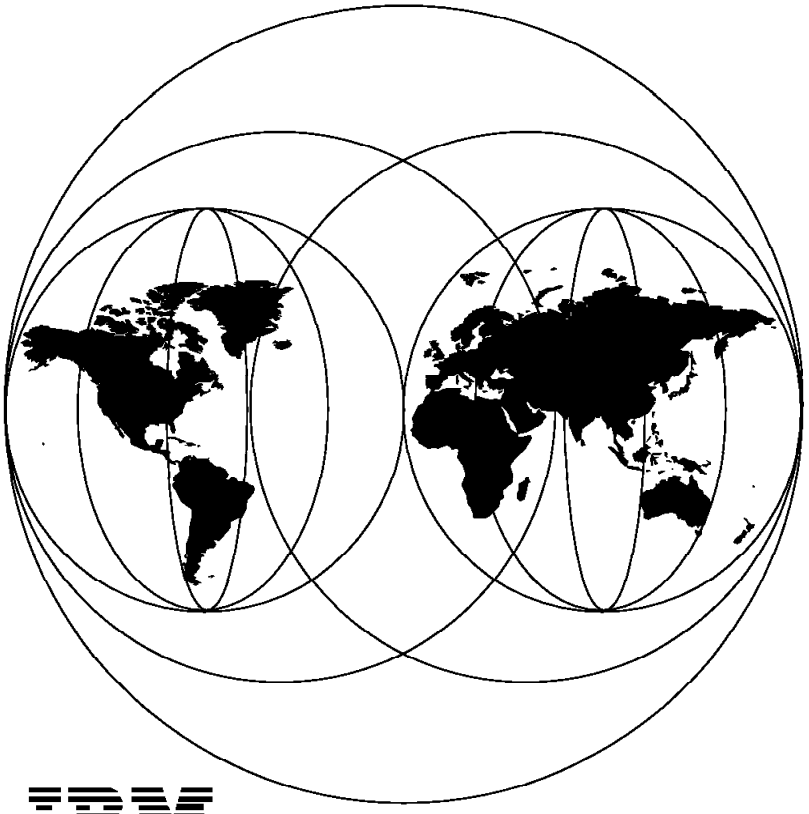
International Technical Support Organization

SG24-4601-00

ADSM for AIX: Advanced Topics

December 1995



IBM

International Technical Support Organization

Austin Center

International Technical Support Organization

SG24-4601-00

ADSM for AIX: Advanced Topics

December 1995

Take Note!

Before using this information and the product it supports, be sure to read the general information under “Special Notices” on page xiii.

First Edition (December 1995)

This edition applies to Version 2 Release 1 of ADSM, Program number 5765-564 and Version 1 Release 2 of SystemView for AIX, Program Number 5765-527 for use with the AIX Operating System

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

An ITSO Technical Bulletin Evaluation Form for reader′s feedback appears facing Chapter 1. If the form has been removed, comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. JN9B Building 045 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Abstract

This document examines some advanced aspects of ADSM for AIX. In particular, it focuses on two areas: disaster recovery management of the ADSM server and integration with SystemView. This book provides detailed information on planning for disaster recovery in the event of a variety of ADSM server problems and includes information on using the Disaster Recovery Manager. Theory and practical examples are provided. ADSM is shipped as part of the SystemView system management product set, and this book also provides detailed information on how ADSM can be more closely integrated with the other products, including theory and practical examples of various levels of integration with the product set.

This document was written for IBM customers, systems engineers and marketing representatives who require knowledge ranging from overview to detailed implementation examples on ADSM disaster recovery and SystemView integration. Some knowledge of ADSM, NetView/6000, NetView DM/6000, Trouble Ticket/6000, and AIX is assumed.

(187 pages)

Contents

Figures

Tables

Special Notices

This publication is intended to help IBM customers, systems engineers and marketing representatives to gain knowledge ranging from informative overviews to in-depth, practical examples of disaster recovery management of the ADSM server and product integration with the SystemView systems-management product set. The information in this publication is not intended as the specification of any programming interfaces that are provided by IBM SytemView for AIX or for ADSM for AIX. See the PUBLICATIONS section of the IBM Programming Announcement for IBM SystemView for AIX and ADSM for AIX for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

ADSTAR	AIX
AIX/6000	AIXwindows
IBM	NetView
RISC System/6000	RS/6000
SystemView	Trouble Ticket

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

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

Windows is a trademark of Microsoft Corporation.

Other trademarks are trademarks of their respective companies.

Preface

This document is intended to examine some of the more advanced aspects of ADSM for AIX. In particular, it focuses on two areas: disaster recovery management of the ADSM server and product integration with the SystemView system-management product set. An overview of disaster recovery in general is provided along with information to assist in the planning process. Detailed information on the practical steps required for effective disaster recovery procedures are then given, along with practical examples. Information on using the new Disaster Recovery Manager application is also included. Part two of this book then looks at the potential for enhanced ADSM integration with other appropriate applications from the SystemView product set. Details ranging from overview, through theory, to practical examples of integration are provided.

This document will be useful to IBM customers, systems engineers and marketing representatives in providing a wide range of advanced ADSM information from overviews to detailed implementation examples in the areas of disaster recovery and SystemView product integration.

How This Document is Organized

The document is organized as follows:

- Chapter 1, “Disaster Recovery Overview”

  This chapter provides details on the functions of the critical elements of the ADSM server and the ways in which these elements can be protected against unforeseen disaster. This chapter can be skipped by those readers who already have a detailed knowledge of the purpose and function of the ADSM database, recovery log and storage pools.

- Chapter 2, “Planning Overview”

  This chapter provides an overview of the major considerations in planning for disaster recovery. Information on the elements that should be protected as well as the way in which this protection should be implemented are provided. This chapter can be skipped by those readers who are already familiar with disaster recovery planning.

- Chapter 3, “On-site Processes”

  This chapter looks in detail at the processes that should be performed on-site to implement an effective disaster recovery plan. The considerations involved in selecting the specific on-site disaster recovery strategy, as well as examples of implementing the strategy are provided. This chapter should be read by everyone intending to implement on-site disaster recover processes.

- Chapter 4, “Off-site Processes”

  This chapter looks in detail at the processes that should be performed off-site to implement an effective disaster recovery plan. The considerations involved in selecting the correct strategy as well as examples of implementing strategies and their use are provided. This chapter should be read by everyone intending to implement off-site disaster recovery processes.

- Chapter 5, “Disaster Recovery Manager”

This chapter looks at the features and functions of a new feature of ADSM, the Disaster Recovery Manager. An overview of its purpose and examples of an implementation using DRM are provided. This chapter provides useful information on automating the disaster recovery process using DRM and should be read by anyone interested in disaster recovery.

- Chapter 6, “Disaster Recovery Scenarios”

  This chapter provides practical examples of various disastrous scenarios, including database failures, media failures and complete server failures. The steps required to recover from the disasters described by using the methodologies presented earlier in the book are shown. This chapter should be read by anyone interested in disaster recovery.

- Chapter 7, “ADSM in a Heterogeneous Network”

  This chapter provides an overview of how ADSM fits into the distributed network environment. It looks at some of the reasons why it is beneficial to achieve a higher degree of integration between the SystemView products and ADSM. This chapter should be read by anyone running ADSM in a distributed environment with SystemView.

- Chapter 8, “Product Overviews”

  This chapter provides an overview of each of the SystemView products, along with a brief description of the potential benefits of integration with ADSM. This chapter can be skipped by those readers who are already familiar with the products included with SystemView.

- Chapter 9, “Integrating ADSM”

  This chapter looks in more detail at the level of integration possible between ADSM and the SystemView products. For those products for which it makes sense to consider a higher degree of integration, this potential is explored, and an overview of an example scenario to illustrate this integration potential is provided. This chapter should be read by anyone who is considering, or who is already operating, ADSM in a distributed network with SystemView.

- Chapter 10, “NetView Scenarios”

  This chapter describes, in detail, the scenarios portrayed in overview in Chapter 9, “Integrating ADSM” on page 81 for NetView/6000. The theory and methods used to achieve integration, as well as two detailed examples, are provided. This chapter should be read by anyone considering, or who is already operating, ADSM in a distributed environment with SystemView.

- Chapter 11, “Trouble Ticket Scenarios”

  This chapter describes, in detail, the scenarios portrayed in overview in Chapter 9, “Integrating ADSM” on page 81 for Trouble Ticket/6000. The theory and methods used to achieve integration as well as a detailed example are provided. This chapter should be read by anyone considering, or who is already operating, ADSM in a distributed environment with SystemView.

- Chapter 12, “NetView DM Scenarios”

  This chapter describes, in detail, the scenarios portrayed in overview in Chapter 9, “Integrating ADSM” on page 81 for NetView DM/6000. The theory and methods used to achieve integration as well as three examples are provided. This chapter should be read by anyone considering, or who is already operating, ADSM in a distributed environment with SystemView.

Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this document.

- *NetView Distribution Manager/6000 User's Guide*, SH19-5003
- *ADSM for AIX Administrator's Guide*, SH35-0134

International Technical Support Organization Publications

- *NetView Distribution Manager/6000 Release 1.2 Agents and Advanced Scenarios*, GG24-4490
- *SystemView for AIX V1R1 Scenarios*, SG24-2564
- *IBM SystemView for AIX: An OverView*, GG24-2541

A complete list of International Technical Support Organization publications, known as redbooks, with a brief description of each, may be found in:

International Technical Support Organization Bibliography of Redbooks, GG24-3070.

To get a catalog of ITSO redbooks, VNET users may type:

```
TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
```

A listing of all redbooks, sorted by category, may also be found on MKTTOOLS as ITSOCAT TXT. This package is updated monthly.

How to Order ITSO Redbooks

IBM employees in the USA may order ITSO books and CD-ROMs using PUBORDER. Customers in the USA may order by calling 1-800-879-2755 or by faxing 1-800-445-9269. Almost all major credit cards are accepted. Outside the USA, customers should contact their local IBM office. For guidance on ordering, send a PROFS note to BOOKSHOP at DKIBMVM1 or E-mail to bookshop@dk.ibm.com.

Customers may order hardcopy ITSO books individually or in customized sets, called BOFs, which relate to specific functions of interest. IBM employees and customers may also order ITSO books in online format on CD-ROM collections, which contain redbooks on a variety of products.

ITSO Redbooks on the World Wide Web (WWW)

Internet users may find information about redbooks on the ITSO World Wide Web home page. To access the ITSO Web pages, point your Web browser to the following URL:

```
http://www.redbooks.ibm.com/redbooks
```

IBM employees may access LIST3820s of redbooks as well. The internal Redbooks home page may be found at the following URL:

```
http://w3.itsc.pok.ibm.com/redbooks/redbooks.html
```

Acknowledgments

This project was designed and managed by:

Nick Higham
International Technical Support Organization, Austin Center

The authors of this document are:

Nick Higham
IBM Austin

Mike Skidmore
IBM UK

Richard Catoire
IBM Netherlands

This publication is the result of a residency conducted at the International Technical Support Organization, Austin Center.

Thanks to the following people for the invaluable advice and guidance provided in the production of this document:

Cyndie Behrens
International Technical Support Organization, San Jose Center

Don Moxley
IBM Tucson

Additional thanks to the editor:

Marcus Brewer
International Technical Support Organization, Austin Center

Part 1. Disaster Recovery

Chapter 1. Disaster Recovery Overview

ADSM is a product that assists in the protection of data against accident and hardware or software failures. The disaster recovery element of ADSM that will be covered in this book is the part that helps to protect and recover the ADSM product itself in case of failures.

