfo* entries, see the UNIX documentation supplied with your system.

Additional *terminfo* Entries

Listed below are the terminal entries UniVerse adds to the *terminfo* database. Most entries include several common names for the terminal, separated by a vertical bar (|). The last part of each entry is the full terminal name.

In addition, if a terminal emulates another terminal, the name of the emulated terminal is separated by a hyphen (–) from the name of the terminal emulating it.

vp60–regent40|Adds viewpoint 60 (emulating Adds regent 40)

In the example, the first part of the entry up to the vertical line gives the *terminfo* name of the terminal (*vp60*) followed by the name of the terminal it emulates (*regent40*); the second part of the entry, after the vertical line, gives the full descriptive name of the terminal.

```
97801-UV|97808-UV|97801-uv|97808-uv|SIEMENS terminal
a210-adm5|adm5|Ampex 210 (emulating Lear Sigler adm5)
a210-hz1410|hz1410|Ampex 210 (emulating Hazletine 1410)
a210-hz1500|hz1500|Ampex 210 (emulating Hazletine 1500)
a210-qt102|qt102|Ampex 210 (emulating Qume 102)
a210-regent25|Ampex 210 (emulating Adds Regent 25)
a210-tvi910+|tvi910+|Ampex 210 (emulating Televideo 910+)
a210-tvi910|tvi910|Ampex 210 (emulating Televideo 910)
a210-tvi920|tvi920|Ampex 210 (emulating Televideo 920)
a210-tvi925|Ampex 210 (emulating Televideo 925)
a210-vp|Ampex 210 (emulating Adds viewpoint)
a210|Ampex 210
aixterm-m|hft-m|hft|ibm5151|IBM 5151 display (Aixterm Emulator
Monochrome)
aixterm|hft-c|IBM Aixterm Terminal Emulator
ansi|ansi-uv|terminal_6|6|ANSI Standard Terminal
att3b1|unixpc|pc7300|s4|at|At&t 3b1 Computer
att4410|4410|At&t 4410
dumb|paper|unknown|dialup|network|terminal_9|9|Dumb terminal
fr200|Liberty Freedom 200
gt|Convergent Technologies GT
hp|Hewlett Packard
ibm3151|ult3151|IBM 3151 display
ibmpc|at386|at386-m|pc386-uv|at386-uv|386at
uv|terminal_8|8|IBM PC-AT Console
ic16404|ICL 6404CG Colour Video Display
in9400-uv-j|in2 terminal with prom J
in9400-uv|in2 terminal with prom K and more
IN|insight terminal
mic5510|Microterm 5510
Mu|sun|Sun Microsystems Workstation console
owl|fox|terminal_4|4|Prime Computer owl/fox terminal
pst100|terminal_5|5|Prime Computer pst100 terminal
```

```

pt200c|terminal_10|10|Prime Computer pt200c
pt200|pt250|terminal_7|7|fenix|performer|Prime Computer
pt200/pt250
pt45|terminal_3|3|Prime Computer pt45
pt|Convergent Technologies PT
regent20|Adds regent 20
regent25|Adds regent 25
regent40|pt25|terminal_1|1|Adds Regent 40/Prime Computer PT25
regent60|Adds regent 60
sun-w|Sun Microsystems Workstation console (132 columns)
tab15|tk4010
tv924|TeleVideo 924
tv925|tvi925|terminal_2|2|Televideo 925
tv950|Televideo 950
tv970|Televideo 970
tvi955|Televideo 955
uviterm-vw|132x40 aiXterm for UniVerse(variable font)
uviterm-v|80x40 aiXterm for UniVerse(variable font)
uviterm-w|132x40 aiXterm for UniVerse(fixed font)
uviterms-vw|132x25 aiXterm for UniVerse(variable font)
uviterms-v|80x25 aiXterm for UniVerse(variable font)
uviterms-w|132x25 aiXterm for UniVerse(fixed font)
uviterms|80x25 aiXterm for UniVerse(fixed font)
uviterm|80x40 aiXterm for UniVerse(fixed font)
uvxterm-vw|132x40 Xterm for UniVerse(variable font)
uvxterm-v|80x40 Xterm for UniVerse(variable font)
uvxterm-w|132x40 Xterm for UniVerse(fixed font)
uvxterms-vw|132x25 Xterm for UniVerse(variable font)
uvxterms-v|80x25 Xterm for UniVerse(variable font)
uvxterms-w|132x25 Xterm for UniVerse(fixed font)
uvxterms|80x25 Xterm for UniVerse(fixed font)
uvxterm|80x40 Xterm for UniVerse(fixed font)
vp60-regent40|Adds viewpoint 60 (emulating Adds regent 40)
vp60|vwpt60|viewpoint60|Adds viewpoint 60
vp90|Adds Viewpoint 90
vp|av|vwpt|viewpoint|Adds viewpoint
vt100|vt100-am|vt100-uv|vt100ssg-uv|DEC vt100 terminal
vt200|vt220|DEC vt200/vt220 8 bit terminal
vt300|DEC vt300 Terminal
vt52|DEC vt52 Terminal
wy200-w|Wyse Technology 200 (132 Columns)
wy200|Wyse Technology 200
wy50-hz1500|Wyse Technology 50 (emulating Hazletine 1500)
wy50-tvi910|Wyse Technology 50 (emulating Televideo 910)
wy50-tvi920|Wyse Technology 50 (emulating Televideo 920)
wy50-tvi925|Wyse Technology 50 (emulating Televideo 925)
wy50-vp|ult50-vp|Wyse Technology 50 (emulating Adds viewpoint)
wy50|wy60|ult50|Wyse Technology 50/60
wy99gt|Wyse Technology 99GT Native Mode

```

terminfo Terminal Capabilities

Terminal descriptions define what sequences of characters are sent to the terminal to perform special functions. There are three kinds of capability:

- **Numeric capabilities** are limited to a length of five characters which must form a valid number. Only nonnegative numbers (0 through 32,767) are allowed. If a value for a particular capability does not apply, the field should be left blank.
- **Boolean capabilities** are indicated by the presence of the line in the file. If the line is omitted, the capability is not present.
- **String capabilities** are limited to a length of 512 bytes. There are two kinds of string capability: string and parameterized string.

String capabilities use the special characters shown in the following table.

Character	Description
\E or \e	The escape character (ASCII 27).
\n or \l	The linefeed character (ASCII 10).
\r	The carriage return character (ASCII 13).
\t	The tab character (ASCII 9).
\b	The backspace character (ASCII 8).
\f	The formfeed character (ASCII 12).
\s	A space (ASCII 32).
^x	Represents a control character (ASCII 0 through 31). The character <i>x</i> can be either uppercase or lowercase. Both ^A and ^a are Ctrl-A, or ASCII 1. ^@ is ASCII 0, ^[is ASCII 27 (or ESCAPE), ^\ is ASCII 28, ^] is ASCII 29, ^^ is ASCII 30, and ^_ is ASCII 31. ^? is the DEL character (ASCII 127).
\nnn	Represents the ASCII character with a value of <i>nnn</i> in octal. For example, \033 is the escape character (ASCII 27).

terminfo String Capabilities

Character	Description
\\	Represents the backslash (\) character.
\,	Represents the comma (,) character.
\^	Represents the caret (^) character.

***terminfo* String Capabilities (Continued)**

Parameterized string capabilities are shown in the following table.

Command	Description
%pn	Pushes parameter number <i>n</i> onto the stack. <i>n</i> is a number from 1 through 9.
%'c'	The ASCII value of character <i>c</i> is pushed onto the stack.
%{nnn}	Decimal number <i>nnn</i> is pushed onto the top of the stack.
%d	Pops the top parameter off the stack, and outputs it as a decimal number.
%nd	Pops the top parameter off the stack, and outputs it as a decimal number in a field <i>n</i> bytes wide. Spaces are used to fill out the field.
%0nd	Like %nd, except that zeros are used to fill out the field.
%c	The top of the stack is taken as a single ASCII character and output.
%s	The top of the stack is taken as a string and output.
%+ %- %* %/	The top two elements are popped off the stack and added, subtracted, multiplied, or divided. The result is pushed back on the stack. The fractional portion of a quotient is discarded.
%m	The second element on the stack is taken modulo of the first element, and the result is pushed onto the stack.
%& % %^	The top two elements are popped off the stack and a bitwise AND, OR, or XOR operation is performed. The result is pushed onto the stack.

***terminfo* Parameterized String Capabilities**

Command	Description
<code>% = % < % ></code>	The second element on the stack is tested for being equal to, less than, or greater than the first element. If the comparison is true, a 1 is pushed onto the stack, otherwise a 0 is pushed.
<code>% ! % ~</code>	The stack is popped, and either the logical or bitwise NOT of the first element is pushed onto the stack.
<code>% i</code>	One (1) is added to the first two parameters. This is useful for terminals that use a one-based cursor address rather than a zero-based.
<code>% P x</code>	Pops the stack, and places the result into variable <i>x</i> , where <i>x</i> is a lowercase letter (a–z).
<code>% g x</code>	Pushes the value of variable <i>x</i> on the top of the stack.
<code>% ? exp % t exp % e exp % ;</code>	Forms an if-then-else expression, with <code>% ?</code> representing IF, <code>% t</code> representing THEN, <code>% e</code> representing ELSE, and <code>% ;</code> terminating the expression. The else expression is optional. Else-If is possible, for example, <code>% ? C1 % t B1 % e C2 % t B2 % e C3 % t B3 % e C4 % t B4 % e % ;</code> <i>Cn</i> are conditions, and <i>Bn</i> are bodies.
<code>% %</code>	Outputs a percent sign (%).

***terminfo* Parameterized String Capabilities (Continued)**

Padding may be necessary for some string capabilities. A delay in milliseconds may appear anywhere within a string capability. A delay is specified by `$<nnn>`, where *nnn* is a decimal number indicating the number of milliseconds (1000ths of a second) of delay desired. A proper number of delay characters will be output, depending upon the current baud rate at the time.

terminfo, *termcap*, and UniVerse

Listed in the following table are the *terminfo* variables and their associated names in *termcap* and UniVerse. The variable type is also included.

terminfo Source	terminfo Usage	termcap	Type	UniVerse
acsc	acs_chars	acs	String	LINEDRAW.CHARACTER
am	auto_right_margi n	am	Boolean	AUTOMATIC.RIGHT. MARGIN
at1	at_1		String	AT.NEGATIVE.1
at2	at_2		String	AT.NEGATIVE.2
at3	at_3		String	AT.NEGATIVE.3
at4	at_4		String	AT.NEGATIVE.4
at5	at_5		String	AT.NEGATIVE.5
at6	at_6		String	AT.NEGATIVE.6
at7	at_7		String	AT.NEGATIVE.7
at8	at_8		String	AT.NEGATIVE.8
at9	at_9		String	AT.NEGATIVE.9
at10	at_10		String	AT.NEGATIVE.10
at11	at_11		String	AT.NEGATIVE.11
at12	at_12		String	AT.NEGATIVE.12
at13	at_13		String	AT.NEGATIVE.13
at14	at_14		String	AT.NEGATIVE.14
at15	at_15		String	AT.NEGATIVE.15
at16	at_16		String	AT.NEGATIVE.16
at17	at_17		String	AT.NEGATIVE.17

Terminal Variables

terminfo Source	terminfo Usage	termcap	Type	UniVerse
at18	at_18		String	AT.NEGATIVE.18
at19	at_19		String	AT.NEGATIVE.19
at20	at_20		String	AT.NEGATIVE.20
at21	at_21		String	AT.NEGATIVE.21
at22	at_22		String	AT.NEGATIVE.22
at23	at_23		String	AT.NEGATIVE.23
at24	at_24		String	AT.NEGATIVE.24
at25	at_25		String	AT.NEGATIVE.25
at26	at_26		String	AT.NEGATIVE.26
at27	at_27		String	AT.NEGATIVE.27
at28	at_28		String	AT.NEGATIVE.28
at29	at_29		String	AT.NEGATIVE.29
at30	at_30		String	AT.NEGATIVE.30
at31	at_31		String	AT.NEGATIVE.31
at32	at_32		String	AT.NEGATIVE.32
at33	at_33		String	AT.NEGATIVE.33
at34	at_34		String	AT.NEGATIVE.34
at35	at_35		String	AT.NEGATIVE.35
at36	at_36		String	AT.NEGATIVE.36
at37	at_37		String	AT.NEGATIVE.37
at38	at_38		String	AT.NEGATIVE.38
at39	at_39		String	AT.NEGATIVE.39
at40	at_40		String	AT.NEGATIVE.40

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
at41	at_41		String	AT.NEGATIVE.41
at42	at_42		String	AT.NEGATIVE.42
at43	at_43		String	AT.NEGATIVE.43
at44	at_44		String	AT.NEGATIVE.44
at45	at_45		String	AT.NEGATIVE.45
at46	at_46		String	AT.NEGATIVE.46
at47	at_47		String	AT.NEGATIVE.47
at48	at_48		String	AT.NEGATIVE.48
at49	at_49		String	AT.NEGATIVE.49
at50	at_50		String	AT.NEGATIVE.50
at51	at_51		String	AT.NEGATIVE.51
at52	at_52		String	AT.NEGATIVE.52
at53	at_53		String	AT.NEGATIVE.53
at54	at_54		String	AT.NEGATIVE.54
at55	at_55		String	AT.NEGATIVE.55
at56	at_56		String	AT.NEGATIVE.56
at57	at_57		String	AT.NEGATIVE.57
at58	at_58		String	AT.NEGATIVE.58
at59	at_59		String	AT.NEGATIVE.59
at60	at_60		String	AT.NEGATIVE.60
at61	at_61		String	AT.NEGATIVE.61
at62	at_62		String	AT.NEGATIVE.62
at63	at_63		String	AT.NEGATIVE.63

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
at64	at_64		String	AT.NEGATIVE.64
at65	at_65		String	AT.NEGATIVE.65
at66	at_66		String	AT.NEGATIVE.66
at67	at_67		String	AT.NEGATIVE.67
at68	at_68		String	AT.NEGATIVE.68
at69	at_69		String	AT.NEGATIVE.69
at70	at_70		String	AT.NEGATIVE.70
at71	at_71		String	AT.NEGATIVE.71
at72	at_72		String	AT.NEGATIVE.72
at73	at_73		String	AT.NEGATIVE.73
at74	at_74		String	AT.NEGATIVE.74
at75	at_75		String	AT.NEGATIVE.75
at76	at_76		String	AT.NEGATIVE.76
at77	at_77		String	AT.NEGATIVE.77
at78	at_78		String	AT.NEGATIVE.78
at79	at_79		String	AT.NEGATIVE.79
at80	at_80		String	AT.NEGATIVE.80
at81	at_81		String	AT.NEGATIVE.81
at82	at_82		String	AT.NEGATIVE.82
at83	at_83		String	AT.NEGATIVE.83
at84	at_84		String	AT.NEGATIVE.84
at85	at_85		String	AT.NEGATIVE.85
at86	at_86		String	AT.NEGATIVE.86

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
at87	at_87		String	AT.NEGATIVE.87
at88	at_88		String	AT.NEGATIVE.88
at89	at_89		String	AT.NEGATIVE.89
at90	at_90		String	AT.NEGATIVE.90
at91	at_91		String	AT.NEGATIVE.91
at92	at_92		String	AT.NEGATIVE.92
at93	at_93		String	AT.NEGATIVE.93
at94	at_94		String	AT.NEGATIVE.94
at95	at_95		String	AT.NEGATIVE.95
at96	at_96		String	AT.NEGATIVE.96
at97	at_97		String	AT.NEGATIVE.97
at98	at_98		String	AT.NEGATIVE.98
at99	at_99		String	AT.NEGATIVE.99
at100	at_100		String	AT.NEGATIVE.100
at101	at_101		String	AT.NEGATIVE.101
at102	at_102		String	AT.NEGATIVE.102
at103	at_103		String	AT.NEGATIVE.103
at104	at_104		String	AT.NEGATIVE.104
at105	at_105		String	AT.NEGATIVE.105
at106	at_106		String	AT.NEGATIVE.106
at107	at_107		String	AT.NEGATIVE.107
at108	at_108		String	AT.NEGATIVE.108
at109	at_109		String	AT.NEGATIVE.109

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
at110	at_110		String	AT.NEGATIVE.110
at111	at_111		String	AT.NEGATIVE.111
at112	at_112		String	AT.NEGATIVE.112
at113	at_113		String	AT.NEGATIVE.113
at114	at_114		String	AT.NEGATIVE.114
at115	at_115		String	AT.NEGATIVE.115
at116	at_116		String	AT.NEGATIVE.116
at117	at_117		String	AT.NEGATIVE.117
at118	at_118		String	AT.NEGATIVE.118
at119	at_119		String	AT.NEGATIVE.119
bel	bell	bl	String	BELL
blink	enter_blink_mode	mb	String	VIDEO.BLINK
bold	enter_bold_mode	md	String	VIDEO.BOLD
bs	backspace	bs	String	BACKSPACE
bw	auto_left_margin	bw	Boolean	AUTOMATIC.LEFT. MARGIN
cbt	back_tab	bt	String	BACK.TAB
chts	cursor_hard		Boolean	HARD.CURSOR
civis	cursor_invisible	vi	String	CURSOR.INVISIBLE
clear	clear_screen	cl	String	ERASE.SCREEN
cmdch	command_character	CC	String	COMMAND.CHARACTER
cnorm	cursor_normal	ve	String	CURSOR.NORMAL
cols	columns	co	Number	COLUMNS

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
cr	carriage_return	cr	String	CARRIAGE.RETURN
csr	change_scroll_region	cs	Prm. String	CHANGE.SCROLL.REGION
ctab	clear_tab		String	TAB.STOP.CLEAR
cub	parm_left_cursor	LE	Prm. String	MOVE.CURSOR.LEFT.PARM
cub1	cursor_left	le	String	MOVE.CURSOR.LEFT
cud	parm_down_cursor	DO	Prm. String	MOVE.CURSOR.DOWN.PARM
cud1	cursor_down	do	String	MOVE.CURSOR.DOWN
cuf	parm_right_cursor	RI	Prm. String	MOVE.CURSOR.RIGHT.PARM
cuf1	cursor_right	nd	String	MOVE.CURSOR.RIGHT
cup	cursor_address	cm	Prm. String	MOVE.CURSOR.TO.ADDRESS
cuu	parm_up_cursor	UP	Prm. String	MOVE.CURSOR.UP.PARM
cuu1	cursor_up	up	String	MOVE.CURSOR.UP
cvvis	cursor_visible	vs	String	CURSOR.VISIBLE
da	memory_above	da	Boolean	MEMORY.ABOVE
db	memory_below	db	Boolean	MEMORY.BELOW
dch	parm_dch	DC	Prm. String	DELETE.CHARACTER.PARM
dch1	delete_character	dc	String	DELETE.CHARACTER
dim	enter_dim_mode	mh	String	VIDEO.DIM
dl	parm_delete_line	DL	Prm. String	DELETE.LINE.PARM

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
dl1	delete_line	dl	String	DELETE.LINE
dldblc	d_ld_botleft		String	DBLE.LDRAW.LO.LEFT. CORNER
dldbrc	d_ld_botright		String	DBLE.LDRAW.LO. RIGHT.CORNER
dldh	d_ld_hor		String	DBLE.LDRAW.HORIZ
dldtcr	d_ld_cross		String	DBLE.LDRAW.CROSS
dldtd	d_ld_tdown		String	DBLE.LDRAW.LO.TEE
dldtl	d_ld_tleft		String	DBLE.LDRAW.LEFT.TEE
dldtr	d_ld_tright		String	DBLE.LDRAW.RIGHT. TEE
dldtu	d_ld_tup		String	DBLE.LDRAW.UP.TEE
dldulc	d_ld_upleft		String	DBLE.LDRAW.UP.LEFT. CORNER
dldurc	d_ld_upright		String	DBLE.LDRAW.UP. RIGHT.CORNER
dldv	d_ld_vert		String	DBLE.LDRAW.VERT
dsl	dis_status_line	ds	String	STATUS.LINE.DISABLE
ebos	clr_bos		String	ERASE.TO.BEGINNING OF.SCREEN
ech	erase_chars	ec	Prm. String	ERASE.CHARACTERS
ed	clr_eos	cd	String	ERASE.TO.END.OF. SCREEN
el	clr_eol	ce	String	ERASE.TO.END.OF. LINE

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
el1	clr_bol	cb	String	ERASE.TO. BEGINNING.OF.LINE
eline	clr_line		String	ERASE.LINE
enacs	ena_acs	eA	String	ENABLE.LINEDRAW
eo	erase_overstrike	eo	Boolean	ERASES.OVERSTRIKE
eslok	status_line_esc_o k	es	Boolean	STATUS.LINE.ESC.OK
ff	form_feed	ff	String	FORM.FEED
flash	flash_screen	vb	String	SCREEN.FLASH
fsl	from_status_line	fs	String	STATUS.LINE.END
gn	generic_type	gn	Boolean	GENERIC.TYPE
gofl	goto_func_line		Prm. String	FUNCTION.LINE. BEGIN
hc	hard_copy	hc	Boolean	HARD.COPY
hd	down_half_line	hd	String	DOWN.HALF.LINE
hfl	has_function_line		Boolean	HAS.FUNCTION.LINE
home	cursor_home	ho	String	MOVE.CURSOR.TO. HOME
hpa	column_address	ch	Prm. String	MOVE.CURSOR.TO. COLUMN
hs	has_status_line	hs	Boolean	HAS.STATUS.LINE
ht	tab	ta	String	TAB
hts	set_tab	st	String	TAB.STOP.SET
hu	up_half_line	hu	String	UP.HALF.LINE

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
hz	tilde_glitch	hz	Boolean	UNABLE.TO.PRINT. TILDE
ich	parm_ich	IC	Prm. String	INSERT.CHARACTER. PARM
ich1	ins_prefix	ic	String	INS.PREFIX
ichx	insert_character		String	INSERT.CHARACTER
if	init_file	if	String	INIT.FILE
il	parm_insert_line	il	Prm. String	INSERT.LINE.PARM
il1	insert_line	al	String	INSERT.LINE
in	insert_null_glitch	in	Boolean	INSERT.NULL.SPECIAL
ind	scroll_forward	sf	String	SCROLL.UP
indn	parm_index	SF	Prm. String	SCROLL.UP.PARM
invis	enter_secure_mode	mk	String	VIDEO.BLANK
ip	insert_padding	ip	String	INSERT.PAD
ipro	init_prog	iP	String	INIT.PROG
is1	init_1string	i1	String	INIT.1STRING
is2	init_2string	is	String	INIT.2STRING
is3	init_3string	i2	String	INIT.3STRING
it	init_tabs		Number	TAB.STOP.INITIAL
kBEG	key_sbeg	&9	String	KEY.SBEG
kCAN	key_scancel	&0	String	KEY.SCANCEL
kCMD	key_scommand	*1	String	KEY.SCOMMAND
kCPY	key_scopy	*2	String	KEY.SCOPY

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kCRT	key_screate	*3	String	KEY.SCREATE
kDC	key_sdc	*4	String	KEY.SDC
kDL	key_sdl	*5	String	KEY.SDL
kEND	key_send	*7	String	KEY.SEND
kEOL	key_seol	*8	String	KEY.SEOL
kEXT	key_sexit	*9	String	KEY.SEXIT
kFND	key_sfind	*0	String	KEY.SFIND
kHLP	key_shelp	#1	String	KEY.SHELP
kHOM	key_shome	#2	String	KEY.SHOME
kIC	key_sic	#3	String	KEY.SIC
kLFT	key_sleft	#4	String	KEY.SLEFT
kMOV	key_smove	%b	String	KEY.SMOVE
kMSG	key_smessage	%a	String	KEY.SMESSAGE
kNXT	key_snext	%c	String	KEY.SNEXT
kOPT	key_soptions	%d	String	KEY.SOPTIONS
kPRT	key_sprint	%f	String	KEY.SPRINT
kPRV	key_sprevious	%e	String	KEY.SPREVIOUS
kRDO	key_sredo	%g	String	KEY.SREDO
kRES	key_sresume	%j	String	KEY.SRESUME
kRIT	key_sright	%i	String	KEY.SRIGHT
kRPL	key_sreplace	%h	String	KEY.SREPLACE
kSAV	key_ssav	!1	String	KEY.SSAVE
kSPD	key_ssuspend	!2	String	KEY.SSUSPEND

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kUND	key_sundo	!3	String	KEY.SUNDO
ka1	key_a1	K1	String	KEY.A1
ka3	key_a3	K3	String	KEY.A3
kb2	key_b2	K2	String	KEY.B2
kbeg	key_beg	@1	String	KEY.BEG
kbs	key_backspace	kb	String	KEY.BACKSPACE
kc1	key_c1	K4	String	KEY.C1
kc3	key_c3	K5	String	KEY.C3
kcan	key_cancel	@2	String	KEY.CANCEL
kcbt	key_cbt	KB	String	KEY.BACK.TAB
kclo	key_close	@3	String	KEY.CLOSE
kclr	key_clear	KC	String	KEY.ERASE.SCREEN
kcmd	key_command	@4	String	KEY.COMMAND
kcpy	key_copy	@5	String	KEY.COPY
kcrt	key_create	@6	String	KEY.CREATE
kctab	key_ctab	kt	String	KEY.TAB.STOP.CLEAR
kcub1	key_left	kl	String	KEY.MOVE.CURSOR. LEFT
kcud1	key_down	kd	String	KEY.MOVE.CURSOR. DOWN
kcuf1	key_right	kr	String	KEY.MOVE.CURSOR. RIGHT
kcuu1	key_up	ku	String	KEY.MOVE.CURSOR. UP

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kdch1	key_dc	kD	String	KEY.DELETE. CHARACTER
kdl1	key_dl	kL	String	KEY.DELETE.LINE
kebck	ke_back		String	KEYEDIT.MOVE. BACKWARD
kebs	ke_backspace		String	KEYEDIT.BACKSPACE
ked	key_eos	kS	String	KEY.ERASE.END.OF. SCREEN
kedel	ke_delete_ character		String	KEYEDIT.DELETE. CHARACTER
keeol	ke_eof		String	KEYEDIT.ERASE.END. OF.FIELD
keera	ke_erase		String	KEYEDIT.ERASE.FIELD
keesc	ke_escape_prefix		String	KEYEDIT.ESCAPE
kefun	ke_function_prefi x		String	KEYEDIT.FUNCTION
kefwd	ke_forward		String	KEYEDIT.MOVE.FORWARD
keins	ke_ins_character		String	KEYEDIT.INSERT. CHARACTER
keiof	ke_insoff		String	KEYEDIT.INSERT. MODE.END
keion	ke_inson		String	KEYEDIT.INSERT. MODE.BEGIN
keitg	ke_ins_toggle		String	KEYEDIT.INSERT. MODE.TOGGLE
kel	key_eol	kE	String	KEY.ERASE.END.OF. LINE

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kend	key_end	@7	String	KEY.END
kent	key_enter	@8	String	KEY.ENTER
kexit	ke_exit_input_mode		String	KEYEDIT.EXIT
kext	key_exit	@9	String	KEY.EXIT
kf0	key_f0	k0	String	KEY.FUNCTION.0
kf1	key_f1	k1	String	KEY.FUNCTION.1
kf2	key_f2	k2	String	KEY.FUNCTION.2
kf3	key_f3	k3	String	KEY.FUNCTION.3
kf4	key_f4	k4	String	KEY.FUNCTION.4
kf5	key_f5	k5	String	KEY.FUNCTION.5
kf6	key_f6	k6	String	KEY.FUNCTION.6
kf7	key_f7	k7	String	KEY.FUNCTION.7
kf8	key_f8	k8	String	KEY.FUNCTION.8
kf9	key_f9	k9	String	KEY.FUNCTION.9
kf10	key_f10	k;	String	KEY.FUNCTION.10
kf11	key_f11	F1	String	KEY.FUNCTION.11
kf12	key_f12	F2	String	KEY.FUNCTION.12
kf13	key_f13	F3	String	KEY.FUNCTION.13
kf14	key_f14	F4	String	KEY.FUNCTION.14
kf15	key_f15	F5	String	KEY.FUNCTION.15
kf16	key_f16	F6	String	KEY.FUNCTION.16
kf17	key_f17	F7	String	KEY.FUNCTION.17