There are three critical elements for the correct functioning of ADSM.

- ADSM database
- Recovery log
- Storage pools

The roles of these three elements are now explained in more detail:

1. ADSM database

   The database manages information about the location of client backup and archive files residing in storage pools, and it records information for ongoing server operations.

   The ADSM database contains information on:

   - Client nodes
   - Administrators
   - Policies
   - Schedules
   - Server settings
   - Storage volume location for backed up, archived and migrated files

   The database also records information needed for the following server operations:

   - Access control information for administrator clients
   - Activity logs, which contains the messages generated by the server
   - Data storage inventory, used to allocate the files in the storage pools
   - Event records that are generated by the processing of scheduled commands
   - Information about registered clients
   - Information about ADSM volumes
   - Policies assigned to client nodes
   - Schedules and their associations with client nodes

2. Recovery log

   The recovery log is used to maintain a consistent database image by recording changes made to the database as a transaction proceeds. A transaction is any exchange between a client and the server. If a transaction completes successfully, database changes are committed, and permanent changes are made to the database. If a transaction fails, database changes are undone by removing them from the database and the recovery log.

   The recovery log is a buffer file in the server that:

- Manages updates that are a result of a client transaction
- Keeps track of all updates that have not yet been written to the database
- Plays a key role in restoring the database to a consistent state in case of a system failure
- Knows which transactions have completed successfully so that they are copied into the recovered database
- Ensures that incomplete transactions are deleted from the database during a recovery

3. Storage pools

   The storage pools contain the actual backup or archive copies of client files.

1.1 Data Base, Recovery Log and Storage Pool Relationships

Information stored in the ADSM database, recovery log and storage pool volumes are closely related. The database and the recovery log contain cross references to client transactions. The result of transactions is that information, files or instructions are sent to the storage pool volumes. It is therefore very important that this data is consistent.

If one of these three is corrupted, the ADSM server will become unavailable.

1.2 How to Protect Data

To assist in the protection and recovery of data, the following strategy should be implemented:

1. Mirror the recovery log and the database

   One of the most catastrophic failures in ADSM is the loss of the database or the recovery log. ADSM helps to protect against this by providing the capability to mirror the database and recovery log, which causes the data to be written to multiple disks simultaneously. However, mirroring does not protect against hardware failures that affect multiple drives or against the loss of the entire system. Administrator control of mirroring can be performed dynamically while ADSM is running.

   Mirroring provides the following advantages:

   - Protects against media failures within the database or recovery log by providing up to three exact copies
   - Allows database operations to continue without interruption if a database or recovery log volume fails by using a mirrored copy of the failed volume. The failed volume is placed offline and is automatically synchronized when repaired and brought back online.
   - Provides a way to avoid costly database recoveries
   - Increases the database read performance by allowing the server to read data from the nearest position on any available copy

   Of course, mirroring will require additional disk space for the mirror copies.

2. Back up the server database periodically on removable media and store the media off-site

Advantage can be taken of the capability to make full or incremental backups of the database while ADSM is operational and available to clients. If a disaster occurs, the backed up copies can be used to restore the database. ADSM provides the capability to recover the database to its most current state (using roll-forward recovery) or to a specific point in time.

3. Back up the storage pools periodically on removable media and store the media off-site

   To make sure that a client's file system can always be restored, backup copies of the primary storage pools (backup, archive and space management) should be regularly made. The backup copies are stored in copy storage pools, which can be used to restore the original files in case they become damaged or lost.

   A typical storage hierarchy migrates from disk to tape, with the primary storage pools being on disk. These primary storage pools should be backed up incrementally to the same copy storage pools each day. Backing up to the same copy storage pool will ensure that files are not re-copied as they migrate to the next storage pool in the hierarchy.

   With correctly scheduled storage pool backups and migrations and with sufficient disk space, most copies can be made from the disk storage pool before the data is migrated to tape; this avoids unnecessary tape mounts.

   Backing up storage pools requires additional space requirements in the database. ADSM keeps track of the location, name and characteristics of all files using the server database. In addition, a small amount of space is needed for internal database indexing. As more files are added, copy storage pools and database storage requirements will need to be continuously evaluated.

4. ADSM also provides a stand-alone database salvage utility that can be used as a final attempt

   In the extremely unlikely event of a seriously corrupted database and when no database backups are available, the ADSM stand-alone database salvage utility can be used to attempt to recover. With this utility, a dump of the database can be made with DUMPDB, which retrieves as many logical database records as possible. Be aware that the dump utility does not access the recovery log, so all uncommitted actions will be lost.

   A new database and recovery log should now be allocated; do not destroy the old ones because the dump and load operation might have to be repeated.

   The database can now be reloaded with LOADDB.

   The database should now be audited with AUDITDB and any errors corrected.

The best protection against any kind of failure is to mirror the database and recovery log and to periodically back up the database and storage pools to off-site media.

Note

It is highly recommended to create Volume History and Device Configuration files.

Chapter 2. Planning Overview

This chapter looks at some of the elements that should be considered when planning for disaster recovery.

- What files should be backed up

  To be certain of a good backup strategy, the functions of and relationships between the essential ADSM files discussed in Chapter 1, “Disaster Recovery Overview” on page 3 should be fully understood.

  The following files should be mirrored and/or backed up:

  1. ADSM database
  2. Recovery log
  3. Storage pool volumes
  4. Device configuration file
  5. Volume history file
  6. Server option file

- Why should these files be backed up?

  These files should be backed up for the following reasons:

  1. The database contains all vital data about the ADSM environment.
  2. The recovery log assists in the recovery of the database to a certain point of time or to the most recent state after a failure.
  3. These copies can be used to restore the original files in case they become damaged.
  4. There should be at least one copy of the device configuration file. To restore a database, ADSM requires a definition for the device class from which the backup data is to be read. While the database is being restored, these definitions cannot be read from the database.
  5. The volume history file contains vital data required to restore the database to a specific point in time.
  6. The server option file contains the pointers to the copies of the device configuration and the volume history files.

- How often should a backup be taken and to what media?

  Backups should be regularly scheduled and should take the form described below:

  1. The database should have at least one mirrored copy; a full backup should be taken at least once a week and an incremental backup at least once a day.
  2. The recovery log should have at least one mirrored copy.
  3. The storage pool volumes should have at least one full weekly backup and a daily incremental backup.
  4. There should be at least one copy of the device configuration file. This file can also be printed out and the hard copy stored off-site.

5. There should be at least one copy of the volume history file, preferably on a separate disk to the one that the database is on. The volume history file can also be printed and the hard copy stored off-site.
6. There should be at least one copy of the server option file. This file can also be printed out and the hard copy stored off-site.

- Should full or incremental backups be taken?

  To determine whether a full or incremental backup should be scheduled depends on how available time is to be allocated. A full backup takes longer to run than a incremental; however, recovery time is faster with a full backup. Up to 32 incremental backups can be executed between every full backup. An incremental backup takes less time to run; however, incremental backups increase the time it takes to recover the database because a full backup must be loaded first, followed by all of the incremental backups.

- When should a full backup be taken?

  It will be necessary to take a full backup after any of the following events:

  – The database has never been backed up.
  – The maximum of 32 incremental backups is reached.
  – The database is extended or reduced.
  – The recovery log mode is changed from normal to roll-forward.

- When should an extra backup be performed?

  To ensure that the database contains only the most recent information, the database should be backed up if:

  – Significant client backup or archive activities have taken place.
  – Migration has moved data between storage pools.
  – Reclamation has moved client files on sequential storage volumes.
  – Move data or delete volume commands have been used to rearrange data or to remove volumes from a storage pool.
  – Storage pool backups have been done.

- How should the sizes of the database and recovery log be estimated?

  In order to calculate how much space to allocate for the database and recovery logs, consider the following:

  1. Database

     The server uses the internal database to keep track of all transactions. The size of the database is mainly dependent on the number of files to be backed up or archived because information about each file is stored into the database. To estimate the size of the database, the number of clients registered as well as how much data they will backup or archive needs to be considered.

     As a general guideline, the initial space allocated for the database is between 1 percent and 5 percent of the total amount of space allocated for the data storage.

     If the primary storage pools are backed up to copy storage pools, the database will require an additional 200 bytes for each file copied.

  2. Recovery log

The server uses the recovery log to keep track of all changes that are made to the database. The number of concurrent client sessions and the number of background processes determine the number of transactions and the size of the recovery log.

A useful guideline is to start with a recovery log of 12 MB, and then monitor the usage of the recovery log to determine whether it needs to be increased or decreased in size.

In ADSM V2, a new command, `RESET LOGConsumption`, and two new lines are added to the output of `Query LOG`. By running these commands every day during one week, the amount of space needed by the recovery log can be determined.

Running in roll-forward mode requires a lot more allocated space for the recovery log since all records are kept in the recovery log until the next database backup. It is recommended to set the database trigger to ensure that the recovery log does not run out of space. The database trigger function is discussed at the end of this section.

- Roll-forward versus point-in-time

  There are several points that need to be considered in order to decide whether to run in roll-forward or in point-in-time mode.

  Roll-forward mode provides the ability to recover the database to the most recent state upon a media failure of a database volume. Point-in-time mode (the default) requires much less recovery log space.

  For most situations, the point-in-time mode is sufficient if there is sufficient resource to mirror the database and recovery log. If the database is being mirrored, roll-forward mode only has value if all copies of the database become unavailable and all recovery log volumes are also available.

  If the database is so large that mirroring is not a viable option, roll-forward mode provides a very good alternative. It allows full recovery to the time of failure. However, some of the storage that was saved by not mirroring the database will be utilized instead by a much larger recovery log in roll-forward mode. To prevent the recovery log from running out of space, the database backup trigger can be used to run a database backup as soon as the recovery log reaches a certain percentage utilization.

Chapter 3. On-site Processes

The critical files for the ADSM server operation are the database, the recovery log and the storage pools. If part or all of the database become unusable for any reason, the entire ADSM server is unavailable. The same applies to the recovery log. If the database is lost and cannot be recovered, all of the backup, archive and space-management data for that server become unavailable. If a storage pool volume becomes unavailable and cannot be recovered, all of the data on that volume is lost.

In this chapter the important issues that should be considered while planning an on-site ADSM server recovery scenario are discussed.

3.1 Database and Recovery Log Protection and Recovery

Media failure of a database volume or a recovery log volume can be recovered in two ways:

- Using mirrored copies of the database or the recovery log
- Using backup copies of the database

The easiest and quickest, but also more expensive, way to protect and recover the ADSM database and recovery log is by mirroring. This way is expensive because it requires a lot more disk storage. Mirroring is not essential to recovery, but its implementation will save time and also allow end-users to continue to work without interruption during a recovery.

There are two types of database recovery:

1. Point-in-time recovery

   Point-in-time recovery restores a database to a point in time when a specific backup was created.

   To restore a database to a specific point in time, a full backup or a full backup and one or more incremental backups need to be restored.