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kf18	key_f18	F8	String	KEY.FUNCTION.18
kf19	key_f19	F9	String	KEY.FUNCTION.19
kf20	key_f20	FA	String	KEY.FUNCTION.20
kf21	key_f21	FB	String	KEY.FUNCTION.21
kf22	key_f22	FC	String	KEY.FUNCTION.22
kf23	key_f23	FD	String	KEY.FUNCTION.23
kf24	key_f24	FE	String	KEY.FUNCTION.24
kf25	key_f25	FF	String	KEY.FUNCTION.25
kf26	key_f26	FG	String	KEY.FUNCTION.26
kf27	key_f27	FH	String	KEY.FUNCTION.27
kf28	key_f28	FI	String	KEY.FUNCTION.28
kf29	key_f29	FJ	String	KEY.FUNCTION.29
kf30	key_f30	FK	String	KEY.FUNCTION.30
kf31	key_f31	FL	String	KEY.FUNCTION.31
kf32	key_f32	FM	String	KEY.FUNCTION.32
kf33	key_f33	FN	String	KEY.FUNCTION.33
kf34	key_f34	FO	String	KEY.FUNCTION.34
kf35	key_f35	FP	String	KEY.FUNCTION.35
kf36	key_f36	FQ	String	KEY.FUNCTION.36
kf37	key_f37	FR	String	KEY.FUNCTION.37
kf38	key_f38	FS	String	KEY.FUNCTION.38
kf39	key_f39	FT	String	KEY.FUNCTION.39
kf40	key_f40	FU	String	KEY.FUNCTION.40

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kf41	key_f41	FV	String	KEY.FUNCTION.41
kf42	key_f42	FW	String	KEY.FUNCTION.42
kf43	key_f43	FX	String	KEY.FUNCTION.43
kf44	key_f44	FY	String	KEY.FUNCTION.44
kf45	key_f45	FZ	String	KEY.FUNCTION.45
kf46	key_f46	Fa	String	KEY.FUNCTION.46
kf47	key_f47	Fb	String	KEY.FUNCTION.47
kf48	key_f48	Fc	String	KEY.FUNCTION.48
kf49	key_f49	Fd	String	KEY.FUNCTION.49
kf50	key_f50	Fe	String	KEY.FUNCTION.50
kf51	key_f51	Ff	String	KEY.FUNCTION.51
kf52	key_f52	Fg	String	KEY.FUNCTION.52
kf53	key_f53	Fh	String	KEY.FUNCTION.53
kf54	key_f54	Fi	String	KEY.FUNCTION.54
kf55	key_f55	Fj	String	KEY.FUNCTION.55
kf56	key_f56	Fk	String	KEY.FUNCTION.56
kf57	key_f57	Fl	String	KEY.FUNCTION.57
kf58	key_f58	Fm	String	KEY.FUNCTION.58
kf59	key_f59	Fn	String	KEY.FUNCTION.59
kf60	key_f60	Fo	String	KEY.FUNCTION.60
kf61	key_f61	Fp	String	KEY.FUNCTION.61
kf62	key_f62	Fq	String	KEY.FUNCTION.62
kf63	key_f63	Fr	String	KEY.FUNCTION.63

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kfind	key_find	@0	String	KEY.FIND
khlp	key_help	%1	String	KEY.HELP
khome	key_home	kh	String	KEY.MOVE.CURSOR. TO.HOME
khts	key_stab	kT	String	KEY.TAB.STOP.SET
kichl	key_ic	kI	String	KEY.IC
kichx	key_ichl		String	KEY.INSERT. CHARACTER
kill	key_il	kA	String	KEY.INSERT.LINE
kind	key_sf	kF	String	KEY.SCROLL.UP
kll	key_ll	kH	String	KEY.MOVE.CURSOR. TO.LAST.LINE
km	has_meta_key	km	Boolean	HAS.META.KEY
kmov	key_move	%4	String	KEY.MOVE
kmrk	key_mark	%2	String	KEY.MARK
kmsg	key_message	%3	String	KEY.MESSAGE
knp	key_npage	kN	String	KEY.NEXT.PAGE
knxt	key_next	%5	String	KEY.NEXT
kopn	key_open	%6	String	KEY.OPEN
kopt	key_options	%7	String	KEY.OPTIONS
kpp	key_ppage	kP	String	KEY.PREVIOUS.PAGE
kprt	key_print	%9	String	KEY.PRINT
kprv	key_previous	%8	String	KEY.PREVIOUS
krdo	key_redo	%0	String	KEY.REDO

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
kref	key_reference	&1	String	KEY.REFERENCE
kres	key_resume	&5	String	KEY.RESUME
krfr	key_refresh	&2	String	KEY.REFRESH
kri	key_sr	kR	String	KEY.SCROLL.DOWN
kmir	key_eic	kM	String	KEY.INSERT.MODE.END
krpl	key_replace	&3	String	KEY.REPLACE
krst	key_restart	&4	String	KEY.RESTART
ksav	key_save	&6	String	KEY.SAVE
ksend	key_send	*7	String	KEY.SEND
kslt	key_select	*6	String	KEY.SELECT
ksmir	key_smir		String	KEY.INSERT.MODE.ON
kspd	key_suspend	&7	String	KEY.SUSPEND
ktbc	key_catab	ka	String	KEY.TAB.STOP.CLEAR.ALL
ktmir	key_toggle_ir		String	KEY.INSERT.MOVE. TOGGLE
kund	key_undo	&8	String	KEY.UNDO
ldatt	line_attribute		Number	LINE.ATTRIBUTE
ldbl	ld_botleft		String	LINEDRAW.LOWER. LEFT.CORNER
ldblcdh	ld_boledho		String	LDRAW.LO.LEFT. CORNER.DBLE.HORIZ
ldblcdv	ld_boledve		String	LDRAW.LO.LEFT. CORNER.DBLE.VERT
ldbr	ld_botright		String	LINEDRAW.LOWER. RIGHT.CORNER

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
ldbrcdh	ld_boridho		String	LDRAW.LO.RIGHT. CORNER.DBLE.HORIZ
ldbrcdv	ld_boridve		String	LDRAW.LO.RIGHT. CORNER.DBLE.VERT
ldbtdh	ld_tbodhor		String	LDRAW.LOWER.TEE. DBLE.HORIZ
ldbtdv	ld_tbodver		String	LDRAW.LOWER.TEE. DBLE.VERT
ldcrdh	ld_cr_d_ho		String	LDRAW.CROSS.DBLE. HORIZ
ldcrdv	ld_cr_d_ve		String	LDRAW.CROSS.DBLE. VERT
ldhb	ld_horbot		String	LINEDRAW.LOWER. HORIZONTAL
ldhc	ld_horctr		String	LINEDRAW.CENTER. HORIZONTAL
ldht	ld_hortop		String	LINEDRAW.UPPER. HORIZONTAL
ldltdh	ld_tledhor		String	LDRAW.LEFT.TEE.DBLE. HORIZ
ldltdv	ld_tledver		String	LDRAW.LEFT.TEE.DBLE. VERT
ldrtdh	ld_tridhor		String	LDRAW.RIGHT.TEE. DBLE.HORIZ
ldrtdv	ld_tridver		String	LDRAW.RIGHT.TEE. DBLE.VERT
ldtc	ld_tcross		String	LINEDRAW.CROSS
ldtd	ld_tdown		String	LINEDRAW.UPPER.TEE
ldtl	ld_tleft		String	LINEDRAW.RIGHT.TEE

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
ldtr	ld_tright		String	LINEDRAW.LEFT.TEE
ldtu	ld_tup		String	LINEDRAW.LOWER.TEE
ldul	ld_upleft		String	LINEDRAW.UPPER. LEFT.CORNER
ldulcdh	ld_upledho		String	LDRAW.UP.LEFT. CORNER.DBLE.HORIZ
ldulcdv	ld_upledve		String	LDRAW.UP.LEFT. CORNER.DBLE.VERT
ldur	ld_upright		String	LINEDRAW.UPPER. RIGHT.CORNER
ldurcdh	ld_upridho		String	LDRAW.UP.RIGHT. CORNER.DBLE.HORIZ
ldurcdv	ld_upridve		String	LDRAW.UP.RIGHT. CORNER.DBLE.VERT
ldutdh	ld_tupdhor		String	LDRAW.UP.TEE.DBLE. HORIZ
ldutdv	ld_tupdver		String	LDRAW.UP.TEE.DBLE. VERT
ldvc	ld_vertcenter		String	LINEDRAW.CENTER. VERTICAL
ldvl	ld_vertleft		String	LINEDRAW.LEFT. VERTICAL
ldvr	ld_vertright		String	LINEDRAW.RIGHT. VERTICAL
lf	linefeed	lf	String	LINE.FEED
lf0	lab_f0	l0	String	LABEL.KEY.FUNCTION.0
lf1	lab_f1	l1	String	LABEL.KEY.FUNCTION.1
lf2	lab_f2	l2	String	LABEL.KEY.FUNCTION.2

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
lf3	lab_f3	l3	String	LABEL.KEY.FUNCTION.3
lf4	lab_f4	l4	String	LABEL.KEY.FUNCTION.4
lf5	lab_f5	l5	String	LABEL.KEY.FUNCTION.5
lf6	lab_f6	l6	String	LABEL.KEY.FUNCTION.6
lf7	lab_f7	l7	String	LABEL.KEY.FUNCTION.7
lf8	lab_f8	l8	String	LABEL.KEY.FUNCTION.8
lf9	lab_f9	l9	String	LABEL.KEY.FUNCTION.9
lf10	lab_f10	la	String	LABEL.KEY.FUNCTION.10
lf11	lab_f11		String	LABEL.KEY.FUNCTION.11
lf12	lab_f12		String	LABEL.KEY.FUNCTION.12
lf13	lab_f13		String	LABEL.KEY.FUNCTION.13
lf14	lab_f14		String	LABEL.KEY.FUNCTION.14
lf15	lab_f15		String	LABEL.KEY.FUNCTION.15
lf16	lab_f16		String	LABEL.KEY.FUNCTION.16
lf17	lab_f17		String	LABEL.KEY.FUNCTION.17
lf18	lab_f18		String	LABEL.KEY.FUNCTION.18
lf19	lab_f19		String	LABEL.KEY.FUNCTION.19
lf20	lab_f20		String	LABEL.KEY.FUNCTION.20
lf21	lab_f21		String	LABEL.KEY.FUNCTION.21
lf22	lab_f22		String	LABEL.KEY.FUNCTION.22
lf23	lab_f23		String	LABEL.KEY.FUNCTION.23
lf24	lab_f24		String	LABEL.KEY.FUNCTION.24
lf25	lab_f25		String	LABEL.KEY.FUNCTION.25

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
lf26	lab_f26		String	LABEL.KEY.FUNCTION.26
lf27	lab_f27		String	LABEL.KEY.FUNCTION.27
lf28	lab_f28		String	LABEL.KEY.FUNCTION.28
lf29	lab_f29		String	LABEL.KEY.FUNCTION.29
lf30	lab_f30		String	LABEL.KEY.FUNCTION.30
lf31	lab_f31		String	LABEL.KEY.FUNCTION.31
lf32	lab_f32		String	LABEL.KEY.FUNCTION.32
lf33	lab_f33		String	LABEL.KEY.FUNCTION.33
lf34	lab_f34		String	LABEL.KEY.FUNCTION.34
lf35	lab_f35		String	LABEL.KEY.FUNCTION.35
lf36	lab_f36		String	LABEL.KEY.FUNCTION.36
lf37	lab_f37		String	LABEL.KEY.FUNCTION.37
lf38	lab_f38		String	LABEL.KEY.FUNCTION.38
lf39	lab_f39		String	LABEL.KEY.FUNCTION.39
lf40	lab_f40		String	LABEL.KEY.FUNCTION.40
lf41	lab_f41		String	LABEL.KEY.FUNCTION.41
lf42	lab_f42		String	LABEL.KEY.FUNCTION.42
lf43	lab_f43		String	LABEL.KEY.FUNCTION.43
lf44	lab_f44		String	LABEL.KEY.FUNCTION.44
lf45	lab_f45		String	LABEL.KEY.FUNCTION.45
lf46	lab_f46		String	LABEL.KEY.FUNCTION.46
lf47	lab_f47		String	LABEL.KEY.FUNCTION.47
lf48	lab_f48		String	LABEL.KEY.FUNCTION.48

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
lf49	lab_f49		String	LABEL.KEY.FUNCTION.49
lf50	lab_f50		String	LABEL.KEY.FUNCTION.50
lf51	lab_f51		String	LABEL.KEY.FUNCTION.51
lf52	lab_f52		String	LABEL.KEY.FUNCTION.52
lf53	lab_f53		String	LABEL.KEY.FUNCTION.53
lf54	lab_f54		String	LABEL.KEY.FUNCTION.54
lf55	lab_f55		String	LABEL.KEY.FUNCTION.55
lf56	lab_f56		String	LABEL.KEY.FUNCTION.56
lf57	lab_f57		String	LABEL.KEY.FUNCTION.57
lf58	lab_f58		String	LABEL.KEY.FUNCTION.58
lf59	lab_f59		String	LABEL.KEY.FUNCTION.59
lf60	lab_f60		String	LABEL.KEY.FUNCTION.60
lf61	lab_f61		String	LABEL.KEY.FUNCTION.61
lf62	lab_f62		String	LABEL.KEY.FUNCTION.62
lf63	lab_f63		String	LABEL.KEY.FUNCTION.63
lh	label_height	lh	Number	LABEL.HEIGHT
lines	lines	li	Number	LINES
ll	cursor_to_ll	ll	String	MOVE.CURSOR.TO. LAST.LINE
lm	lines_of_memory	lm	Number	LINES.OF.MEMORY
lw	label_width	lw	Number	LABEL.WIDTH
macs	move_alternate_charset		Boolean	MOVE.LINEDRAW. MODE
mc0	print_screen	ps	String	PRINT.SCREEN