2. Roll-forward recovery

   Roll-forward recovery restores a database to its most current state by using the following:

   - The last full backup of the database
   - Any incremental backups
   - The recovery log records to restore the changes made since the last backup was made

3.1.1 Recovery using Mirroring

Mirroring provides the following advantages:

- Protects against media failures within the database and recovery log.
- Allows ADSM operations to continue without interruption. The failing volume is automatically placed offline and synchronized when brought back online.
- Provides a way to avoid costly database recoveries.

- Increases the database read performance by allowing the server to read data from any available volume.

Of course, mirroring requires additional disk storage for the mirrored volumes.

3.1.1.1 What to Consider when Deciding to Mirror

Be sure that the mirrored copies are on different physical disks and even on different controllers if possible. By separating the volume copies on different physical devices, the server is more protected against media failures, and the availability of the database and recovery log are increased.

3.1.1.2 Setting Up Mirroring

In order to implement mirroring of the database and recovery log, use the following procedure:

1. Determine the current state and size of the database and the recovery log by issuing the following commands:
   - `Query DB`

     This will show allocation information for the database.
   - `Query DBVolume`

     This will show information about the database volumes and any database copies.
   - `Query LOG`

     This will show allocation information for the recovery log.
   - `Query LOGVolume`

     This will show information for the recovery log and any recovery log copies.
2. Allocate space to be used for the mirrored database and recovery log volumes:

   Use the `dsmfmt` command to format the new database and recovery log volumes.
3. Connect the database and recovery log volumes with their mirrored copies:

   Use `DEFINE DBCOPY` and `DEFINE LOGCOPY` to link the original volumes to the new copies.
4. View the status of the database and recovery log volumes again to ensure that all volumes have been mirrored.

   Use `Query DBVolume` and `Query LOGVolume` to check that the mirror copies are functioning correctly.

For a more detailed description of these commands and how to read the output, please refer to *The Administrator's Reference*. The graphical user interface can also be used to implement all of the steps apart from the format of the new volumes.

3.1.1.3 Recovery from a Failing Database Volume

Recovery from a media failure of a database volume can be effected by taking the following steps:

1. View the current status of the database volumes.

   Use Query DB, Query DBVolume or the Administrator GUI to determine the status, size, name and number of volumes.

2. Verify that the failing database volume is offline and delete it from ADSM.

   Use DELete DBVolume or the Administrator GUI.

3. Allocate space to be used for the new mirror of the database.

   Use dsmfmt to pre-allocate the file space. ;i1.dsmfmt

4. Connect the new mirrored copy to the database volume.

   Use DEFine DBCopy or the Administrator GUI to define a database volume copy.

   Bear in mind that when a failure occurs in one of the copies, the other copies are still considered as valid. Defining a new database copy means adding a new copy to the existing copies.

5. View the current state of the database volumes.

   Ensure that the new copy is added to the database volumes and that all copies are online. Use Query DBVolume or the Administrator GUI.

3.1.1.4 Recovery from a Failing Recovery Log Volume

Recover from a media failure of a recovery log volume can be effected by taking the following steps:

1. View the current status of the recovery log volumes.

   Use Query LOG, Query LOGVolume or the Administrator GUI to determine the status, size, name and number of volumes.

2. Verify that the failing recovery log volume is offline and delete it from ADSM.

   Use DELete LOGVolume or the Administrator GUI.

3. Allocate space to be used for the new mirror of the Recovery Log volume.

   Use dsmfmt to pre-allocate the file space.

4. Connect the new mirrored copy to the recovery log volume.

   Use DEFine LOGCopy or the Administrator GUI to define a Recovery Log volume copy.

   Bear in mind that when a failure occurs in one of the copies, the other copies are still considered as valid. Defining a new recovery log copy means adding a new copy to the existing copies.

5. View the current state of the recovery log volumes.

   Ensure that the new copy is added to the recovery log volumes and that all copies are online. Use Query LOGVolume or the Administrator GUI.

3.1.2 Recovery Using Backups

If mirroring has not been implemented then it will be necessary to take full and incremental backups of the database while the server is operational and available to clients. If a disaster occurs, the backup copies can be used to restore the database. Be aware that the loss of the recovery log will result in the need to restore the database. ADSM provides the capability to restore to either the most current state (roll-forward mode) or to a certain point in time (point-in-time mode).

To be able to restore the ADSM database copies of the volume history files and device configuration files will need to have been created.

The database can be restored to its most current state even if it has not been mirrored. In this case, backup copies of the database and recovery look will be needed. It is therefore advisable to always mirror the recovery log. Mirroring the recovery log requires only a small amount of additional space, but the benefit is high.

To recover the database using the `dsmserv restore db` command, backup versions that were created with either the `BACKUP DB` command or backup versions that were created automatically by ADSM based on settings specified using the `DEFINE DBBACKUPTRIGGER` command must exist.

3.1.2.1 Setting-Up for Recovery Using Backups

The following steps need to be taken to set up recovery using backups:

1. Define the device class for the backup.
2. Create the device configuration backup files.
3. Create the volume history backup files.
4. Perform the database backup.
5. Set the recovery log mode.
6. Set the database trigger.
7. Adjust, if necessary, the size of the recovery log.
8. Collect the output of `QUERY DBVolume` and `QUERY LOGVolume`, both with the option `Format=Detailed`.
9. Make a copy of the dsmserv.opt file.
10. Optional, but strongly advised, is the mirroring of the recovery log.

The type of backup (full or incremental), the way in which the backup is run (manual or DBBACKUPTRIGGER) and the recovery log mode (normal or roll-forward) will affect the level of recovery that is possible in each environment.

3.1.2.2 Recovery with Database Backups

There are two ways to recover from a database failure. The database can either be restored to a point in time or to its most current state.

To restore from a database or recovery log failure, the following information must be available:

- Backup copies of the database volumes
- The volume history file

- The device configuration file
- Output of QUERY DBVolume and QUERY LOGVolume, both with the option Format=Detailed
- The server options file

Restoring a Database Using Point-in-Time Recovery: To restore to a certain point in time, use the command:

```
> dsmserv restore db todate=mm/dd/yy
```

An optional parameter is totime=hh:mm:ss. ADSM will perform following tasks to restore the database:

- Read volume history file

  ADSM reads the volume history file to locate the last full backup that was made on or before the TODATE and the TOTIME parameters.

- Mount first backup tape

  Using the device configuration file, a mount of the first tape containing the start of the last full backup is requested.

- Restore the database

  The database is restored from the first volume and a request is issued to mount the next tape from the full or the following incremental backup.

From the old volume history file, a list of all the volumes that were deleted, reused or added since the original backup is required. With this list, the following steps need to be performed:

1. Audit all disk volumes, reused volumes and deleted volumes.

   This process identifies files recorded in the database that can no longer be found on the volume. If a copy of the file resides in a copy storage pool then the file on the audited volume is marked as damaged. Otherwise the file is deleted from the database and is lost. The RESTORE STGPOOL command can be used to restore those files that have been marked damaged.

2. Mark lost files as destroyed.
3. If any volumes are found that cannot be located, mark them as destroyed and recover them using copies from the copy storage pool volumes. If no backups are available, delete the volumes from the database by using the command:

   ```
   DELete Volume volname DISCARDdata=Yes
   ```

4. Re-add any volumes that were added since the backup.

Some files may be lost if they were moved since the backup as a result of migration, reclamation or move data requests and ADSM cannot locate them. This loss can be minimized by using the REUSEDELAY parameter when defining or updating storage pools. This parameter delays the return of volumes for scratch use or reuse.

If the storage pools have been backed up, loss of data can be minimized by restoring the storage pools after the reused volumes have been updated and the

deleted volumes marked as DESTROYED. To further minimize potential loss of data, the following points should be considered:

- Backup the database immediately after any storage pool backups.
- Turn off migration and reclamation while the database is being backed up.

  This is to prevent files being moved while the backup is in progress.
- Do not perform any move data operations during the backup of the database.
- Use the REUSEDELAY parameter to prevent copy storage pool volumes from being deleted or reused until specifically required.
- Use off-site copy storage pool volumes.

  Those volumes marked as OFFSITE are not reused or deleted until they are brought on-site. Ensure that these volumes are marked as off-site before a database backup is initiated.

Restoring a Database to its Most Current State: A database can be restored to its most current state if:

- There is at least one copy of the recovery log available and all recovery log volumes are intact
- The log mode was set to roll-forward continuously from the time the last full backup was made until the time the database got damaged
- A volume history file is available

For roll-forward recovery, ADSM uses the last full backup and all incremental backups, if any, and the recovery log records that reflect changes made after the last backup was made.

To restore the database to its most current state, use the command `dsmserv restore db`. This command restores the database as follows:

- Read the recovery log

  The recovery log is read to determine which volumes contain the most recent database backup.
- Request volume mounts

  Volume mounts are requested to load the most recent backup series (the last full backup and any incremental backups).
- Update database
- The recovery log is used to update the database to its most current state.

3.1.3 Salvage Utilities

In the unlikely event of a seriously corrupted database and having no backups available, the ADSM stand-alone database salvage utilities can be used to attempt to recover the database. These utilities require the following steps:

1. Halt the server.
2. Ensure that the device configuration file is available.
3. Execute `dsmserv dumpdb`.

   This command will retrieve as many logical database records as possible and write them into the database dump.

4. Execute `dsmserv install`.

   This command will initialize a new database and recovery log. New space should be allocated for the data base and recovery log rather than using the old space, just in case the dump and load process needs to be repeated.

5. Execute `dsmserv loaddb`.

   This command will load the dumped database into the newly installed server.

6. Execute `dsmserv auditdb`.

   This command will ensure that the database is returned to a consistent state. The command should be run with Fix=Yes to recover the database. The recovery log mode must also be set to normal otherwise the server may run out of recovery log space.

7. Restart the server.

8. Execute `AUDit Volume`.

   This command will synchronize the database with the storage pool volumes.

   Note

   While an AUDIT process is active, clients cannot restore files from the volume being audited.

After these steps have been completed and any errors corrected, the database should be in an identical state to that just before the failure. The only transactions that might have been lost are those transactions that were not yet committed at the moment of failure.

3.2 Storage Pool Protection and Recovery

To ensure full protection of clients files that are backed up, archived or migrated, the following steps, at a minimum, should be followed:

- Perform a full database backup once a week and an incremental database backup every day.
- Perform a backup of all storage pools every night.
- Keep at least one copy of all backups on tape off-site.

Storage pool volumes contain all backed up, archived and space management files.

In ADSM V2, there are two types of storage pools:

1. Primary storage pools

   These contain data that is backed up, archived or migrated.

2. Copy storage pools

   These are backup copies of primary storage pools.

The copy storage pool provides a way of recovering from a disaster, data-integrity or a media failure. Copy storage pools cannot be used as:

1. The next storage pool in a migration hierarchy
2. A management class destination for space-management files

3. A copy group destination to backup or archive files

3.2.1 Setting Up a Copy Storage Pool

To set up a copy storage pool, perform the following steps:

1. Define a copy storage pool by issuing the command:

```
DEFine STGpool <poolname> <devclassname> POoltype=COpy
```

2. Define the same number of volumes in the copy storage pool as already defined in the primary storage pool by issuing the command:

```
DEFine Volume <poolname> <volname> (ACCess=OFfsite)
```

   The parameter ACCess=OFfsite is optional, but must be set if the volumes are to be kept off-site.

3.2.2 Starting the Copy Procedure

To start the copy procedure use the following steps:

1. To copy the primary storage pool to the copy storage pool, issue the command:

```
BAckup STGpool <primarypool> <copypool>
```

2. Set up a schedule to start the copy operation automatically if required. This schedule can be set up by issuing the command:

```
DEFine SCHedule <domainname> <schedulename> Type=Administrative
                CMD=''BAckup STGpool <primarypool> <copypool>''
                ACTIVE=Yes STARTTime=20:00 PERiod=1
```

3.2.3 Recovery from a Copy Storage Pool

By using backup copies of primary storage pools recovery from disaster or media failure can be easily effected.

3.2.3.1 Restoring after a Media Failure

Use the following steps to recover from a media failure:

- Change the access mode of the damaged volume to UNAVAILABLE by issuing the command:

```
UPDate Volume <volumename> ACCess=UNAVailable
```

- Determine which copy pool volumes are needed to restore the primary storage pool volume by issuing the command:

```
RESTORE Volume <volumename> Preview=Yes
```

- Set the access mode of those copy pool volumes to UNAVAILABLE.

  This prevents the volumes from being used before the damaged volume is restored. Changing the access mode is done with the command:

```
UPDate Volume <volumename> ACCess=UNAVailable
```

- Restore the destroyed volume by issuing the command:

```
RESTORE Volume <volumename>
```

- Set the access mode of the copy pool volumes used back to OFFSITE with the command:

```
UPDate Volume <volumename> ACCess=OFfsite
```

3.2.3.2 Restoring after a Disaster

Use the following steps to recover from a disaster and to restore all storage pools and the database:

- Restore the volume history, the device configuration and the server options files.
- Restore the database from the latest backup by issuing the command:

```
> dsmserv restore db
```

- Change the access mode of all primary storage pool volumes to DESTROYED by issuing the command:

```
UPDate Volume * ACCess=DEStroyed
```

- Identify any volumes in the copy storage pool that were lost due to the disaster and delete those volumes by issuing the command:

```
DELete Volume <volumename> DISCARDdata=Yes
```

- Change the access mode of the remaining copy storage pool volumes to READWRITE by issuing the command:

```
UPDate Volume * ACCess=READWrite WHERESTGpool=<poolname>
```

- Define new primary storage pool volumes so the files in the damaged volumes can be restored to these new volumes.

  Do this by issuing the command:

```
DEFine Volume <poolname> <volumename>
```

  If only scratch volumes are being used, there is no need to perform this step.

- Restore the files from the copy storage pool volumes into the newly created primary storage pool volumes by issuing the command:

```
RESTORE STGpool <primarypool>
```

- To insure against another loss of data, immediately backup all storage pool volumes and the database.
- Normal activity can now be resumed.

3.3 Rebuilding the Server

If the entire server is destroyed, the operating system should be restored along with the ADSM server code. After a successful install, proceed as described in 3.2.3.2, “Restoring after a Disaster” on page 19.

Chapter 4. Off-site Processes

This chapter looks at the important issues that should be considered in planning for off-site server recovery.

4.1 Database and Storage Pool Protection and Recovery

This chapter provides a guide to backup procedures and scenarios for the ADSM server. In order to recover after a disaster, a good backup plan and procedures are necessary.

A complete loss of the entire site were the ADSM server was located will be assumed. It will be demonstrated that with a good disaster recovery plan, the ADSM server can easily be restored on a backup system or even in another location.

4.1.1 Database and Storage Pool Protection

To be able to recover from any kind of media failure or any disaster, there should be one copy of the database and the storage pools on-site and another copy off-site.

Two more important points are:

1. Database and storage pool backups can be initiated at any time. The volumes created during backup should be stored in an off-site location together with all other backups necessary for a disaster recovery. How often the backups are taken and the data moved off-site determines to which point in time recovery can be effected. If the process is implemented every weekend, then any recovery necessary will only restore to the previous weekend. If the process is implemented daily, then recovery can be effected as far as the previous day.
2. Once an ADSM disaster recovery plan has been developed, the procedures should be regularly tested. At a minimum, this should occur once a year. During the testing, a great deal of attention should be paid to the documentation to ensure that the procedures could be correctly followed by someone not involved with the plan development.

4.1.2 Database Backup and Copy Storage Pool Management

How important it is to have a good backup plan has already been discussed in Chapter 3, “On-site Processes” on page 11.

Remember

The better the plan, the easier it will be to recover.

In order to have the capability to recover from any kind of disaster, the following steps should be taken:

- Make a mksysb tape that reflects the most current AIX system configuration.
- Perform a full database backup once a week and an incremental backup every day.

- Backup the primary storage pools every night to the copy storage pools.

  Ensure that a database backup is made right after the primary storage pool backups.

- Make a backup copy of the volume history file, the device configuration file and the server options file.

  This should be performed once a night after the primary storage pool and database backups are done.

- Mark all volumes used as access mode OFFSITE and move them to an off-site location.

  This will include the backups of the database, the volume history file, the device configuration file and the server options file.

Recovery from a disaster should be possible if this process is implemented; there are also a few additional maintenance steps that need to be performed:

- Identify once a day all off-site volumes that are ready to be returned to the on-site location.

  These volumes can be returned to the on-site location because they are no longer required. The volumes have been stored for the delay time specified with the REUSEDELAY parameter, and can now be reclaimed.

- Return these volumes to the on-site location.

  These volumes will now be automatically returned to scratch.

4.2 Off-site Database Copies

A copy of the database should be stored off-site to guard against complete disaster having eradicated any local copies.

4.2.1 Creating Off-site Database Copies

The off-site database backup can be made in two ways, each of which has advantages and disadvantages.

4.2.1.1 The DBBACKUPTRIGGER Copy

The DBBACKUPTRIGGER was designed to be used to protect against the situation of the recovery log running out of space, but it can also be used to implement an automatic full backup of the database. If this method is only being used for emergency purposes, then regular scheduled backups must be still be made of the database.

The disadvantages of this method are:

- The DBBACKUPTRIGGER is started whenever the criteria it was set with occur; as a result, there might be no operator available to do any mounts that are necessary.
- A backup for the on-site copy will need to be scheduled as soon as possible after the off-site copy has been made. Other products, such as NetView or JobScheduler, will be needed to catch the messages from the DBBACKUPTRIGGER off-site copy operation.
- If the recovery log fills faster than had been calculated and the DBBACKUPTRIGGER copy starts a day earlier, the off-site recovery procedure will be out off sync and might fail.

4.2.1.2 The SCHEDULED Database Copy

It is possible to only use this method of creating an off-site copy of the database; if this is done, then it will be necessary to be careful that the recovery log does not fill up.

The advantages for this method are:

- The time of backup is known, so an operator can be made available to do any necessary mounts.
- Full control of the timing of backups is possible; there are no dependencies on the filling of the recovery log.

The disadvantages for this method are:

- It will still be necessary to use DBBACKUPTRIGGER to automatically catch the situation where the recovery log runs out of space.
- Since it is not known when the database off-site copy terminates it is not possible to know when to start the on-site copy. A product such as JobScheduler may be used to assist with this.

4.2.1.3 AIX Copy Commands

Another, inadvisable method is to use AIX copy commands, such as `dd` or `tcopy`, to make a copy of the on-site database backup copy and take it off-site. This method is not advised because ADSM will not know about this copy of the original on-site backup copy.

4.3 Off-site Storage Pool Copies

As ADSM V2 supports multiple copy storage pools for one primary storage pool, it is very easy to create two identical copies and then separate these copies physically. Just define two different copy storage pools for every primary storage pool and initiate a backup command for the primary storage pool to each copy storage pool.

The on-site copy storage pool is used for media failures and the off-site copy storage pool is used for major disasters.

4.3.1 Creating Off-site Storage Pool Copies

The following steps will illustrate the creation of the copy storage pools and the commands necessary to create an off-site copy:

- As a first step, the administrator will have to define a copy storage pool, called DIS_RECOVER in this example

  This is done by the command:

```
DEFine STGpool DIS_RECOVER <devclassname> POoltype=COpy MAXSCRatch=100
```

- It is then necessary to initiate the backup of the primary storage pool into the copy storage pool with the following command, which should be executed for each primary storage pool:

```
BAckup STGpool <primarypool> DIS_RECOVER.
```

- After the primary storage pool has been backed up to the copy storage pool, the administrator must check which volumes in the DIS_RECOVER pool have an access mode of READWRITE or READONLY and which are at least partially filled.

  The command to do this is:

```
Query Volume ACCess=READWrite,READOnly STGPOOL=DIS_RECOVER
             STatus=FULl,FILling Format=Detailed
```

- The access mode of these volumes should be changed to OFFSITE.

  This is done with the command:

```
UPDate Volume * ACCess=OFfsite LOcation="locationname"
              WHERESTGpool=DIS_RECOVER WHERESTatus=FULl,FILling
              WHEREACCess=READWrite,READOnly
```

  By specifying the location name in the update volume command, it will be possible to later query all tapes for a certain off-site location.

- To check if any off-site volumes have become empty as a result of tape reclamation, the administrator can issue the command:

```
Query Volume * ACCess=OFfsite STGpool=DIS_RECOVER STatus=EMPty
```

  These tapes can be returned to the on-site location and added to the scratch pool by changing the access mode from off-site to read/write.

  This can be done with the following command:

```
UPDate Volume * ACCess=READWrite WHERESTGpool=DIS_RECOVER
              WHERESTatus=EMPty WHEREACCess=OFfsite
```

4.4 Recovering from a Disaster

Follow the steps described below to recover from a disaster and to restore the entire storage pool and database:

- If the entire server is destroyed, the operating system and ADSM server code should first be restored. This is usually done using a mksysb tape made previously.
- Restore the volume history, the device configuration and the server options files.
- Restore the database from the latest backup by issuing the command:

```
> dsmserv restore db
```

- Change the access mode of all primary storage pool volumes to destroyed by issuing the command:

```
UPDate Volume * ACCess=DEStroyed.
```

- Identify any volumes in the copy storage pool that were lost due to the disaster, and delete these volumes by issuing the command:

```
DELete Volume <volumename> DISCARDdata=Yes
```

- Change the access mode of the remaining copy storage pool volumes to read/write by issuing the command:

```
UPDate Volume * ACCess=READWrite WHERESTGpool=<poolname>
```

- Define new primary storage pool volumes so the files in the damaged volumes can be restored to these new volumes.

  Do this by issuing the command:

```
DEFine Volume <poolname> <volumename>
```

  If scratch volumes only are being used, there is no need to perform this step.

- Restore the files from the copy storage pool volumes into the newly created primary storage pool volumes by issuing the command:

```
RESTORE STGpool <primarypool>
```

- To insure against loss of data, immediately backup all storage pool volumes and the database.
- Normal activity can now be resumed.

Chapter 5. Disaster Recovery Manager

An optional feature called Disaster Recovery Manager (DRM), will become available in a future release of ADSM. This feature will assist in the preparation of a disaster recovery plan. DRM uses off-site data to recover from a disaster; it is even possible for people who are not familiar with ADSM or DRM to restore machines, replacement machines or even machines at another location.

DRM performs three main tasks:

- Automatically creates a recovery plan
- Keeps track of the on-site and off-site media
- Stores client recovery and machine information in the ADSM database

Important

It is vital that the disaster recovery process be fully understood, both theory and practice. A program should never be relied on unless its function is also fully understood. Thus, before employing a tool such as DRM, ensure that these principles are understood and the functions and purposes of all the components known.