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
mc4	prtr_off	pf	String	PRINT.MODE.BEGIN
mc5	prtr_on	po	String	PRINT.MOVE.END
mc5i	prtr_silent	5i	Boolean	PRTR.SILENT
mc5p	prtr_non	pO	String	PRTR.NON
mgc	clear_margins	MC	String	CLEAR.MARGINS
mir	move_insert_mode	mi	Boolean	MOVE.INSERT.MODE
mrcup	cursor_mem_address	CM	Prm. String	CURSOR.MEM. ADDRESS
msgr	move_standout_mode	ms	Boolean	MOVE.VIDEO.MODE
nel	newline	nw	String	NEWLINE
nlab	num_labels	Nl	Number	NUM.LABELS
norm	enter_normal_video		String	VIDEO.NORMAL
npc	no_pad_char	NP	Boolean	NO.PAD.CHAR
nrrmc	non_rev_rmcup	NR	Boolean	NON.REV.RMCUP
nxon	needs_xon_xoff	nx	Boolean	NEEDS.XON.XOFF
os	over_strike	os	Boolean	OVERSTRIKES
pad	pad_char	pc	String	PADDING.CHARACTER
pb	padding_baud_rate	pb	Number	PAD.BAUD.RATE
pblnk	pro_blink		String	PROTECT.VIDEO.BLINK
pbold	pro_bold		String	PROTECT.VIDEO.BOLD
pdim	pro_dim		String	PROTECT.VIDEO.DIM

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
pkkey	pkey_key	pk	String	PKEY.KEY
pfloc	pkey_local	pl	String	PKEY.LOCAL
pfx	pkey_xmit	px	String	PKEY.XMIT
pinv	pro_blank		String	PROTECT.VIDEO.BLANK
pln	plab_norm	pn	String	PLAB.NORM
pnorm	pro_normal		String	PROTECT.VIDEO. NORMAL
prcol	protect_column		String	WRITE.PROTECT. COLUMN
prev	pro_reverse		String	PROTECT.VIDEO. REVERSE
prot	enter_protected_ mode	mp	String	ENTER.PROTECTED. MODE
pso	pro_standout		String	PROTECT.VIDEO. STANDOUT
pulin	pro_underline		String	PROTECT.VIDEO. UNDERLINE
rc	restore_cursor	rc	String	CURSOR.RESTORE
refl	leave_func_line		String	FUNCTION.LINE.END
rep	repeat_char	rp	String	REPEAT.CHAR
rev	enter_reverse_mo de	mr	String	VIDEO.REVERSE
rf	reset_file	rf	String	RESET.FILE
rfi	req_for_input	RF	String	REQ.FOR.INPUT
ri	scroll_reverse	sr	String	SCROLL.DOWN

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
rin	parm_rindex	SR	Prm. String	SCROLL.DOWN.PARM
rmacs	exit_alt_charset_ mode	ae	String	LINEDRAW.END
rmam	exit_am_mode	RA	String	EXIT.AM.MODE
rmblok	end_block_mode		String	BLOCK.MODE.END
rmclk	click_off		String	KEYCLICK.OFF
rmcup	exit_ca_mode	te	String	EXIT.CA.MODE
rmde	exit_delete_mode	ed	String	EXIT.DELETE.MODE
rmir	exit_insert_mode	ei	String	INSERT.MODE.END
rmkx	keypad_local	ke	String	KEYPAD.LOCAL
rmlock	exit_keyboard_lo ck		String	KEYBOARD.LOCK.OFF
rmln	label_off	LF	String	LABEL.OFF
rmmeta	meta_off	mo	String	META.OFF
rmmon	exit_monitor_mo de		String	MONITOR.MODE.OFF
rmpp	char_padding	rP	String	CHAR.PADDING
rmpro	exit_screen_prote ct		String	SCREEN.PROTECT.END
rmscr	end_scroll_mode		String	SCROLL.MODE.END
rmso	exit_standout_mo de	se	String	EXIT.STANDOUT.MODE
rmul	exit_underline_m ode	ue	String	EXIT.UNDERLINE.MODE
rmwp	exit_write_protect		String	WRITE.PROTECT.END
rmxon	exit_xon_mode	RX	String	EXIT.XON.MODE

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
rs1	reset_1string	r1	String	RESET.1STRING
rs2	reset_2string	r2	String	RESET.2STRING
rs3	reset_3string	r3	String	RESET.3STRING
sc	save_cursor	sc	String	CURSOR.SAVE
seom	store_eom		String	STORE.END.OF.MESSAGE
sgr	set_attributes	sa	Prm. String	SET.ATTRIBUTES
sgr0	exit_attribute_mode	me	String	EXIT.ATTRIBUTE.MODE
smacs	enter_alt_charset_mode	as	String	LINEDRAW.BEGIN
smam	enter_am_mode	SA	String	ENTER.AM.MODE
smbk	begin_block_mode		String	BLOCK.MODE.BEGIN
smclk	klick_on		String	KEYCLICK.ON
smcup	enter_ca_mode	ti	String	ENTER.CA.MODE
smdc	enter_delete_mode	dm	String	ENTER.DELETE.MODE
smgl	set_left_margin	ML	String	SET.LEFT.MARGIN
smgr	set_right_margin	MR	String	SET.RIGHT.MARGIN
smir	enter_insert_mode	im	String	INSERT.MODE.BEGIN
smkx	keypad_xmit	ks	String	KEYPAD.XMIT
smkck	enter_keyboard_lock		String	KEYBOARD.LOCK.ON
smln	label_on	LO	String	LABEL.ON
smm	meta_on	mm	String	META.ON

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
smmon	enter_monitor_mode		String	MONITOR.MODE.ON
smpro	enter_screen_protect		String	SCREEN.PROTECT.BEGIN
smscr	begin_scroll_mode		String	SCROLL.MODE.BEGIN
smso	enter_standout_mode	so	String	VIDEO.STANDOUT
smul	enter_underline_mode	us	String	VIDEO.UNDERLINE
smwp	enter_write_protect		String	WRITE.PROTECT.BEGIN
smxon	enter_xon_mode	SX	String	ENTER.XON.MODE
sndla	send_aline		String	SEND.LINE.ALL
sndlu	send_uline		String	SEND.LINE. UNPROTECTED
sndma	send_amessage		String	SEND.MESSAGE.ALL
sndmu	send_umessage		String	SEND.MESSAGE. UNPROTECTED
sndpa	send_apage		String	SEND.PAGE.ALL
sndpu	send_upage		String	SEND.PAGE. UNPROTECTED
ssom	store_som		String	STORE.START.OF.MESSAG E
tbc	clear_all_tabs	ct	String	CLEAR.ALL.TAB.STOPS
termf	field_terminator		String	TERMINATE.FIELD
termf	line_terminator		String	TERMINATE.LINE

Terminal Variables (Continued)

terminfo Source	terminfo Usage	termcap	Type	UniVerse
termpp	page_terminator		String	TERMINATE.PAGE
tsl	to_status_line	ts	String	STATUS.LINE.BEGIN
uc	underline_char	uc	String	UNDERLINE.CHAR
ul	transparent_ underline	ul	Boolean	TRANSPARENT. UNDERLINE
vidof	video_off		String	SCREEN.VIDEO.OFF
vidon	video_on		String	SCREEN.VIDEO.ON
vpa	row_address	cv	Prm. String	MOVE.CURSOR.TO.ROW
vt	virtual_terminal		Number	VIRTUAL.TERMINAL
wind	set_window	wi	Prm. String	SET.WINDOW
wsl	width_status_line	ws	Number	STATUS.LINE.WIDTH
xenl	eat_newline_glitch	xn	Boolean	RIGHT.MARGIN.EATS. NEWLINE
xhp	ceol_standout_glitch	xs	Boolean	CEOL.STANDOUT.GLITCH
xmc	magic_cookie_glitch	sg	Number	VIDEO.SPACES
xoffc	xoff_character	XF	String	XOFF.CHARACTER
xon	xon_xoff	xo	Boolean	XON.XOFF
xonc	xon_character	XN	String	XON.CHARACTER
xb	beehive_glitch	xb	Boolean	BEEHIVE.GLITCH
xt	dest_tabs_magic_smo	xt	Boolean	DEST.TABS.MAGIC.SMSO

Terminal Variables (Continued)

The Wide Zero Parameter in UniVerse

UniVerse performs all arithmetic using double-precision floating-point numbers. Floating point is a system for representing numbers in a computer. This appendix describes floating-point numbers and explains why UniVerse has a user-configurable wide zero. Most of this discussion assumes the Institute of Electrical and Electronics Engineers (IEEE) standard floating-point format, but the principle, if not the actual arrangement of bits, applies to all machines.

Number Systems

Most people use the decimal system for representing numbers. This system uses the numerals 0 through 9 to represent the numbers 0 through 9, and uses a place value system to represent numbers larger or smaller. For example, the numeral 12 (in base 10) means 1 times 10 to the first power plus 2 times 10 to the 0 power (any number to the 0 power equals 1) for a total of 12. The decimal point allows negative powers of 10 to represent quantities less than 1. The numeral 1.2 means 1 times 10 to the 0 power plus 2 times 10 to the minus one power (or one-tenth). Only numbers that can be expressed in terms of sums of integers times powers of 10 can be exactly represented in a finite number of digits. The quantity one-third cannot be represented exactly in base 10 any more than you can divide a dollar three ways without a penny left over.

Computers use the binary system for representing numbers. This system uses only two numerals, 0 and 1. In the machine, these are represented by differing levels of electrical voltage. The numeral 101_2 means 1 times 2 to the second power (4_{10}) plus 0 times 2 to the first power plus 1 times 2 to the 0 power, for a total of 5. Like base 10, base 2 can be used for quantities less than 1 if you introduce the concept of the binary point. Thus 1.1_2 means 1 times 2 to the 0 power plus 1 times 2 to the minus one power (or $1/2$), for a total of $1\frac{1}{2}$. Only numbers that can be built out of sums of powers of 2 can be represented exactly in binary notation.

People using base 10 have sometimes found the place value system inconvenient for representing very large or very small numbers. Scientific notation is a scheme that uses a two-part numeral. The first part, called the *mantissa*, is used to represent the significant digits, which are the nonzero digits of the number. The second part of the numeral is the *exponent*, which represents the magnitude of the number. For example, in scientific notation you can write $2.3E10$ to represent 23,000,000,000.00.

Floating-Point Numbers

Similarly, computers use a floating-point system, which keeps the absolute size of a number separate from the significant bits. (A bit is a Binary digIT.) In a floating-point numeral, 1 bit, called the sign bit, is usually reserved to indicate whether or not the number is greater than or less than 0. Some number of bits are reserved for the exponent part of the numeral. This is the power of 2 to which the remaining bits (the mantissa) need to be raised to yield the actual value.

The IEEE has defined a format for double-precision floating-point numbers that is 64 bits long. The leftmost bit is reserved as the sign bit. This bit is set (or true or 1) if the number being represented is negative; it is cleared (or 0) if the number is positive. The next 11 bits are used for the exponent of the number. This exponent is *biased*; that is, to determine its actual value you must subtract a constant number from this number to obtain the actual value. This technique is used because the range of numbers representable by these bits is not the most desirable range. By adding a bias you can move the range of numbers representable to the range you would like to represent. In the case of IEEE floating-point, the constant is 1023₁₀. The rest of the bits are used to store the mantissa of the number. Knowing that, you can examine the double-precision floating-point representation of a decimal number.

Consider the decimal number 49.75. 49 comprises a 32, a 16, and a 1. In binary that would be 110001_2 . The fractional part comprises one-half and one-fourth, so the whole number in binary would be 110001.11_2 . Since the mantissa is always stored as a fraction, you must move the binary point six places to the left to make the significant bits of the number fractional. Storing the mantissa as a fraction allows an arbitrary number of trailing zero trailing bits. The floating-point system provides for an extra bit of precision by always shifting one place less than needed so that the high-order bit is always 1 and therefore need not be stored. Shifting the binary point to the left five places instead of six adds five to the exponent (already 1023_{10} from the bias) so the exponent value becomes 1028_{10} or 10000000100_2 . The final floating-point number becomes

[illegible]

or

```
0100000001001000111000000000000000000000000000000000000000000000
```

Obviously, working in binary can become tedious. Hexadecimal notation is a system that uses the numerals 0 through 9 and the letters A through F to represent numbers from 0 through 15. In this way a single character can represent four bits. By separating the bits in the number above into groups of four and using hexadecimal notation, you can write that number as 4048E0000000₁₆.

In any base there will be some numbers that cannot be represented exactly. The 64-bit floating-point numbers UniVerse uses provide an enormous amount of precision, so that calculations involving numbers which must be approximated still give answers correct to many decimal places. There is, however, one problem that needs to be addressed: number comparison.

The UniVerse Wide Zero Feature

Computers compare numbers by subtracting one from the other and looking at the difference. If there are any bits set in the difference, the numbers are not equal. Even though the internal representation of a number may be an approximation so close to exact that the error is never noticed by the user, the question of equality is always binary. That is, the numbers are equal or they are not. The computer does not have a concept of “close enough.”

But UniVerse does have a “close enough.” This is the wide zero feature. It insulates users from the tiny errors introduced by representing numbers in a finite number of bits. When UniVerse compares two numbers for equality, it looks at the difference between the numbers and decides if the difference is large enough to consider the numbers not equal. The value that is used to determine whether or not a difference is large enough is called the *wide zero*. The default wide zero UniVerse uses allows very small differences between numbers to be considered nonzero, but it is not adjusted so finely that what are essentially “noise” bits will affect results.