5.1 Purpose of DRM

The purpose of DRM can be summarized as follows:

1. Generate a disaster recovery plan
2. Track disaster recovery media movement off-site
3. Restore server using recovery plan file
4. Restore client files from:
   - Copy storage pools
   - Restored primary pools

5.2 Setting Up DRM

The setup of DRM is accomplished using the following series of commands:

- Specify the copy storage pools to be used by DRM.

  Execute the following command:

```
> set drmcopystgpool <copystgpoolname> .. <copystgpoolname>
```

- Specify the primary storage pools to be used by DRM.

  Execute the following command:

```
> set drmprimstgpool <primarystgpoolname> .. <primarystgpoolname>
```

- Specify the pathname and prefix of the recovery instructions source file.

  Execute the following command:

```
> set drminstprefix <prefixname>
```

- Specify the character to be added at the end of the replacement volume names in the disaster recovery plan.

  Execute the following command:

```
> set drmplanvpostfix <character>
```

- Specify the path name and the prefix for the generated recovery plan.

  Execute the following command:

```
> set drmplanprefix <planprefixname>
```

- Specify that the MOVE DRMEDIA and QUERY DRMEDIA commands should process the backup volumes

  Execute the following commands:

```
> set drmfileprocess yes
```

There are a few other settings that can be implemented; refer to the *ADSM Administrators Guide, SH35-0134*, for more information.

5.3 DRM Test Scenario Overview

A DRM test scenario could be implemented as follows:

1. Testcase setup
   - Make the necessary directories.
   - Set the necessary variables and export them.
   - Allocate the space for database and recovery log.
   - Set the necessary variables and export them.
   - Register the correct licenses for ADSM.
   - Create a new administrator.
   - Configure the server.
   - Perform some housekeeping to prepare for this test case.
2. Main setup
   - Create the logical volumes required.
   - Set up all the primary and copy storage pools.
   - Implement necessary the DRM settings.
   - Prioritize ADSM client machines.
   - Define site-specific ADSM server recovery instructions.
3. Day-to-day operations
   - Day 1
     - Back up client files.
     - Back up ADSM server storage pools.
     - Back up ADSM server database (full).
     - Eject the backup volumes from the library.
     - Hand the volumes to the courier.
     - Generate the recovery plan.

- Day 2
  - Back up client files.
  - Back up ADSM server storage pools.
  - Back up ADSM server database (full).
  - Transfer new backup volumes off-site.
  - Acknowledge receipt of previously transferred volumes at vault.
  - Generate the recovery plan.
- Day 3
  - Erase copy storage pool volume (simulate reclamation).
  - Back up ADSM server database (incremental).
  - Transfer new backup volumes off-site.
  - Acknowledge receipt of previously transferred volumes at vault.
  - Note that volumes need to be retrieved from the vault.
  - Generate the recovery plan.

4. Simulate disaster
   - Destroy ADSM client machine by removing all files.
   - Destroy ADSM server machine by removing all files and logical volumes.
5. Recover using the created macros
   - Restore ADSM server using the latest recovery plan.
   - Restore client machine files from ADSM server copy storage pools.
   - Restore ADSM server primary storage pools.
   - Move database backup and copy storage pool volumes back to the vault.
6. Day-to-day operations

   Restore client machine files from ADSM server primary storage pools.

This concludes a typical example of DRM operations.

5.4 DRM Test Scenario

The following scripts can be used to effect the typical test scenario defined in the previous section.

5.4.1 Testcase Setup

This script performs the testcase setup for the DRM test case.

```
#-------------------------------------------------------------------------
# begin drm prepare testcase common setup         defstart.src
#-------------------------------------------------------------------------
banner "initialize"
COMPID="prepare"

mkdir /home/guest/drmtest
mkdir /home/guest/drmtest/$COMPID
mkdir /home/guest/drmtest/$COMPID/tservr
mkdir /home/guest/drmtest/$COMPID/plandir
mkdir /home/guest/drmtest/$COMPID/scratchdir

if [ 0$TCPPORT = 0 ]
```

```
  then TCPPORT=1509
fi
if [ O$TCNAME = O ]
  then TCNAME="TCNAME"
fi

# You can specify an alternate volume group for mklv in file TESTVG.
cat TESTVG | read TESTVG
if [ $TESTVG ]
  then echo "Test logical volumes may be placed in volume group "$TESTVG
else TESTVG="rootvg"
  echo "Test logical volumes may be placed in volume group "$TESTVG
fi
if [ O$SVRNAME = O ]
  then SVRNAME="DSM"$TCPPORT
fi

SERVERDIR=/home/guest/drmtest/$COMPID/tservr/$SVRNAME
rm -r $SERVERDIR
mkdir $SERVERDIR

# You can specify an alternate server code location in file SERVERCODE.
cat SERVERCODE | read SERVERCODE
if [ $SERVERCODE ]
  then echo "Server code will be obtained from directory "$SERVERCODE
else SERVERCODE="/usr/lpp/adsmserv/bin"
  echo "Server code will be obtained from directory "$SERVERCODE
fi

cp $SERVERCODE/dsmserv $SERVERDIR/dsmserv
chmod a+w $SERVERDIR/dsmserv
cp $SERVERCODE/dsmameng.hlp $SERVERDIR/dsmameng.hlp
cp $SERVERCODE/dsmameng.txt $SERVERDIR/dsmameng.txt
cp $SERVERCODE/pkmonx $SERVERDIR/pkmonx
cp $SERVERCODE/dsmreg.lic $SERVERDIR/dsmreg.lic
cp $SERVERCODE/planexpl.awk.smp $SERVERDIR/planexpl.awk.smp
cp $SERVERCODE/machchar.awk.smp $SERVERDIR/machchar.awk.smp

DRMPLANDIR=/home/guest/drmtest/$COMPID/plandir/$SVRNAME
#DRMPLANDIR=/prepare
rm -r $DRMPLANDIR
mkdir $DRMPLANDIR

SCRATCHDIR=/home/guest/drmtest/$COMPID/scratchdir/$SVRNAME
rm -r $SCRATCHDIR
mkdir $SCRATCHDIR

TRACEFILE=/home/guest/drmtest/$COMPID/$TCNAME.log
CMDOUT=$SCRATCHDIR/cmdout.txt
DSMOUT=$SCRATCHDIR/dsmout.txt
rm $TRACEFILE

xl=/usr/bin/xl$SVRNAME
cat <<XXX >$xl
echo \$@ | tee -a $TRACEFILE
# echo \$@ | tee -a $TRACEFILE
\$@ 1>$CMDOUT 2>&1
# \$@ 1>$CMDOUT 2>&1
rc=\$?
```

```
cat $CMDOUT | tee -a $TRACEFILE
date | read thetime
echo "rc=" \$rc "                          " \$thetime | tee -a $TRACEFILE
exit \$rc
XXX
chmod a+x $xl

xp=/usr/bin/xp$SVRNAME
cat <<XXX >$xp
print "\$@" 2>&1 | tee -a $TRACEFILE
# print \$@ 2>&1 | tee -a $TRACEFILE
XXX
chmod a+x $xp

pause=/usr/bin/pause$SVRNAME
cat <<XXX >$pause
echo "hit enter to continue..."
read JUNK
XXX
chmod a+x $pause

$xl date
$xl ls -l $SERVERCODE
THISHOST=$(hostname)
$xl echo $THISHOST
STARTFILE=dsmtest.ksh
MACFILE=newserver.mac
CLIENTDIR=/usr/lpp/adsm/bin
PW=YAHOO
DRMDIR=/usr/lpp/drm/bin

echo "------------------ begin testcase source --------------------------"
             >> $TRACEFILE
cat $TCNAME.src >> $TRACEFILE
echo "------------------ end   testcase source --------------------------"
             >> $TRACEFILE

$xp "AIX fix level is:"
$xl lslpp -m bos.obj

cd $SERVERDIR
$xl loadpkx -f pkmonx

$xl echo remove old files
$xl rm $SERVERDIR/db01x
$xl rm $SERVERDIR/lg01x
$xl rm $SERVERDIR/bk01x
$xl rmlv -f $SVRNAME"db01x"
$xl rmlv -f $SVRNAME"lg01x"

$xl echo create new ones...
$xl dsmfmt -m -db $SERVERDIR/db01x 9
$xl dsmfmt -m -log $SERVERDIR/lg01x 8
$xl dsmfmt -m -data $SERVERDIR/bk01x 5
$xl mklv -y $SVRNAME"db01x" $TESTVG 4
$xl mklv -y $SVRNAME"lg01x" $TESTVG 4
```

```
rm $SCRATCHDIR/$TCNAME.stat
date | read THETIME
echo $TCNAME testcase verification $THETIME >> \
   $SCRATCHDIR/$TCNAME.stat

xver=/usr/bin/xver$SVRNAME
cat <<XXX >$xver
# args are:   verifyid receivedrc comparison expectedrc
eval "if [ \$2 \$3 \$4 ]
then
  echo verify \$1 true   \$2 \$3 \$4 >> $SCRATCHDIR/$TCNAME.stat
  echo verify \$1 true   \$2 \$3 \$4 | tee -a $TRACEFILE
else
  echo verify \$1 failed \$2 \$3 \$4 >> $SCRATCHDIR/$TCNAME.stat
  echo verify \$1 failed \$2 \$3 \$4 | tee -a $TRACEFILE
fi"
XXX
chmod a+x $xver

  cat <<XXX >$SERVERDIR/dsmserv.optx
* Server options file located in $SERVERDIR/dsmserv.optx
TCPPort $TCPPORT
VOLUMEHISTORY $SERVERDIR/volhistory.txtx
DEVCONFIG     $SERVERDIR/devconfig.txtx
XXX

cat <<XXX >$SERVERDIR/$STARTFILE
#!/bin/ksh
export DSMSERV_CONFIG=$SERVERDIR/dsmserv.optx
export DSMSERV_DIR=$SERVERDIR
export DSM_CONFIG=$SERVERDIR/dsm.optx
export DSM_LOG=$SERVERDIR
export DRM_CONFIG=$SERVERDIR/drm.optx

cd $SERVERDIR
echo Please start new ADSM server console with command \
   dsmadmc -CONSOLE
nohup $SERVERDIR/dsmserv quiet &
XXX
chmod a+x $SERVERDIR/$STARTFILE

export DSMSERV_CONFIG=$SERVERDIR/dsmserv.optx
export DSMSERV_DIR=$SERVERDIR
export DSM_CONFIG=$SERVERDIR/dsm.optx
export DSM_LOG=$SERVERDIR
export DRM_CONFIG=$SERVERDIR/drm.optx

$xl banner "install server"

cd $SERVERDIR
$xl dsmserv -F install  2 /dev/r$SVRNAME"lg01x" $SERVERDIR/lg01x \
   2 /dev/r$SVRNAME"db01x" $SERVERDIR/db01x

cat <<XXX >$SERVERDIR/$MACFILE
update copygroup standard standard standard standard type=backup vere=nolimit
activate policyset standard standard
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
```

```
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
register license xxxxx SORRY NO REAL KEYS IN THIS SAMPLE xxxxx
register node $THISHOST $PW
register admin root $PW
grant authority root cl=system
def vol BACKUPPOOL $SERVERDIR/bk01x acc=READW
set drmfileprocess yes
set passexp 9999
halt
XXX

$xl banner "config server"
cd $SERVERDIR
(sleep 20; cat -u $SERVERDIR/$MACFILE ) | $SERVERDIR/dsmserv \
   1>> $TRACEFILE 2>&1

echo use $SERVERDIR/$STARTFILE to start the server...

grep -q "SERVERNAME.*$SVRNAME" $CLIENTDIR/dsm.sys
if [ $? != 0 ]
  then
  echo "SERVERNAME        $SVRNAME " >> $CLIENTDIR/dsm.sys
  echo " COMMMETHOD        TCPIP      " >> $CLIENTDIR/dsm.sys
  echo " TCPSERVERADDRESS  $THISHOST  " >> \
    $CLIENTDIR/dsm.sys
  echo " TCPPORT           $TCPPORT   " >> $CLIENTDIR/dsm.sys
fi

cat <<XXX >$SERVERDIR/dsm.optx
SERVERNAME         $SVRNAME
TAPEPROMPT         NO
XXX

$xl echo "create some handy utility scripts..."

xdsmenv=/usr/bin/xdsmenv$SVRNAME
cat <<XXX >$xdsmenv
# handy set env. run with . xdsmenv
export DSMSERV_CONFIG=$SERVERDIR/dsmserv.optx
export DSMSERV_DIR=$SERVERDIR
export DSM_CONFIG=$SERVERDIR/dsm.optx
export DSM_LOG=$SERVERDIR
export DRM_CONFIG=$SERVERDIR/drm.optx
XXX
chmod a+x $xdsmenv

cat <<XXX >$DRMPLANDIR/localmods.awk
# Make local modifications to the plan explode awk procedure.
# User would normally manually edit a copy of the planexpl.awk.smp file
# instead of using an awk script like this.
           {print \$0}
 /aline =/ {
  print "sub(\"read pause\",\"sleep 60\",aline)\;";
  }
XXX
```

```
xdsmadmc=/usr/bin/xdsmadmc$SVRNAME
cat <<XXX >$xdsmadmc
echo \$@ | tee -a $TRACEFILE
dsmadmc -server=$SVRNAME -id=ROOT -pass=$PW \
  \$@ \
  2>&1 | tee $DSMOUT
cat $DSMOUT >> $TRACEFILE
XXX
chmod a+x $xdsmadmc
xdsmadmc="ksh -f "$xdsmadmc

xdsmc=/usr/bin/xdsmc$SVRNAME
cat <<XXX >$xdsmc
echo \$@ | tee -a $TRACEFILE
# echo \$@ | tee -a $TRACEFILE
keywd=\$1
if [ \$# -ne 0 ]
  then shift
fi
dsmc \$keywd -server=$SVRNAME -pass=$PW \
  \$@ \
  2>&1 | tee $DSMOUT
cat $DSMOUT >> $TRACEFILE
XXX
chmod a+x $xdsmc
xdsmc="ksh -f "$xdsmc

$xl banner "start server"

echo "sleep 15 seconds.... let stuff settle down"
$xl sleep 15
rm $SERVERDIR/nohup.out
# $xl $SERVERDIR/$STARTFILE
$SERVERDIR/$STARTFILE
echo "sleep 30 seconds.... give server chance to startup"
$xl sleep 30
#-------------------------------------------------------------------------
# end drm prepare testcase common setup
#-------------------------------------------------------------------------

$xp remove old files
$xl rmlv -f $SVRNAME"bk02"
$xl rmlv -f $SVRNAME"bk02@"
$xl rmlv -f $SVRNAME"bk02@@"
$xl rmlv -f $SVRNAME"bk02@@@"
```