In the unlikely event that you need to adjust this value, the wide zero is configurable. That is, users can change the value that UniVerse uses to decide when the difference between two numbers is so small that it should be considered 0.

When comparing two numbers, UniVerse examines the exponent of the difference between them. If the exponent of the difference indicates that the difference is a very small number, then UniVerse concludes that the numbers are equal. As shipped, UniVerse assumes that any difference smaller than $2.91\text{E}-11_{10}$ is equal to 0. This provides precision somewhat greater than the 10 decimal digits advertised by UniVerse BASIC programs. The default wide zero mask for IEEE-compliant machines is $3\text{dc}00000_{16}$.

The following table lists the exponents of 1 times various powers of 10. Use this table as a guide to configure your wide zero value. Since the numbers are represented in binary, the exact point where the numbers become equal to 0 is at power of two boundaries. For example, with a mask of $0\times 3\text{fb}00000$ all numbers from .062 to –.062 test equal to 0.



Note: If you change the value of this mask, test the behavior of the system thoroughly before using it.

Power of Ten	Mask
0	0x3ff00000
-1	0x3fb00000
-2	0x3f800000
-3	0x3f500000
-4	0x3f100000
-5	0x3ee00000
-6	0x3eb00000
-7	0x3e700000
-8	0x3e400000
-9	0x3e100000
-10	0x3dd00000
-11	0x3da00000
-12	0x3d700000
-13	0x3d300000
-14	0x3d000000
-15	0x3cd00000
-16	0x3c900000

To adjust this value in your version of UniVerse, see Chapter 4, “[Local and System Files](#).” The value in the configuration file is expressed in hexadecimal notation and represents the high-order, 32 bits of the mask. The mask must be expressed in hexadecimal, and the initial 0x is required.



Warning: *If your default mask differs from that shown previously, your machine is not IEEE-compliant. Extra care must be taken in these cases. In any event, changing this value can have a serious impact on your system. We do not advise you to change the default values without consulting IBM Customer Support.*

Fault Numbers and Error Codes

Fatal errors in UniVerse often cause the terminal to display messages containing numeric error codes. The meaning of these codes is explained in this appendix.

Fault Numbers

The following table lists the fault numbers associated with the message:

```
Abnormal termination of UniVerse.  
Fault type is %d. Layer type is %s.
```

This message indicates a bug in UniVerse and should be reported to IBM Customer Support exactly as it appears, along with the process that caused the error. The fault type (%d) is the same as those generated by the UNIX *signal(3C)* facility. The layer type (%s) is important only to the IBM engineer who is to fix the bug.

The starred signals cause a core image to be saved.

<i>signal</i> Value	Description
1	(SIGHUP) hangup
2	(SIGINT) interrupt
3*	(SIGQUIT) quit
4*	(SIGILL) illegal instruction
5*	(SIGTRAP) trace trap
6*	(SIGIOT) IOT instruction
7*	(SIGEMT) EMT instruction
8*	(SIGFPE) floating-point exception
9	(SIGKILL) kill (cannot be caught or ignored)
10*	(SIGBUS) bus error
11*	(SIGSEGV) segmentation violation
12*	(SIGSYS) bad argument to system call
13	(SIGPIPE) write on a pipe with no one to read it
14	(SIGALRM) alarm clock
15	(SIGTERM) software termination signal

Fault Type for Abnormal UniVerse Termination

<i>signal</i> Value	Description
16	(SIGURG) urgent condition present on socket
17	(SIGSTOP) stop (cannot be caught or ignored)
18	(SIGTSTP) stop signal generated from keyboard
19	(SIGCONT) continue after stop
20	(SIGCHLD) child status has changed
21	(SIGTTIN) background read attempted from control terminal
22	(SIGTTOU) background write attempted to control terminal
23	(SIGIO) I/O is possible on a descriptor
24	(SIGXCPU) CPU time limit exceeded
25	(SIGXFSZ) file size limit exceeded
26	(SIGVTALRM) virtual time alarm
27	(SIGPROF) profiling timer alarm
Fault Type for Abnormal UniVerse Termination (Continued)	

Fatal Error Codes

Some fatal errors in UniVerse are reported in a message such as the following:

```
A fatal error has occurred in UniVerse
%e
```

The error number (*%e*) is the same as those generated by the UNIX system calls described in *intro(2)*. A list of these errors follows:

1. **(EPERM) Not owner.** Typically this error indicates an attempt to modify a file in some way forbidden except to its owner or to a UniVerse Administrator. It is also returned when ordinary users try to do things allowed only to UniVerse Administrators. Use the *ls -l* command to check file permissions.
2. **(ENOENT) No such file or directory.** This error occurs when a filename is specified and the file should exist but doesn't, or when one of the directories in a pathname does not exist. Make sure the filename or pathname is valid and that it is correctly typed.
3. **(ESRCH) No such process.** The process whose number was given to kill and *ptrace* does not exist or is already dead. Use the *ps -ae* command to check the numbers of all current processes.
4. **(EINTR) Interrupted system call.** An asynchronous signal (such as interrupt or quit), which the user has elected to catch, occurred during a system call. If execution is resumed after processing the signal, it appears as if the interrupted system call returned this error condition.
5. **(EIO) I/O error.** Some physical I/O error occurred during a read or write. This error may in some cases occur on a call following the one to which it actually applies.
6. **(ENXIO) No such device or address.** I/O on a special file refers to a subdevice which does not exist or is beyond the limits of the device. It may also occur when, for example, an illegal tape drive unit number is selected or a disk pack is not loaded on a drive.
7. **(E2BIG) Argument list too long.** An argument list longer than the maximum allowed by your system is presented to *execve*.
8. **(ENOEXEC) Execute format error.** A request is made to execute a file which, although it has the appropriate permissions, does not start with a valid magic number.

9. **(EBADF) Bad file number.** Either a file descriptor refers to no open file, or a read (resp. write) request is made to a file which is open only for writing (resp. reading).
10. **(ECHILD) No children.** Wait, and the process has no living or unwaited-for children.
11. **(EAGAIN) No more processes.** In a fork, the system's process table is full, or the user is not allowed to create any more processes. This parameter may be tunable.
12. **(ENOMEM) Not enough core.** During an *execve* or *break*, a program asks for more core or swap space than the system is able to supply. A lack of swap space is normally a temporary condition, but a lack of core is not a temporary condition. The maximum size of the text, data, and stack segments is a system parameter and may be tunable.
13. **(EACCES) Permission denied.** An attempt was made to access a file in a way forbidden by the protection system.
14. **(EFAULT) Bad address.** The system encountered a hardware fault in trying to access the arguments of a system call.
15. **(ENOTBLK) Block device required.** A plain file was mentioned where a *ck* device was required.
16. **(EBUSY) Mount device busy.** An attempt was made to mount a device that was already mounted or to dismount a device on which there is an active file directory (open file, current directory, home directory, mounted-on file, active text segment).
17. **(EEXIST) File exists.** An existing file was mentioned in an inappropriate context, such as a link.
18. **(EXDEV) Cross-device link.** A hard link to a file on another file system was attempted.
19. **(ENODEV) No such device.** An attempt was made to apply an inappropriate system call to a device, for example, to read a write-only device.
20. **(ENOTDIR) Not a directory.** A nondirectory was specified where a directory is required, for example, in a path or as an argument to *chdir*.
21. **(EISDIR) Is a directory.** An attempt was made to write on a directory.
22. **(EINVAL) Invalid argument.** Here are some examples of invalid arguments: dismounting a nonmounted device, mentioning an unknown signal in signal, reading or writing a file for which seek has generated a negative pointer. EINVAL is also set by math functions.

23. **(ENFILE) File table overflow.** The system's table of open files is full, and temporarily no more opens can be accepted.
24. **(EMFILE) Too many open files.** Customary configuration limit is 62 per process. This parameter may be tunable.
25. **(ENOTTY) Not a typewriter.** The file mentioned in an *ioctl* is not a terminal or one of the other devices to which these calls apply.
26. **(ETXTBSY) Text file busy.** An attempt was made to execute a pure-procedure program that is currently open for writing (or reading), or to open for writing a pure-procedure program that is being executed.
27. **(EFBIG) File too large.** The size of a file exceeded the maximum (about 1.0e9 bytes).
28. **(ENOSPC) No space left on device.** During a write to an ordinary file, there is no free space left on the device.
29. **(ESPIPE) Illegal seek.** An *lseek* was issued to a pipe. This error may also be issued for other nonseekable devices.
30. **(EROFS) Read-only file system.** An attempt to modify a file or directory was made on a device mounted read-only.
31. **(EMLINK) Too many links.** An attempt to make more hard links to a file than are allowed by your system.
32. **(EPIPE) Broken pipe.** A write on a pipe or socket for which there is no process to read the data. This condition normally generates a signal. The error is returned if the signal is ignored.
33. **(EDOM) Math argument.** The argument of a function in the *math(3M)* package is out of the domain of the function.
34. **(ERANGE) Result too large.** The value of a function in the *math(3M)* package is unrepresentable within machine precision.
35. **(EWOULDBLOCK) Operation would block.** An operation that would cause a process to block was attempted on an object in nonblocking mode.
36. **(EINPROGRESS) Operation now in progress.** An operation that takes a long time to complete was attempted on a nonblocking object.
37. **(EALREADY) Operation already in progress.** An operation was attempted on a nonblocking object that already had an operation in progress.
38. **(ENOTSOCK) Socket operation on non-socket.**
39. **(EDESTADDRREQ) Destination address required.** A required address was omitted from an operation on a socket.

- 40. **(EMSGSIZE) Message too long.** A message sent on a socket was larger than the internal message buffer.
- 41. **(EPROTOTYPE) Protocol wrong type for socket.** A protocol was specified that does not support the semantics of the socket type requested. For example, you can't use the ARPA Internet UDP protocol with type SOCK_STREAM.
- 42. **(ENOPROTOOPT) Bad protocol option.** A bad option was specified in a *getsockopt(2)* or *setsockopt(2)* call.
- 43. **(EPROTONOSUPPORT) Protocol not supported.** The protocol has not been configured into the system, or no implementation for it exists.
- 44. **(ESOCKTNOSUPPORT) Socket type not supported.** The support for the socket type has not been configured into the system, or no implementation for it exists.
- 45. **(EOPNOTSUPP) Operation not supported on socket.** Such an operation might be an attempt to accept a connection on a datagram socket.
- 46. **(EPFNOSUPPORT) Protocol family not supported.** The protocol family has not been configured into the system, or no implementation for it exists.
- 47. **(EAFNOSUPPORT) Address family not supported by protocol family.** An address incompatible with the requested protocol was used. For example, you shouldn't necessarily expect to be able to use PUP Internet addresses with ARPA Internet protocols.
- 48. **(EADDRINUSE) Address already in use.** Only one use of each address is normally permitted.
- 49. **(EADDRNOTAVAIL) Can't assign requested address.** This normally results from an attempt to create a socket with an address not on this machine.
- 50. **(ENETDOWN) Network is down.** A socket operation encountered a dead network.
- 51. **(ENETUNREACH) Network is unreachable.** A socket operation was attempted to an unreachable network.
- 52. **(ENETRESET) Network dropped connection on reset.** The host you were connected to crashed and rebooted.
- 53. **(ECONNABORTED) Software caused connection abort.** A connection abort was caused internal to your host machine.
- 54. **(ECONNRESET) Connection reset by peer.** A connection was forcibly closed by a peer. This normally results from the peer executing a *shutdown(2)* call.

55. **(ENOBUFFS) No buffer space available.** An operation on a socket or pipe was not performed because the system lacked sufficient buffer space.
56. **(EISCONN) Socket is already connected.** A connect request was made on an already connected socket, or a *sendto* or *sendmsg* request on a connected socket specified a destination other than the connected party.
57. **(ENOTCONN) Socket is not connected.** A request to send or receive data was disallowed because the socket is not connected.
58. **(ESHUTDOWN) Can't send after socket shutdown.** A request to send data was disallowed because the socket had already been shut down with a previous *shutdown(2)* call.
59. **Unused.**
60. **(ETIMEDOUT) Connection timed out.** A connect request failed because the connected party did not properly respond after a period of time. (The timeout period is dependent on the communication protocol.)
61. **(ECONNREFUSED) Connection refused.** No connection could be made because the target machine actively refused it. This usually results from trying to connect to a service that is inactive on the foreign host.
62. **(ELOOP) Too many levels of symbolic links.** A path lookup involved more than eight symbolic links.
63. **(ENAMETOOLONG) File name too long.** A component of a pathname exceeded 247 characters, or an entire pathname exceeded 1023 characters.
64. **(ENOTEMPTY) Directory not empty.** A directory with entries other than “.” and “..” was supplied to a remove directory or rename call. Use either the *mv* command to move, or the *rm* command to remove, all remaining files from the directory, then try to remove or rename the directory.

Initialization Errors

The following error message is sometimes displayed during execution of *DBsetup* or during the initialization phase of UniVerse:

```
An error has occurred during UniVerse initialization
Please contact the system administrator
Error code: %i %e
```

%e is one of the codes listed in “[Fatal Error Codes](#)” on page 4. *%i* is one of the following:

1. **Unable to create a signature.** The *getid* system call failed.
2. ***DBsetup* was unable to create the Disk Shared Memory Segment.** Check the maximum allowable shared memory segment size in the UNIX kernel configuration to see if it is set to the minimum specified in the UniVerse release notes.
3. ***DBsetup* was unable to create the file lock semaphore.**
4. **System V semaphores are probably not configured in the kernel.**
5. ***DBsetup* or *uv* was unable to attach to the Disk Shared Memory Segment.** Use the *ipcs* command to see if the Disk Shared Memory Segment (the one with an ID of the form 0xacecrrrr) exists.
6. **Attempt to initialize UniVerse after expiration date.** Check the expiration date of your license. You may need to order an extension of the license.
7. ***DBsetup* was unable to map the ALM file (Sequent DYNIX only).**
DBsetup was unable to open the ALM file (Sequent DYNIX only).
8. ***DBsetup* was unable to set close on exec for the ALM file (Sequent DYNIX only).**

Index

Numerics

64BIT_FILES parameter 4-4

A

Account Admin window 6-4

accounts

see also UniVerse accounts

flavors 6-5, 6-15, 6-19, A-96

IDEAL 6-19

IN2 6-19

INFORMATION 6-19

PICK 6-19

PIOPEN 6-19, A-20

importation function A-78

restoring 7-3

restoring from tape A-82

restoring from UNIX shell 7-8

transferring 7-2, 7-3

Accounts menu A-16, A-19, A-22

ACCOUNT-SAVE

command A-79, A-82

tape 7-3, 7-6, 7-11, 7-12, 7-13, 7-14,
A-84

ACCOUNT.FILE.STATS

command 1-9

acct.restore command 7-3, 7-6, 7-8,
7-10, 7-11, 7-12

ACCT.RESTORE screen A-82

acct.restore.exe program 7-15

Activate Files for Logging

menu A-50 to A-51

activating recoverable files A-50

activity log file 11-21

ACTLIST command 1-13

act.log file 11-21

adding

local machine user 19-17

new user 19-16

new users 5-4

nodes A-9

printer groups A-69

UniVerse accounts A-22

user accounts A-22

user groups 5-3

administering

users 19-15

administering UniRPC 19-4

ALLOWNFS parameter 4-4

ALTER.TABLE command 13-40

ANALYZE.FILE command 1-9

ANALYZE.SHM command 1-12,
14-8

application conversion functions A-96

APP.PROGS.O file 1-7

archive mode

setting A-46

archive types

TAPE A-46

AVAIL command 1-12, 14-16

B

B-tree files 1-6

background processes, logging

out 14-7

backing up

Full log files A-49

backing up files 12-7

backups

see also hot standby

- backup
 - information about 15-10
- Backup menu A-28
- backups 12-7
 - full 12-3
 - incremental 12-13
 - strategies for 12-3
 - using multiple devices 12-10
 - using multiple tapes 12-9
- BASIC command 18-3
- BASIC object code 18-3
- BASIC program
 - defining which to run in shared memory 18-11
 - removing from shared memory 18-17
 - updating in shared memory 18-16
- BASIC programs 18-2
 - adding applications 18-3
 - loading into shared memory 18-13
- bin* directory 1-6, 11-35
- bin/uv* command A-1
- BLKMAX parameter 4-4
- block size 12-8
- booting 3-10, 3-11
- Bourne shell (*sh*) A-20
- BP file 1-7, 18-3
- BP.O file 1-7

C

- CAT file 1-7
- CATALOG command 18-4
- catalog shared memory 18-9
 - loading programs into 18-13, A-75
 - modifying programs in 18-16, A-75
 - setting up 18-9
 - using programs in 18-15
- catalog shared memory segment
 - removing 18-18
- catalog space 18-3, 18-7
 - initializing 18-5
 - managing 18-5
- catdir*
 - directory 1-6
 - file 18-5, A-72
- CENTURYPIVOT parameter 4-5
- changing

- node ID A-10
- node name A-10
- port number A-11
- printer groups A-69
- UniVerse accounts A-24
- user accounts A-22
- user groups 5-3, A-18
- CHAP command 1-11
- checking for deadlocks 9-17
- checkpoint information file A-48
- checkpoint mode
 - setting A-46
- chmod* command 8-3, A-62, A-92
- clearing locks 9-12
- CLEAR.FILE command 13-40
- client-side licensing, *see* device licensing
- CNAME command 13-40
- command history
 - using 16-8
- commands
 - aborting 16-7
 - editing 16-8
 - executing from UniVerse
 - Admin 16-3
 - reexecuting 16-8
 - saving 16-9
 - and UniVerse Command Output window 16-6
- compacting directories 14-18
- CONFIG command 1-12, 4-21
- configurable parameters 4-4, 4-22
 - 64BIT_FILES 4-4
 - ALLOWNFS 4-4
 - BLKMAX 4-4
 - CENTURYPIVOT 4-5
 - CSHDISPATCH 4-5
 - DOSDISPATCH 4-5
 - EXACTNUMERIC 4-5
 - FLTABSZBLK 4-5
 - FSEMNUM 4-6
 - GLTABSZ 4-6
 - GSEMNUM 4-6
 - HISTSTKP 4-6
 - ISOMODE 4-6
 - JOINBUF 4-6
 - LAYERSEL 4-6
 - LOGBLNUM 4-6
 - LOGBLSZ 4-6

- LOGSYCNT 4-7
- LOGSYINT 4-7
- MALLOCTRACING 4-7
- MAXERRLOGENT 4-7
- MAXKEYSIZE 4-7
- MAXRLOCK 4-7
- MFILES 4-7, 19-5
- MODFPTRS 4-7
- modifying 4-19
- NETTIME 4-7
- NLSDEFDEVMAP 4-8
- NLSDEFDIRMAP 4-8
- NLSDEFFILEMAP 4-8
- NLSDEFGCIMAP 4-8
- NLSDEFPTRMAP 4-8
- NLSDEFSEQMAP 4-8
- NLSDEFSSRVLC 4-8
- NLSDEFSSRVMAP 4-8
- NLSDEFTERMMP 4-9
- NLSLCDEF 4-9
- NLSLCMODE 4-9
- NLSMODE 4-9
- NLSNEWDIRMAP 4-9
- NLSNEWFILEMAP 4-9
- NLSOSMAP 4-9
- NLSREADELSE 4-9
- NLSWRITEELSE 4-10
- OCVDATE 4-10
- OPENCHK 4-10
- OPTMEM 4-10
- PAKTIME 4-10
- PICKNULL 4-11
- PIOPENDEFAULT 4-11
- PKRJUS 4-11
- PROCACMD 4-11
- PROCPRMT 4-11
- PROCRCMD 4-11
- PSEMNUM 4-11
- QBREAK 4-12
- QDEPTH 4-12
- QSBRNCH 4-12
- QSDEPTH 4-12
- QSMXKEY 4-12
- QSRUNSZ 4-12
- RLOWNER 4-12
- RLTABSZ 4-12
- saving 4-18
- SCRMAX 4-13
- SCRMIN 4-13

SCRSIZE 4-13
 SELBUF 4-13
 SHDISPATCH 4-13
 SYNCALOC 4-14
 T30FILE 4-14
 THDR512 4-14
 TSTIMEOUT 4-14
 TXMEM 4-15
 TXMODE 4-15
 UDRBLKS 4-16
 UDRMODE 4-16, 13-7, 13-38,
 13-39
 ULIMIT 4-15
 UVSPPOOL 4-16
 UVSYNC 4-16
 UVTEMP 4-16
 VDIVDEF 4-17
 WIDE0 4-17
 Configure Log Resources
 menu A-45 to A-46
 configuring
 hot standby subscriber 13-35
 printers 10-2, 11-4, A-90
 replication 13-12, 13-13
 subscribers 13-24
 tape devices 10-2, 10-5, A-87
 terminals 10-16, 10-28
 connecting to a UniVerse server 20-6
 directly 20-6
 multiple-tier 20-6
 two-tier 20-6
 connection timeout 19-9
 Control Panel (UniVerse)
 starting services 3-5
 stopping services 3-7
 Control Panel (Windows NT)
 starting services 3-6
 stopping services 3-7
 CONVERT.ACCOUNT
 command A-96
 CORE command 1-12
 CREATE.LDIR command 1-13, A-45
 CREATE.LFILE command 1-13,
 A-45
 creating
 log directory A-45
 log files A-45
 creating backup image 7-22
 cron command 14-19

CSHDISPATCH parameter 4-5
 customizing
 NEWACC files 6-20
 VOC files 6-19

D

daemons
 spooler 11-33
 unirpcd 19-4, A-8
 uvdlockd 9-13
 damaged files
 repairing 15-17
 data backups 12-7
 data entry screens A-2
 dd filters 7-12
 deactivating recoverable files A-50
 DEACTLIST command 1-14
 Dead Locks Administration
 window 9-14
 deadlock daemon (*uvdlockd*),
 administering A-12
 deadlock manager 9-13
 deadlocks
 checking 9-17
 log files 9-18
 managing 9-13
 pending 9-15
 resolution strategy 9-18
 state 9-15
 defining
 devices 10-13
 printers 11-23
 DELETE.CATALOG command 18-7
 DELETE.LFILE command 1-14, A-46
 deleting
 Available log files A-46
 device definitions 10-14
 information files A-48
 nodes A-10
 printer definitions 11-7
 printer groups 11-12, A-69
 rolled-forward log files A-53
 tape drive definitions 10-12
 user accounts A-22
 user groups 5-3, A-18
 DEL.RFILE command 1-14
 device drivers 11-38

complex shell scripts 11-40
 errors when opening 11-51
 and remote printing 11-40
 setting interface characteristics
 for 11-41
 using the Bourne shell 11-39
 using UNIX executables 11-38
 device licensing 20-3
 modes 20-3
 requirements 20-5
 device subkey 20-7
 devices
 defining 10-13
 deleting device definitions 10-14
 modifying device definitions 11-4
 viewing device definitions 10-14
 Devices menu A-85
 df command 14-15
 diagnostic level
 setting 15-14
 DICT.DICT file 1-7
 direct connection to UniVerse
 server 20-6
 disabling transaction logging A-47
 disk space and replication 13-41
 disk usage 14-15
 dispatch type field (VOC entry) 18-20
 displaying spooler log files 11-28
 domain user
 adding 19-16
 DOSDISPATCH parameter 4-5
 drivers (printer), definition A-62, A-92
 DROP.TABLE command 13-40
 du command 14-16

E

editing
 commands 16-8
 ENABLE.RECOVERY
 command 1-14, A-47
 enabling
 transaction logging A-47
 ENVIRONMENT command 1-11
 environment variables, TERM 10-15,
 10-16, 10-19
 errlog file 1-6
 Error and Activity Logs menu A-70

error codes
 fatal E-4
 fault numbers and E-1
errors
 clearing 3-11
 during initialization E-10
 logging 1-6
err.log file 11-21
EXACTNUMERIC parameter 4-5
exclusive file locks 9-7

F

fail-over mode 13-5, 13-35
 turning on 13-37
file
 SHM.TO.LOAD 18-11
file diagnostics
 running 15-13
 viewing errors 15-16
file locks 9-10
File Recovery menu A-53 to A-54
file statistics
 viewing information 15-11
file types
 B-tree 1-6
 hashed 1-5
 nonhashed 1-5
files
 activity log 11-21
 act.log 11-21
 APP.PROGS.O 1-7
 B-tree 1-6
 backing up 12-7
 base information 15-6
 BP 1-7, 18-3
 BP.O 1-7
 CAT 1-7
 catdir 18-5
 DICT.DICT 1-7
 errlog 1-6
 err.log 11-21
 gettydefs 10-16, 10-18, 10-19
 GLOBAL.CATDIR 18-5
 group 8-8, A-18
 hashed 1-5
 header information 15-7
 hosts 19-6
 inconsistent A-54
 inittab 10-16
 log 11-28
 LOGIN 6-17
 naming on UNIX 7-24
 naming on Windows NT 7-24
 NEWACC 1-7, 6-16, 6-19, 18-19
 NLS information 15-8
 nonhashed 1-5
 passwd A-21
 pqic.results 7-9
 printer configuration 11-35
 print_group 11-8, A-70
 PTERM.FILE 1-7
 publishing 13-12, 13-15, 13-17
 repairing damaged 15-17
 restoring 12-15
 rolling forward A-54
 services 19-11
 special 10-15
 spooler
 error log 11-21
 lock 11-48
 queue log 11-28
 sp.config 11-4, 11-35, 11-53,
 A-67 to A-69, A-85
 subscribing 13-27, 13-29
 SYS.MESSAGE 1-7, A-31, A-36
 unirpcservices 19-4, 19-9 to 19-10
 UniVerse account control 6-13
 unpublishing 13-21
 unsubscribing 13-32
 usplog 11-28, 11-37
 uvchkd.info A-48
 uvconfig 1-7, 4-2
 uvlogd.info A-48
 uvregen 4-2
 uvrolf.info A-48
 UV.ACCOUNT 1-7, 6-14, A-16
 UV.FLAVOR 1-7, 6-15
 UV.LOGIN 6-17
 UV.LOGINS 6-19
 uv.rc 3-10, 11-20, 11-35
 UV_LOGS A-47
 viewing information about 15-12
 VOC 5-3, 8-9, 8-10
 VOCLIB 8-12
 &DEVICE& 1-7, 10-3, A-46, A-61,
 A-85, A-87, A-90, A-91

 &MAP& 18-7
 &TEMP& 6-16
 &TRUNCATED& 7-10
 &UVFIX& 15-29
 .profile 8-5, 10-19, 10-28
 .uvconfig.bak 4-3
 /etc/group A-16
 /etc/hosts 19-4, A-9
 /etc/passwd A-16
 files that grow 14-17
 filters
 dd 7-12
 tapein 7-12
 find command 12-13, 14-16
 fixing files 15-29
 FLTABSZ parameter 4-5
 format conversion utility 15-21
 FORMAT.CONV command 15-21
 FSEMNUM parameter 4-6
 Full log files
 backing up and releasing A-49
 listing A-49

G

gcidir directory 1-6
getty program 3-9, 10-16, 10-17
gettydefs file 10-16, 10-18, 10-19
GLOBAL.CATDIR file 18-5
GLTABSZ parameter 4-6
group file 8-8, A-18
group locks 9-10
groups, *see* printer groups, user groups
GSEMNUM parameter 4-6

H

hashed files 1-5
HISTSTKP parameter 4-6
hold printer status 11-49
hosts file
 adding a new node 19-6
 modifying a node 19-7
hot backup, *see* hot standby
hot standby 13-3, 13-5, 13-12
 configuring subscriber 13-35
 fail-over mode 13-5, 13-35
 managing 13-5, 13-35 to 13-39

reconciling with publisher 13-37
 subscriber 13-5
 turning on fail-over mode 13-37
 hsrexec
 starting 19-19