5.4.2 Setup

This script will perform the setup required for the DRM test scenario.

```
$xl banner "begin testcase"

$xl mklv -y $SVRNAME"bk02"  $TESTVG 4
$xl dsmfmt -m -data $SERVERDIR/ar01 5

$xdsmadmc DEFINE DEVCLASS FILES   DEVTYPE=FILE \
   MAXCAPACITY=4096K MOUNTLIMIT=2 \
   DIRECTORY=$SERVERDIR
$xdsmadmc DEFINE DEVCLASS FILESSM DEVTYPE=FILE \
   MAXCAPACITY=100K MOUNTLIMIT=2 \
```

```
   DIRECTORY=$SERVERDIR
$xdsmadmc DEFINE DEVCLASS COOL8MM DEVTYPE=8MM \
   FORMAT=DRIVE MOUNTLIMIT=1 MOUNTWAIT=60 \
   MOUNTRETENTION=60 PREFIX=ADSM LIBRARY=ITSIBITSI
$xdsmadmc DEFINE DEVCLASS LIB8MM  DEVTYPE=8MM \
   FORMAT=DRIVE MOUNTLIMIT=1 MOUNTWAIT=60 \
   MOUNTRETENTION=60 PREFIX=ADSM LIBRARY=REALTHING
$xdsmadmc DEFINE LIBRARY  ITSIBITSI LIBTYPE=MANUAL
$xdsmadmc DEFINE DRIVE    ITSIBITSI ONLYONE DEVICE=/dev/mt0
$xdsmadmc define stgpool CSTORAGEPF  FILES   pooltype=copy
$xdsmadmc define stgpool CSTORAGEPFA FILESSM pooltype=copy
$xdsmadmc define stgpool CSTORAGEP   COOL8MM pooltype=copy
$xdsmadmc define stgpool BACKUPPOOLT COOL8MM
$xdsmadmc define stgpool BACKUPPOOLF FILES
$xdsmadmc define stgpool CSTORAGEPSM FILESSM \
   pooltype=copy reusedelay=9999
$xdsmadmc def vol BACKUPPOOLT BACK4X acc=READW
$xdsmadmc def vol BACKUPPOOLF $SERVERDIR/bk03 acc=READW
$xdsmadmc def vol BACKUPPOOL /dev/r$SVRNAME"bk02" \
   access=READW
$xdsmadmc update stgpool BACKUPPOOL MAXSIZE=10K \
   NEXT=BACKUPPOOLF
$xdsmadmc define vol cstoragepf $SERVERDIR/bk05
$xdsmadmc define vol cstoragepfa $SERVERDIR/bk99
$xdsmadmc define vol ARCHIVEPOOL $SERVERDIR/ar01 access=READW

$xl banner 'setup'

$xdsmadmc set drmfileprocess yes
$xdsmadmc set drmcourier fedex
$xdsmadmc set drmvault ironmnt

$xl banner "save descriptions of priority ADSM client machines"
# Note: to make testcase execution simple, our test ADSM 'client' machine is
# the same machine that the ADSM server runs on.

$xdsmadmc 'define machine acctsrecvsrv desc="accounts receivable server" \
   bu=21 fl=2 ro=2749 pri=1'
$xdsmadmc define machnodeassoc acctsrecvsrv $THISHOST
$xdsmadmc 'define recoverym filesrvimage vol="mkssy1" \
   desc="mksysb image of payroll server machine OS" \
   loc=ironmnt acc=offs'
$xdsmadmc 'update recoverym filesrvimage type=boot \
   prod=mksysb producti=help!
$xdsmadmc define recmedmachass filesrvimage acctsrecvsrv

$xl cp $SERVERDIR/machchar.awk.smp $DRMPLANDIR/machchar.awk
echo "devices"                          > $SCRATCHDIR/clientinfo.txt
lsdev -C | sort -d -f                 >> $SCRATCHDIR/clientinfo.txt
echo "logical volumes by volume group" >> $SCRATCHDIR/clientinfo.txt
lsvg -o | lsvg -i -l                  >> $SCRATCHDIR/clientinfo.txt
echo "file systems"                     >> $SCRATCHDIR/clientinfo.txt
df                                      >> $SCRATCHDIR/clientinfo.txt
awk -f $DRMPLANDIR/machchar.awk -v machine=acctsrecvsrv $SCRATCHDIR/
   clientinfo.txt > $SCRATCHDIR/clientinfo.mac
$xl cat $SCRATCHDIR/clientinfo.mac
$xdsmadmc macro $SCRATCHDIR/clientinfo.mac
```

```
$xdsmadmc query machine acctsrecvsrv bu=21 f=d
$xl grep -i "Node.Name:."$THISHOST $DSMOUT
$xver v1 $? -eq 0

$xl banner "define site specific adsm server recovery instructions"
   cat <<XXX \ >$SERVERDIR/RECOVERY.INSTRUCTIONS.GENERAL
 This ADSM server contains the backup and archive data for XYZ corporation
 accounts receivable system. It also is used by various end users in the
 finance and materials distribution organizations.
 The storage administrator in charge of this server is Jane Doe 444-561-3406.
 If a disaster is declared, here is the outline of steps that must be done.
 1. Determine the recovery site. Our alternate recovery site vendor is IBM
    BRS in Tampa, Fl, USA 213-884-5647.
 2. Get the list of required recovery volumes from this recovery plan file
    and contact our offsite vault so that they can start pulling the
    volumes for transfer to the recovery site.
 3. etc...
XXX
   cat <<XXX \ >$SERVERDIR/RECOVERY.INSTRUCTIONS.OFFSITE
 Our offsite vaulting vendor is ironmnt.
 Their telephone number is 514-555-2341. Our account rep is Joe Smith.
 Our account number is 1239992. Their address is ...
 Here is a map to their wharehouse ...
 Our courier is ...
XXX
   cat <<XXX \ >$SERVERDIR/RECOVERY.INSTRUCTIONS.INSTALL
 The base ADSM server system is AIX 4.1.2 running on an RS6K model 320.
 Use mksysb volume serial number svrbas to restore this system image.
 A copy of this mksysb tape is stored at the vault. There is also a copy
 in bldg 24 room 4 cabinet a. The image includes the ADSM server code.
 The system programmer responsible for this image is Fred Myers.
 Following are the instructions to do a mksysb based OS install:
                   .
                   .
                   .
XXX
```