I

IDEAL flavor 6-5, 6-7
 Import Account window 7-4
 Import menu A-78, A-82
 IN2 flavor 6-5, 6-7
 inconsistency flag A-54
 indexes
 information about 15-10
INfilter.exe program 7-20
 information files
 deleting A-48
 listing A-47, A-48
 subscribing 13-24
 INFORMATION flavor 6-5, 6-7
init process 3-9, 10-16
 initialization files
 LOGIN 6-17
 UV.LOGIN 6-17
 initialization process 3-9
 errors E-10
 INITIALIZE.CATALOG
 command 18-5
inittab file 10-16
 intent file locks 9-6
 IP address, modifying 19-7
 ISOMODE parameter 4-6

J

Job Control menu A-57, A-59
 job ID for print jobs 11-22
 Job pull-down menu 11-18
 JOINBUF parameter 4-6

K

keep alive parameters 19-12
 killing print jobs 11-25

L

label records on tape 7-14
 labels, removing 7-11
 LAN 20-5
 LAYERSEL parameter 4-6
 License Administration A-12
 licensing
 device 20-3
 license tool 20-8
 modes 20-3
 session 20-3
 LIMIT command 1-9
 listing nodes A-11
 LISTU command 1-12, 14-18
 Lock Administration window 9-9
 lock files A-65, A-88, A-95
 lock-waiter tables 9-19
 locks
 clearing 9-12
 compatibility 9-3
 deadlocks 9-13
 exclusive file lock 9-7
 file lock 9-10
 granularity 9-3
 group lock 9-10
 intent file lock 9-6
 managing 9-9
 shared file lock 9-6
 shared record lock 9-4
 transactions and 9-8
 types 9-3
 update record lock 9-5
 log directory
 creating A-45
 log files
 creating A-45
 deadlocks 9-18
 deleting A-46
 rolled-forward log files A-53
 listing Full log files A-49
 listing information files A-47
 restoring from tape A-52
 LOGBLNUM parameter 4-6
 LOGBLSZ parameter 4-6
 logging information file A-48
 logging level
 setting for telnet session 19-14
 logging out

background processes 14-7
 user processes 14-7
 LOGIN entry 3-3, 6-17, 8-5, 10-28,
 A-6, A-20
 default 6-17
 in UV account A-6
login program 3-9, 10-16
 logon banner
 specifying 19-14
 LOGSYCNT parameter 4-7
 LOGSYINT parameter 4-7
 LOGTO command 5-3
 LOG.RESTORE command 1-14
 long names
 support for 7-11
 truncation of 7-10
 LONGNAMES command 7-11
 LPTR keyword 11-36

M

magrst command 7-3, 7-6, 7-8, 7-9,
 7-11
 MAGRST screen A-79
magrst.exe program 7-15
 MAGSAV
 command A-79
 tape 7-3, 7-6, 7-12, 7-14, A-84
 MAIL command 1-11
 Maintain & DEVICE & File
 window 10-4
 MAKE.MAP.FILE command 18-6,
 18-7, 18-20
 MALLOCTRACING parameter 4-7
 Manage Log Transfers
 menu A-48 to A-50
 Manage Logging Activity
 menu A-46 to A-48
 managing
 deadlocks 9-13
 hot standby operations 13-5,
 13-35 to 13-39
 replication 13-3, 13-8, 13-12
 MAP command 18-6, 18-20
 MAXERRLOGENT parameter 4-7
 MAXKEYSIZE parameter 4-7
 MAXRLOCK parameter 4-7
 Media Recovery menu A-51 to A-53

menus A-1
 Accounts A-16, A-19, A-22
 Activate Files for
 Logging A-50 to A-51
 Backup A-28
 Configure Log
 Resources A-45 to A-46
 Devices A-85
 Error and Activity Logs A-70
 File Recovery A-53 to A-54
 Import A-78, A-82
 Job Control A-57, A-59
 Manage Log Transfers A-48 to A-50
 Manage Logging
 Activity A-46 to A-48
 Media Recovery A-51 to A-53
 Modify Job Characteristics A-58
 Package A-7
 Queue Management A-55, A-57
 Recovery A-27
 Restoration A-28
 Shared Memory A-72
 Spooler A-55, A-57, A-61, A-63,
 A-70, A-91, A-93
 System Administration A-1, A-6
 Transaction Logging A-44
 transaction logging A-44 to A-54
 user A-96
 MENU.FILE A-6
mesg command 17-6
 MESSAGE command 17-7
 message of the day 17-8
 messages
 sending on Windows platforms 17-5
 waiting for cataloged shared
 memory 18-14
 MFILES parameter 4-7, 19-5
 MKFILELIST command 1-14
 modes
 setting A-46
 MODFPTRS parameter 4-7
 Modify Job Characteristics menu A-58
 modifying
 configurable parameters 4-19
 device definitions 10-14
 IP address 19-7
 printer definitions 11-7
 tape drive definitions 10-12
 modifying programs

 in catalog shared memory 18-16
motd command 17-8
 MTU keyword 12-23
 multiple-tier connection to UniVerse
 server 20-6
 multiuser mode 3-9

N

naming
 files on UNIX 7-24
 files on Windows NT 7-24
 NETTIME parameter 4-7
 network services
 starting the UniRPC daemon 19-8
 stopping the UniRPC daemon 19-8
 NEWACC files 1-7, 6-13, 6-16, 6-19,
 18-19
 customizing 6-20
nls directory 1-6
 NLSDEFDEVMAP parameter 4-8
 NLSDEFDIRMAP parameter 4-8
 NLSDEFFILEMAP parameter 4-8
 NLSDEFGCIMAP parameter 4-8
 NLSDEFPTRMAP parameter 4-8
 NLSDEFSEQMAP parameter 4-8
 NLSDEFSRVLC parameter 4-8
 NLSDEFSRVMAP parameter 4-8
 NLSDEFTERMMAP parameter 4-9
 NLSLCDEF parameter 4-9
 NLSLCMODE parameter 4-9
 NLSMODE parameter 4-9
 NLSNEWDIRMAP parameter 4-9
 NLSNEWFILEMAP parameter 4-9
 NLSOSMAP parameter 4-9
 NLSREADELSE parameter 4-9
 NLSWRITEELSE parameter 4-10
 node ID, changing A-10
 node name, changing A-10
 nodes
 adding A-9
 deleting A-10
 listing A-11
 nonhashed files 1-5
 NORESET option A-65, A-95

O

OCVDATE parameter 4-10
 OPENCHK parameter 4-10
 OPTMEM parameter 4-10

P

Package menu A-7
 PAKTIME parameter 4-10
 PASSWD command 1-11, 8-6
passwd file A-21
 passwords 8-6
 assigning 8-6
 permissions 8-5
 assigning 8-5
 default 8-3
 Pick accounts, restoring from tape
 (Windows NT) 7-15
 PICK flavor 6-5, 6-7
 PICKNULL parameter 4-11
 PIOPEN flavor 6-5, 6-7, 6-19
 PIOPENDFAULT parameter 4-11
 PKRJJUST parameter 4-11
 port number 19-6
 changing A-11
 PORT.STATUS command 1-12, 14-6
 PostScript filter 11-39
pgic.results file 7-9
 Prime INFORMATION accounts,
 restoring from tape
 (Windows NT) 7-15
 print jobs
 changing characteristics of 11-22
 changing order of 11-20
 continuing suspended 11-27
 determining when printed 11-29
 directed to wrong print queue 11-50
 failing to go to Active state 11-48
 failing to print 11-50
 getting mixed up 11-56
 holding 11-25
 incorrect form 11-49
 killing 11-25
 outputting to a file 11-42
 printing in the wrong order 11-55
 releasing from Hold status 11-26
 reprinting 11-26

- retaining a print file 11-24
- setting priority 11-23
- specifying lines to print 11-24
- specifying pages to print 11-24
- specifying when to print 11-23
- suspending 11-27
- PRINT ON statement 11-36
- printer drivers 11-38
 - definition A-62, A-92
- printer groups 11-8
 - adding A-69
 - adding users or printers to 11-11
 - changing A-69
 - deleting 11-12, A-69
 - removing users or printers from 11-11
- printers
 - attaching a form 11-23
 - configuring 10-2, 11-4, A-90
 - defining 11-4, 11-23
 - deleting printer definitions 11-7
 - enabling and disabling printing A-63, A-93
 - forms 11-50, A-62, A-92
 - modifying 11-12
 - mounting a form 11-5, 11-13
 - printer configuration file 11-35
 - setting queuing options for 11-14
 - starting 11-14
 - status
 - hold 11-49
 - wait 11-50
 - stopping 11-14
 - viewing printer definitions 11-7
- PRINT.ADMIN command A-96
- print_group* file 11-8, A-70
- PROCACMD parameter 4-11
- processes, terminating 14-7
- processor mode field (VOC entry) 18-21
- PROCPRMT parameter 4-11
- PROCRCMD parameter 4-11
- PSEMNUM parameter 4-11
- PTERM command 10-28, A-65, A-95, B-1
- PTERM.FILE file 1-7
- publisher 13-4
 - reconciling hot standby with 13-37
- publishing files 13-12, 13-15, 13-17

- publishing system 13-13
 - starting 13-15
 - stopping 13-15

Q

- QBREAK parameter 4-12
- QDEPTH parameter 4-12
- QSBRNCH parameter 4-12
- QSDEPTH parameter 4-12
- QSMXKEY parameter 4-12
- QSRUNSZ parameter 4-12
- Queue Management menu A-55, A-57

R

- rc* 3-9
- rc.local* 3-9
- READL locks, *see* shared record locks
- READU locks, *see* update record locks
- REALITY accounts, transferring to UniVerse 7-4
- REALITY flavor 6-5, 6-7
- rebooting, *see* booting
- reconciling hot standby with publisher 13-37
- record locks 9-10
- recoverable files
 - activating A-50
 - deactivating A-50
 - listing status A-51
- recovering *.uvconfig* file 4-3
- Recovery
 - menu A-27
 - option A-27
- RECOVERY.CHECKPOINT
 - command 1-14, A-52
- RECOVERY.CONSISTENT
 - command 1-14, A-54
- reexecuting commands 16-8
- RELEASE.LFILE command 1-14, A-50
- releasing
 - Full log files for reuse A-49
 - full tape devices for reuse A-50
- removing
 - catalog shared memory segment 18-18
- labels from tapes 7-11
- replication 13-3 to 13-43
 - and ALTER.TABLE 13-40
 - and CLEAR.FILE 13-40
 - and CNAME 13-40
 - configuring 13-12, 13-13
 - configuring a subscriber 13-24
 - and disk space 13-41
 - and DROP.TABLE 13-40
 - fail-over mode 13-5
 - failure 13-43
 - hot standby 13-5, 13-12
 - managing 13-3, 13-8, 13-12
 - publishing 13-12
 - publishing files 13-15, 13-17
 - and RESIZE 13-40
 - restrictions 13-40
 - setting up 13-6 to 13-7
 - starting the publishing system 13-15
 - starting the subscribing system 13-25
 - stopping the publishing system 13-15
 - stopping the subscribing system 13-25
 - subscribing 13-12
 - subscribing files 13-27, 13-29
 - and triggers 13-40
 - unpublishing files 13-21
 - unsubscribing files 13-32
 - using the Replication window 13-8
 - using the subscribing system 13-24
- Replication window 13-8
- reprinting print jobs 11-26
- reset* command 10-28
- RESIZE command 13-40
- resolution strategy for deadlocks 9-18
- response time monitoring 14-18
- Restoration menu A-28
- restoring
 - accounts 7-3
 - accounts (UNIX) 7-8
 - backup image 7-23
 - files 12-15
 - choosing what to restore 12-19
 - reporting options 12-21
 - specifying the device 12-15
 - UVRestore window 12-17
 - viewing the backup index 12-20
- Pick accounts from tape A-82

Prime accounts from tape A-79
 restoring log files from tape A-52
 ReVise 18-4
 RLOWNER parameter 4-12
 RLTAbsZ parameter 4-12
 roll-forward information file A-48
 roll-forward recovery A-51
 rolling forward
 a single file A-54
 restored log files A-53
root 1-4, 2-2, A-1
 protecting 8-4
 RUN command 18-3