5.4.3 Day-to-Day Operations

This script will implement the day-to-day operations described in the overview.

```
$xl banner 'day to day operations'

$xl banner 'day 1'

$xl date 9506011200
$xdsmadmc set drmdbbackupexp 60

$xl banner "backup client files"

$xl rm $SCRATCHDIR/*20kbytesfile*
$xl cp $SERVERDIR/dsmserv.optx $SCRATCHDIR/lt20kbytesfile
$xl cp $SERVERCODE/pkmonx $SCRATCHDIR/gt20kbytesfile
$xl cp $SERVERDIR/dsmserv.optx $SCRATCHDIR/lt20kbytesfile.arc

$xdsmc sel $SCRATCHDIR/lt20kbytesfile
$xdsmc sel $SCRATCHDIR/gt20kbytesfile
$xdsmc arc $SCRATCHDIR/lt20kbytesfile.arc
```

```
$xdsmadmc 'query drmedia * wherestate=mountable'
$xl grep "bk05" $DSMOUT
$xver V2 $? -ne 0
$xl grep "bk99" $DSMOUT
$xver V3 $? -ne 0
$xl grep "bk06" $DSMOUT
$xver V4 $? -ne 0

$xl banner "backup adsm server stgpools"

$xdsmadmc backup stgpool backuppool cstoragepf
$xdsmadmc backup stgpool backuppoolf cstoragepf
$xdsmadmc backup stgpool archivepool cstoragepfa
$xl sleep 30

$xl banner "backup adsm server database"
$xdsmadmc backup db type=full devclass=files \
   volumename=$SERVERDIR/bk06
$xl sleep 10

#
$xl banner 'what backup volumes were created?'
#

$xdsmadmc 'query drmedia * wherestate=mountable'
$xl grep "bk05" $DSMOUT
$xver V5 $? -eq 0
$xl grep "bk99" $DSMOUT
$xver V6 $? -eq 0
$xl grep "bk06" $DSMOUT
$xver V7 $? -eq 0

$xl banner 'eject the volumes from the library'

$xdsmadmc 'move drmedia * wherestate=mountable'
$xl sleep 10

$xdsmadmc 'query drmedia * wherestate=mountable'
$xl grep "bk05" $DSMOUT
$xver V8 $? -ne 0
$xl grep "bk99" $DSMOUT
$xver V9 $? -ne 0
$xl grep "bk06" $DSMOUT
$xver V10 $? -ne 0

$xdsmadmc 'query drmedia * wherestate=notmountable'
$xl grep "bk05" $DSMOUT
$xver V11 $? -eq 0
$xl grep "bk99" $DSMOUT
$xver V12 $? -eq 0
$xl grep "bk06" $DSMOUT
$xver V13 $? -eq 0

$xl banner 'hand the volumes to the courier'

$xdsmadmc 'move drmedia * wherestate=notmountable'
$xl sleep 10
```

```
$xdsmadmc 'query drmedia * wherestate=notmountable'
$xl grep "bk05" $DSMOUT
$xver V14 $? -ne 0
$xl grep "bk99" $DSMOUT
$xver V15 $? -ne 0
$xl grep "bk06" $DSMOUT
$xver V16 $? -ne 0

$xdsmadmc 'query drmedia * wherestate=courier'
$xl grep "bk05" $DSMOUT
$xver V17 $? -eq 0
$xl grep "bk99" $DSMOUT
$xver V18 $? -eq 0
$xl grep "bk06" $DSMOUT
$xver V19 $? -eq 0

$xl banner 'create the plan file'
$xdsmadmc prepare planprefix=$DRMPLANDIR/
$xl sleep 10

$xp display the plan file
ls $DRMPLANDIR/19* | tail -1 | read PFN
$xl echo $PFN
$xl cat $PFN

$xl banner 'day 2'
$xl date 9506021200

cp $SCRATCHDIR/lt20kbytesfile $SCRATCHDIR/lt20kbytesfile2 \
  1>> $TRACEFILE 2>&1

$xdsmc sel $SCRATCHDIR/lt20kbytesfile2
$xl sleep 10

$xl banner 'backup adsm stgpool and db'

$xdsmadmc define vol cstoragepf $SERVERDIR/bk07
$xdsmadmc backup stgpool backuppool cstoragepf
$xl sleep 10
$xdsmadmc backup db type=full devclass=files \
   volumename=$SERVERDIR/bk08
$xl sleep 10

$xl banner 'get the new backup volumes offsite'
$xdsmadmc 'query drmedia * wherestate=mountable'
$xl grep "bk07" $DSMOUT
$xver V20 $? -eq 0
$xl grep "bk08" $DSMOUT
$xver V21 $? -eq 0

$xdsmadmc 'move drmedia * wherestate=mountable'
$xl sleep 10

$xdsmadmc 'query drmedia * wherestate=notmountable'
$xl grep "bk07" $DSMOUT
$xver V22 $? -eq 0
$xl grep "bk08" $DSMOUT
$xver V23 $? -eq 0
```

```
$xdsmadmc 'move drmedia * wherestate=notmountable'
$xl sleep 10

$xdsmadmc 'query drmedia * wherestate=courier'
$xl grep "bk07" $DSMOUT
$xver V24 $? -eq 0
$xl grep "bk08" $DSMOUT
$xver V25 $? -eq 0

$xl banner 'acknowledge receipt of previous sent volumes at vault'
$xdsmadmc move drmedia $SERVERDIR/bk05 wherestate=courier
$xdsmadmc move drmedia $SERVERDIR/bk99 wherestate=courier
$xdsmadmc move drmedia $SERVERDIR/bk06 wherestate=courier
$xl sleep 15

$xdsmadmc 'query drmedia * wherestate=vault'
$xl grep "bk05" $DSMOUT
$xver V26 $? -eq 0
$xl grep "bk99" $DSMOUT
$xver V27 $? -eq 0
$xl grep "bk06" $DSMOUT
$xver V28 $? -eq 0

$xl banner 'build the recovery plan'
$xdsmadmc prepare planprefix=$DRMPLANDIR/
$xl sleep 10

$xp display the plan file
ls $DRMPLANDIR/19* | tail -1 | read PFN
$xl echo $PFN
$xl cat $PFN

$xl banner 'day 3'
$xl date 9509031200

$xp "make copystg pool volume bk05 empty (simulate reclamation)"
$xdsmadmc update stgpool cstoragepf reusedelay=0
$xdsmadmc define vol cstoragepf $SERVERDIR/bk70
$xdsmadmc move data $SERVERDIR/bk05 stgpool=cstoragepf
$xl sleep 15

$xdsmadmc backup db type=inc devclass=files \
   volumename=$SERVERDIR/bk50
$xl sleep 10

$xl banner 'get the new backup volumes offsite'
$xdsmadmc 'query drmedia * wherestate=mountable'
$xl grep "bk70" $DSMOUT
$xver V29 $? -eq 0
$xl grep "bk50" $DSMOUT
$xver V30 $? -eq 0

$xdsmadmc 'move drmedia * wherestate=mountable'
$xl sleep 5
$xdsmadmc 'move drmedia * wherestate=notmountable'
$xl sleep 5

$xdsmadmc 'query drmedia * wherestate=courier'
$xl grep "bk70" $DSMOUT
```

```
$xver V31 $? -eq 0
$xl grep "bk50" $DSMOUT
$xver V32 $? -eq 0

$xl banner 'acknowledge receipt of previous sent volumes at vault'
$xdsmadmc move drmedia $SERVERDIR/bk07 wherestate=courier
$xl sleep 5
$xdsmadmc move drmedia $SERVERDIR/bk08 wherestate=courier
$xl sleep 5

$xdsmadmc 'query drmedia * wherestate=vault'
$xl grep "bk07" $DSMOUT
$xver V33 $? -eq 0
$xl grep "bk08" $DSMOUT
$xver V34 $? -eq 0

$xl banner 'some volumes no longer have valid data and need to return'
$xdsmadmc 'query drmedia * wherestate=vaultret'
$xl grep "bk05" $DSMOUT
$xver V35 $? -eq 0
$xl grep "bk06" $DSMOUT
$xver V36 $? -eq 0

$xl banner 'build the recovery plan'
$xdsmadmc prepare planprefix=$DRMPLANDIR/
$xl sleep 10

$xp "display the plan file"
ls $DRMPLANDIR/19* | tail -1 | read PFN
$xl echo $PFN
$xl cat $PFN

$xp "we will check plan file stanza SERVER.REQUIREMENTS later"
$xdsmadmc query db f=d
grep "Physical.Volumes" $DSMOUT | read SR1
$xl echo $SR1
$xdsmadmc query log f=d
grep "Available.Space" $DSMOUT | read SR2
$xl echo $SR2

$xdsmadmc query volhist
$xdsmadmc query vol

$xdsmadmc query actlog

$xl setclock
```

5.4.4 Simulate Disaster

The following script will simulate a disaster as outlined in the overview.

```
$xl banner 'simulate disaster'

$xl banner "trash adsm client machine"
$xl rm $SCRATCHDIR/lt20kbytesfile
$xl rm $SCRATCHDIR/lt20kbytesfile2
$xl rm $SCRATCHDIR/gt20kbytesfile
$xl rm $SCRATCHDIR/lt20kbytesfile.arc
$xl ls $SCRATCHDIR/lt20kbytesfile $SCRATCHDIR/lt20kbytesfile2 \
```

```
   $SCRATCHDIR/gt20kbytesfile $SCRATCHDIR/lt20kbytesfile.arc
$xver v37 $? -ne 0

$xl banner "trash adsm server machine"
$xl sleep 5
echo "y" | $xdsmadmc halt

$xl sleep 5
$xl rm $SERVERDIR/bk01x
$xl rm $SERVERDIR/bk03
$xl rm $SERVERDIR/volhistory.txtx
$xl rm $SERVERDIR/devconfig.txtx
$xl rm $SERVERDIR/dsmserv.optx
$xl rm $SERVERDIR/ar01

$xl rmlv -f $SVRNAME"db01x"
$xl rmlv -f $SVRNAME"lg01x"
$xl rmlv -f $SVRNAME"bk02"
$xl rm $SERVERDIR/db01x
$xl rm $SERVERDIR/lg01x

$xp "following 2 vols are expired or empty server backup volumes"
# (make sure we do not try to use them, hehe)
$xl rm $SERVERDIR/bk05
$xl rm $SERVERDIR/bk06
```