S

sample directory 1-6
sample/tapein.c 7-14
 saving
 commands 16-9
 configurable parameters 4-20
 SCRMAX parameter 4-13
 SCRMIN parameter 4-13
 SCRSIZE parameter 4-13
 security on UNIX systems 8-3
 security subroutines 8-11
 SELBUF parameter 4-13
 server (UniVerse) 20-5
 services
 eligible nodes 19-9
 names of 19-9
 pathnames of 19-9
 starting 19-19
 timeout 19-9
services file 19-11
 session licensing 20-3
 SETPTR command A-56
 setting
 transaction logging modes A-46
 setting up transaction logging A-45
 SET.LOG.ATTR command 1-14,
 A-46
 SET.TERM.TYPE command 10-25,
 10-26, 10-29
sh shell 6-18
 shared file locks 9-6
 shared memory 18-9
 analysis utility 14-8

changing memory allocation of A-75
 loading programs into A-75
 removing program from 18-17
 updating program in 18-16
 Shared Memory menu A-72
 shared record locks 9-4
 SHDISPATCH parameter 4-13
 shell startup files, */usr/ardent/uv/bin/uv* 8-10
 SHM.TO.LOAD file
 adding programs to 18-12
 defined 18-11
 removing programs from 18-12
shutdown command 3-11
 shutdown (system) 3-9, 3-10
 SHUTDOWN.RECOVERY
 command 1-14, A-47
 single-user mode 3-10
 special files 10-15
 SPOOL command 11-51, A-55, A-56
 Spooler
 menu A-55, A-57, A-61, A-63, A-70,
 A-91, A-93
 option A-61, A-66, A-70, A-91
 window 11-15
 spooler
 altering response time 11-20
 attaching a form to a job 11-23
 changing characteristics of print
 jobs 11-22
 changing order of print jobs 11-20
 continuing suspended print
 jobs 11-27
 daemon 11-35, 11-36
 management 11-31
 resetting when hung 11-31
 starting 11-31
 stopping 11-31
 defining a printer 11-23
 determining when a job was
 printed 11-29
 displaying log files 11-28
 enabling and disabling
 queuing 11-49, A-63, A-93
 enabling error logging 11-21
 error log files 11-21
 holding print jobs 11-25
 incorrect form 11-49
 killing print jobs 11-25

lock file 11-48
 moving spool directory 11-20
 moving spooler directory 11-34
 nonexistent driver 11-51
 printer groups 11-8
 printing to a file 11-42
 queue administration A-96
 releasing print jobs from Hold
 status 11-26
 reprinting a print job 11-26
 retaining a file in the print
 queue 11-24
 setting job priority 11-23
 specifying lines to print 11-24
 specifying number of copies to
 print 11-23
 specifying pages to print 11-24
 spool directory 11-33
 spool queue log file 11-28
 suspending print jobs 11-27
 using device drivers 11-38
 Spooler menu A-55, A-63, A-93
sp.config file 11-4, 11-35, 11-53,
 A-67 to A-69, A-85
 SQL and database security 15-26
sql/catalog directory 1-6
 starting
 printers 11-14
 publishing system 13-15
 spooler daemon 11-31
 subscribing system 13-25
 UniRPC daemon 19-8
 UniVerse Admin 2-3
 UniVerse Resource service 3-5
 startup (system) 3-9
 STATUS command 1-12, 14-18
 STATUS USERS command 10-16
 stopping
 printers 11-14
 publishing system 13-15
 spooler daemon 11-31
 subscribing system 13-25
 UniRPC daemon 19-8
 UniRPC service 3-6
 UniVerse Resource service 3-6
 UniVerse Telnet service 3-6
stty command 10-28, 11-41, B-1
 subkey 20-7
 SUBROUTINE statement 8-11

- subscriber 13-4
 - configuring hot standby 13-35
 - hot standby 13-5
- subscribers, configuring 13-24
- subscribing
 - and replication 13-12
- subscribing files 13-27, 13-29
- subscribing system 13-24
 - starting 13-25
 - stopping 13-25
- superuser, *see* *root*
- suspending
 - transaction logging A-47
- suspending print jobs 11-27
- SUSPEND.RECOVERY
 - command 1-14, A-47
- sync* command 3-11
- SYNCALOC parameter 4-14
- System Administration menu A-6
- System Administration
 - menus A-1 to A-96
- system backup A-30
- system security, *see* security on UNIX systems
- system shutdown 3-9, 3-10
- system startup 3-9
- SYSTEM.ADMIN command A-6
- SYS.MESSAGE file 1-7, A-31, A-36

T

- T30FILE parameter 4-14
- TANDEM command 1-12
- TAPE archive type A-46
- tape devices
 - configuring 10-2, 10-5, A-87
 - defining on UNIX systems 10-5
 - defining on Windows NT systems 10-9
 - deleting tape definitions 10-12
 - modifying tape definitions 10-12
 - releasing for reuse A-50
 - specifying A-46
 - viewing tape definitions 10-12
- tapein* command 7-12
 - modifying 7-14
- tapein.c*, compiling 7-14
- tapein.exe* program 7-17

- tar* command 12-27
- TCP/IP 19-4, 20-5
 - hosts file, *see* */etc/hosts*
- telnet connection
 - setting parameters 19-13
- telnet port number
 - displaying 19-11
- telnet session
 - changing port number 19-12
 - connection parameters 19-12
 - defining user policies 19-12
 - keep alive parameters 19-12
 - modifying parameters 19-12
 - user policy 19-12
- telnet sessions
 - controlling access to UniVerse 6-19
- TERM
 - command 10-25
 - environment variable 10-15, 10-16, 10-19
- termcap* C-1, C-7
- terminals 10-15, C-1
 - configuring 10-16, 10-28
- terminating processes 14-7
- terminfo* C-1, C-2, C-4, C-7
- terminfo* directory 1-7
- terminfo.src* C-1
- THDR512 parameter 4-14
- timeouts
 - open connection 19-9
- transaction logging 1-13, A-27
 - disabling A-47
 - enabling A-47
 - information 15-9
 - menus A-44 to A-54
 - setting modes A-46
 - setting up A-45
 - shutting down, *see* *disabling*
 - suspending A-47
- Transaction Logging menu A-44
- transactions, locks and 9-8
- transferring accounts 7-2
 - from UNIX to Windows NT 7-22
 - non-UniVerse accounts to UniVerse 7-3
- triggers 13-40
- troubleshooting the spooler
 - configuration changes do not take effect 11-53

- jobs do not go active 11-50
- TSTIMEOUT parameter 4-14
- tty* command 10-15
- two-tier connection to UniVerse
 - server 20-6
- TXMEM parameter 4-15
- TXMODE parameter 4-15

U

- UCI (Uni Call Interface) 20-3
- UDRBLKS parameter 4-16
- UDRMODE parameter 4-16, 13-7, 13-38, 13-39
- ULIMIT parameter 4-15
- UMASK command 1-11, 6-17, 8-3
- umask* command 8-3, 8-5
- UNASSIGN command 1-8
- UniObjects 20-3, 20-6
- UniObjects for Java 20-3, 20-6
- UniRPC
 - administering 19-4
 - BASIC administration
 - programs 19-4
 - port number A-11
- unirpc
 - starting 19-19
- UniRPC daemon (*unirpcd*) 19-4
 - starting 19-8, A-8
 - stopping 19-8, A-11
- UniRPC service
 - stopping 3-6
- unirpcd* daemon
 - starting A-8
 - stopping A-11
- unirpcd* daemon, *see* UniRPC daemon
- unirpcservices* file 19-4, 19-9 to 19-10
- UniVerse
 - controlling access to 6-18, 6-19
 - shutting down 3-3, 3-10
 - starting 19-19
 - starting up 3-3, 3-10
- UniVerse account control files 6-13
- UniVerse accounts 5-2, 6-4
 - adding A-22
 - changing A-24
 - creating on UNIX servers 6-4
 - creating on Windows NT servers 6-6

- deleting on UNIX servers 6-10
- deleting on Windows NT
 - servers 6-11
- description 6-3
- flavors 6-5
- modifying account details 6-9
- setting permissions for 6-6
- transferring from UNIX to
 - Windows NT 7-22
- viewing account details 6-9
- UniVerse Admin 2-2, 20-3
 - starting 2-3
- UniVerse Administrator,
 - definition 1-3
- UniVerse Command Output
 - window 12-20, 16-6
- UniVerse Command window 16-3
- UniVerse commands
 - ACCOUNT-SAVE A-82
 - ACCOUNT.FILE.STATS 1-9
 - ACTLIST 1-13
 - ANALYZE.FILE 1-9
 - ANALYZE.SHM 1-12, 14-8
 - AVAIL 1-12, 14-16
 - BASIC 18-3
 - CATALOG 18-4
 - CHAP 1-11
 - CONFIG 1-12, 4-21
 - CONVERT.ACCOUNT A-96
 - CORE 1-12
 - CREATE.LDIR 1-13, A-45
 - CREATE.LFILE 1-13, A-45
 - DEACTLIST 1-14
 - DELETE.CATALOG 18-7
 - DELETE.LFILE 1-14, A-46
 - DEL.RFILE 1-14
 - editing 16-8
 - ENABLE.RECOVERY 1-14, A-47
 - ENVIRONMENT 1-11
 - FORMAT.CONV 15-21
 - INITIALIZE.CATALOG 18-5
 - LIMIT 1-9
 - LISTU 1-12, 14-18
 - LOGTO 5-3
 - LOG.RESTORE 1-14
 - MAIL 1-11
 - MAKE.MAP.FILE 18-7
 - MAP 18-7
 - MESSAGE 17-7
 - MKFILELIST 1-14
 - PASSWD 1-11, 8-6
 - PORT.STATUS 1-12, 14-6
 - PRINT.ADMIN A-96
 - PTERM 10-28
 - RECOVERY.CHECKPOINT 1-14, A-52
 - RECOVERY.CONSISTENT 1-14, A-54
 - reexecuting 16-8
 - RELEASE.LFILE 1-14, A-50
 - RUN 18-3
 - saving 16-9
 - SET.LOG.ATTR 1-14, A-46
 - SET.TERM.TYPE 10-25, 10-26, 10-29
 - SHUTDOWN.RECOVERY 1-14, A-47
 - STATUS 1-12, 14-18
 - STATUS USERS 10-16
 - SUSPEND.RECOVERY 1-14, A-47
 - TANDEM 1-12
 - TERM 10-25
 - UMASK 1-11, 6-17, 8-3
 - UNASSIGN 1-8
 - USERS 1-12
 - uvbackup 12-12
 - uvdlockd 9-18
 - uvrestore 12-24
 - VCATALOG 18-6
- UniVerse Configuration Editor
 - window 4-19
- UniVerse file types, *see* file types
- UniVerse Resource service
 - starting 3-5
 - stopping 3-6
- UniVerse servers 20-5
 - connecting to a server 20-6
- UniVerse System Administration
 - menus, *see* System Administration menus
- UniVerse Telnet service
 - stopping 3-6
- UniVerse User Administration
 - window 14-7
- UNIX
 - naming files 7-24
 - security, *see* security on UNIX systems
- UNIX account environment 5-2
- UNIX commands
 - acct.restore* 7-3, 7-6, 7-8, 7-10, 7-11, 7-12
 - analyze.shm* 14-8
 - chmod* 8-3, A-62, A-92
 - cron* 14-19
 - df* 14-15
 - du* 14-16
 - find* 12-13, 14-16
 - magrst* 7-3, 7-6, 7-8, 7-9, 7-11
 - mesg* 17-6
 - newgrp* 8-8
 - reset* 10-28
 - shutdown* 3-11
 - stty* 10-28, 11-41, B-1
 - sync* 3-11
 - tapein* 7-12
 - tar* 12-27
 - tty* 10-15
 - umask* 8-3, 8-5
 - usa* 11-37
 - usm* 11-37
 - usp* 11-35
 - uv* 6-4
 - uvregen* 4-2
 - uv.rc* 3-10
 - who* 14-18
 - write* 17-6
- UNIX scheduler 11-51
- UNIX spooler 11-44
- unpublishing files 13-21
- unsubscribing files 13-32
- update record locks 9-5
- usa* command 11-37
- usd* spooler daemon 11-36
- user
 - adding 19-16
 - adding domain 19-16
 - adding local machine 19-17
- user accounts (UNIX) 5-2
 - adding A-22
 - changing A-22
 - deleting A-22
 - for groups 5-3
 - for individual users 5-3
- user groups 8-8
 - adding 5-3
 - changing 5-3, A-18

- deleting 5-3, A-18
- used with UniVerse accounts 8-9
- user menus A-96
- user policies
 - defining for telnet session 19-12
- users
 - adding 5-4
 - administering 19-15
 - deleting A-21
 - modifying A-21
- USERS command 1-12
- usm* command 11-37
- usp* command 11-35
- usplog* file 11-28, 11-34
- uv*
 - command 6-4
 - directory A-1
- UV account A-1
- UV account directory 4-21
 - errlog* 1-6
- uvbackup* command 12-12
- UVBACKUP screen A-30
- UVBackup window 12-7
- uvchkd.info* file A-48
- uvconfig* file 1-7, 4-2, 4-22, 11-34
- uvdlockd*
 - command 9-18
 - daemon, administering A-12
- uvfixfile* utility 15-28
- uvlctool* 20-3, 20-8
- uvlogd.info* file A-48
- uvmt.exe* program 7-18
- uvregen*
 - command 4-2
- uvregen* program 4-2
- uvrestore* command 12-24
- UVRESTORE screen A-35
- UVRestore window 12-17
- uvrolf.info* file A-48
- UVSPOOL parameter 4-16
- UVSYNC parameter 4-16
- uvtelnet
 - starting 19-19
- UVTEMP parameter 4-16
- UV.ACCOUNT file 1-7, 6-14, A-16
- UV.FLAVOR file 1-7, 6-15
- UV.LOGIN entry 3-3
- UV.LOGINS file 6-16, 6-19
- uv.rc*

- command 3-10
- file 3-9, 11-20, 11-35
- UV.TRANS file
 - listing A-51
- UV/Net timeout 19-9
- UV_LOGS file
 - listing A-47

V

- VCATALOG command 18-6
- VDIVDEF parameter 4-17
- viewing
 - device definitions 10-14
 - printer definitions 11-7
 - tape drive definitions 10-12
- VOC entry
 - dispatch type field 18-20
 - fields in 18-19
 - processor mode field 18-21
- VOC file 5-3, 8-9, 8-10
 - adding commands to 18-19
 - creation 6-16
 - customizing 6-19
- VOCLIB file 8-12

W

- wait printer status 11-50
- well-known port number, *see* port number
- who* command 14-18
- wide zero parameter D-1
- WIDE0 parameter 4-17
- Windows Print Manager 8-2, 11-3
- Windows NT
 - naming files 7-24
- word length 11-5
- write* command 17-6

Symbols

- &DEVICE& file 1-7, 10-3, A-46, A-61, A-85, A-87, A-90, A-91
- &MAP& file 18-7
- &TEMP& file 6-16
- &TRUNCATED& file 7-10

- &UVFIX& file 15-29
- .profile* file 8-5, 10-19, 10-28, A-20
- .uvconfig* file 4-2
 - recovering 4-3
- .uvconfig.bak* file 4-3
- /bin/sh* 6-18
- /dev* 10-15, 10-16
- /etc/gettydefs* 10-18
- /etc/group* A-16
- /etc/hosts* 19-4, A-9
- /etc/passwd* A-16
 - see also* group file
- /usr/ardent/uv/errlog* 1-6
- /usr/ardent/uv/NEWACC* 6-16, 6-20
- /usr/ardent/uv/sample* A-20
- /usr/ardent/uv/sample/uv.rc* 3-9