5.4.5 Recovery Using the Plan File

The following script recovers using the macros created by DRM.

```
$xl banner 'recover'

$xl banner "restore adsm server"

$xl banner "run locally written procedure to explode the recovery plan"
cd $DRMPLANDIR
awk -f localmods.awk $SERVERDIR/planexpl.awk.smp > planexpl.awk
$xp "get name of latest recovery plan file and run explode it"
ls 19* | tail -1 | read PFN
$xl awk -f planexpl.awk $PFN

$xp "view site specific instructions in the recovery plan"
$xl grep "in.charge.of.this.server.is.Jane.Doe" $DRMPLANDIR/ \
   RECOVERY.INSTRUCTIONS.GENERAL
$xver V70 $? -eq 0

$xp "view the required backup volumes listed in the recovery plan"
$xl grep "bk07" $DRMPLANDIR/RECOVERY.VOLUMES.REQUIRED
$xver V71 $? -eq 0
$xl grep "bk08" $DRMPLANDIR/RECOVERY.VOLUMES.REQUIRED
$xver V72 $? -eq 0
$xl grep "bk50" $DRMPLANDIR/RECOVERY.VOLUMES.REQUIRED
$xver V73 $? -eq 0
$xl grep "bk70" $DRMPLANDIR/RECOVERY.VOLUMES.REQUIRED
$xver V74 $? -eq 0
$xl grep "bk99" $DRMPLANDIR/RECOVERY.VOLUMES.REQUIRED
$xver V75 $? -eq 0
```

```
$xp "view the server requirements in the recovery plan"
grep "$SR1" $DRMPLANDIR/SERVER.REQUIREMENTS
$xver v80 $? -eq 0
grep "$SR2" $DRMPLANDIR/SERVER.REQUIREMENTS
$xver v81 $? -eq 0

$xp "view the recovery devices required in the recovery plan"
grep "Device Class Name: FILES" \
   $DRMPLANDIR/RECOVERY.DEVICES.REQUIRED
$xver v82 $? -eq 0
grep "Device Class Name: FILESSM" \
   $DRMPLANDIR/RECOVERY.DEVICES.REQUIRED
$xver v83 $? -eq 0

cd $SERVERDIR
$xl rm nohup.out

$xl cat \
   $DRMPLANDIR/RECOVERY.SCRIPT.DISASTER.RECOVERY.MODE

$xl banner 'restore the ADSM server database'
$DRMPLANDIR/RECOVERY.SCRIPT.DISASTER.RECOVERY.MODE \
   root $PW $SVRNAME

$xl cat $DRMPLANDIR/LOGANDDB.VOLUMES.CREATE.log
$xl cat $DRMPLANDIR/LOGANDDB.VOLUMES.INSTALL.log
$xl cat $DRMPLANDIR/PRIMARY.VOLUMES.DESTROYED.log
$xl cat $DRMPLANDIR/COPYSTGPOOL.VOLUMES.DESTROYED.log
$xl cat $DRMPLANDIR/COPYSTGPOOL.VOLUMES.AVAILABLE.log

$xdsmadmc query dbvol
$xdsmadmc query logvol
$xdsmadmc query vol f=d

$xl banner "what is top priority client node in bldg 21?"
$xdsmadmc 'query machine pri=1 bu=21 f=d'
$xl grep "ACCTSRECVSRV" $DSMOUT
$xver v38 $? -eq 0
$xdsmadmc 'query machine acctsrecvsrv f=ch'
$xl grep "logical.volumes.by.volume.group" $DSMOUT
$xver V90 $? -eq 0

$xl banner "restore client machine files from adsm server copy stg pools"

$xdsmc res $SCRATCHDIR/lt20kbytesfile
$xdsmc res $SCRATCHDIR/lt20kbytesfile2
$xdsmc res $SCRATCHDIR/gt20kbytesfile
$xdsmc ret $SCRATCHDIR/lt20kbytesfile.arc

$xl ls $SCRATCHDIR/lt20kbytesfile $SCRATCHDIR/lt20kbytesfile2 \
   $SCRATCHDIR/gt20kbytesfile $SCRATCHDIR/lt20kbytesfile.arc
xver v39 $? -eq 0

$xl banner "restore adsm server primary storage pools"

$xl cat $DRMPLANDIR/RECOVERY.SCRIPT.NORMAL.MODE

$DRMPLANDIR/RECOVERY.SCRIPT.NORMAL.MODE root $PW $SVRNAME
$xl sleep 30
```

```
$xl cat \ $DRMPLANDIR/PRIMARY.VOLUMES.REPLACEMENT.CREATE.log
$xl cat $DRMPLANDIR/PRIMARY.VOLUMES.REPLACEMENT.log
$xl cat $DRMPLANDIR/STGPOOLS.RESTORE.log

$xdsmadmc 'query drmedia * wherestate=mountable'
$xdsmadmc 'query drmedia * wherestate=notmountable'
$xdsmadmc 'query drmedia * wherestate=courier'
$xdsmadmc 'query drmedia * wherestate=vault'

$xl banner "reset db backup volumes ormstate to onsite"
$xdsmadmc update volhist $SERVERDIR/bk50 devcl=files  \
   ORMSTATE=MOUNTABLE
$xdsmadmc update volhist $SERVERDIR/bk08 devcl=files \
   ORMSTATE=MOUNTABLE
$xl sleep 5

$xdsmadmc 'query drmedia * wherestate=mountable'
$xdsmadmc 'query drmedia * wherestate=notmountable'
$xdsmadmc 'query drmedia * wherestate=courier'
$xdsmadmc 'query drmedia * wherestate=vault'

$xl banner "get our priceless db and stgpool backups back to the vault"
$xdsmadmc 'move drmedia * wherestate=mountable'
$xl sleep 5
$xdsmadmc 'move drmedia * wherestate=notmountable'
$xl sleep 5

$xp "Make sure we are restoring from restored primary pool."
$xdsmadmc 'query drmedia * wherestate=mountable'
$xdsmadmc 'query drmedia * wherestate=notmountable'
$xdsmadmc 'query drmedia * wherestate=courier'
$xl grep "bk70" $DSMOUT
$xver V40 $? -eq 0
$xl grep "bk50" $DSMOUT
$xver V41 $? -eq 0
$xl grep "bk07" $DSMOUT
$xver V42 $? -eq 0
$xl grep "bk08" $DSMOUT
$xver V43 $? -eq 0
$xl grep "bk99" $DSMOUT
$xver V44 $? -eq 0
$xdsmadmc 'query drmedia * wherestate=vault'

$xdsmadmc query vol f=d

$xl banner 'day to day operations'

$xl banner "restore client machine files from adsm server primary stg pools"

$xl rm $SCRATCHDIR/lt20kbytesfile
$xl rm $SCRATCHDIR/lt20kbytesfile2
$xl rm $SCRATCHDIR/gt20kbytesfile
$xl rm $SCRATCHDIR/lt20kbytesfile.arc
$xl ls $SCRATCHDIR/lt20kbytesfile $SCRATCHDIR/lt20kbytesfile2 \
   $SCRATCHDIR/gt20kbytesfile $SCRATCHDIR/lt20kbytesfile.arc
$xver v45 $? -ne 0
```

```
$xdsmc res $SCRATCHDIR/lt20kbytesfile
$xdsmc res $SCRATCHDIR/lt20kbytesfile2
$xdsmc res $SCRATCHDIR/gt20kbytesfile
$xdsmc ret $SCRATCHDIR/lt20kbytesfile.arc

$xl ls $SCRATCHDIR/lt20kbytesfile $SCRATCHDIR/lt20kbytesfile2 \
   $SCRATCHDIR/gt20kbytesfile $SCRATCHDIR/lt20kbytesfile.arc
$xver v46 $? -eq 0

# Testcase Cleanup

$xdsmadmc query actlog
$xdsmadmc halt
echo "You can restart the server with command:"
echo $SERVERDIR/$STARTFILE

# cat $SERVERDIR/nohup.out >> $TRACEFILE
cat $SCRATCHDIR/$TCNAME.stat | tee -a $TRACEFILE
echo "$TCNAME" | tee -a $TRACEFILE
grep "failed" $SCRATCHDIR/$TCNAME.stat
if [ $? -ne 0 -a -r $SCRATCHDIR/$TCNAME.stat ]
then
  banner "success" | tee -a $TRACEFILE
  exit 0
else
  banner "oops" | tee -a $TRACEFILE
  exit 1
fi
```

5.5 Recovery Plan File Stanzas

This book will not describe all of these stanzas because they are described in the *ADSM Administrator's Guide - SH35-0134*. Some samples can be found below along with a short description; these stanzas were created by a test case that used an 8mm tape unit as the output device.

The RECOVERY.DEVICES.REQUIRED file provides the recovery plan with details about the device necessary to read the backup volumes.

```
 Purpose: Description of the devices required to read the
          volumes listed in the recovery volumes required stanza.

             Device Class Name: LIB8MM
        Device Access Strategy: Sequential
            Storage Pool Count: 2
                   Device Type: 8MM
                        Format: DRIVE
         Est/Max Capacity (MB): 0.0
                   Mount Limit: 1
              Mount Wait (min): 120
         Mount Retention (min): 1
                  Label Prefix: ADSM
                       Library: REALTHING
                     Directory:
Last Update by (administrator): ROOT
         Last Update Date/Time: 09/03/95 14:03:53
```

The RECOVERY.INSTRUCTIONS.GENERAL file provides the administrator with instructions on what to do in case of a disaster.

```
 This ADSM server contains the backup and archive data for XYZ corporation
 accounts receivable system. It also is used by various end users in the
 finance and materials distribution organizations.
 The storage administrator in charge of this server is Jane Doe 444-561-3406.
 If a disaster is declared, here is the outline of steps that must be done.
 1. Determine the recovery site. Our alternate recovery site vendor is IBM
    BRS in Tampa, Fl, USA 213-884-5647.
 2. Get the list of required recovery volumes from this recovery plan file
    and contact our offsite vault so that they can start pulling the
    volumes for transfer to the recovery site.
 3. etc...
```

The RECOVERY.INSTRUCTIONS.INSTALL file contains the instructions on how to restore the server operating system.

```
 The base ADSM server system is AIX 4.1.2 running on an RS6K model 320.
 Use mksysb volume serial number svrbas to restore this system image.
 A copy of this mksysb tape is stored at the vault. There is also a copy
 in bldg 24 room 4 cabinet a. The image includes the ADSM server code.
 The system programmer responsible for this image is Fred Myers.
 Following are the instructions to do a mksysb based OS install:
                   .
                   .
                   .
```

The RECOVERY.INSTRUCTIONS.OFFSITE contains information about the off-site location.

```
Our offsite vaulting vendor is ironmnt.
Their telephone number is 514-555-2341. Our account rep is Joe Smith.
Our account number is 1239992. Their address is ...
Here is a map to their warehouse ...
Our courier is ...
```

The RECOVERY.SCRIPT.NORMAL.MODE script restores the primary storage pools.

```
# Purpose: This script contains the steps required to recover the server
#          primary storage pools. This mode allows you to return the
#          copy storage pool volumes to the vault and to run the
#          server as normal.
# Note: This script assumes that all volumes necessary for the restore
#   have been retrieved from the vault and are available. This script
#   assumes the recovery  environment is compatible (essentially the
#   same) as the original. Any deviations require modification to this
#   this script and the macros and shell scripts it runs. Alternatively,
#   you can use this script as a guide, and manually execute each step.

# Format replacement volumes in the primary storage pools (If any
# are implemented as disk but not logical volume.)
# Recovery administrator: Edit this section for your replacement
# volumes.
/home/guest/drmtest/prepare/plandir/DSM1509/PRIMARY.VOLUMES.REPLACEMENT. \
CREATE 2>&1
| tee /home/guest/drmtest/prepare/plandir/DSM1509/PRIMARY.VOLUMES. \
REPLACEMENT.CREATE.log

# Define replacement volumes in the primary storage pools. Must
# have different name than original.
# Recovery administrator: Edit this section for your replacement
# volumes.
dsmadmc -id=$1 -pass=$2 -serv=$3 -ITEMCOMMIT \
  -OUTFILE=/home/guest/drmtest/prepare/plandir/DSM1509/PRIMARY. \
  VOLUMES.REPLACEMENT.log
     macro /home/guest/drmtest/prepare/plandir/DSM1509/PRIMARY. \
     VOLUMES.REPLACEMENT

# Restore the primary storage pools from the copy storage pools.
dsmadmc -id=$1 -pass=$2 -serv=$3 -ITEMCOMMIT \
  -OUTFILE=/home/guest/drmtest/prepare/plandir/DSM1509/STGPOOLS. \
  RESTORE.log
     macro /home/guest/drmtest/prepare/plandir/DSM1509/STGPOOLS. \
     RESTORE
```

The RECOVERY.SCRIPT.DISASTER.RECOVERY.MODE script helps to restore the database and restart the server.

```
 # Purpose: This script contains the steps required to recover the server
 # to the point where client restore requests can be satisfied directly
 # from available copy storage pool volumes.
 # Note: This script assumes that all volumes necessary for the restore have
 #   been retrieved from the vault and are available. This script assumes
 #   the recovery  environment is compatible (essentially the same) as the
 #   original. Any deviations require modification to this script and the
 #   macros and shell scripts it runs. Alternatively, you can use this
 #   script as a guide, and manually execute each step.
 # Load the kernel extension.
/usr/lpp/adsmserv/bin/loadpkx -f pkmonx

 # Restore server options, volume history, device configuration files.
cp /home/guest/drmtest/prepare/plandir/DSM1509/DSMSERV.OPT \
    /home/guest/drmtest/prepare/tservr/DSM1509/dsmserv.optx
cp /home/guest/drmtest/prepare/plandir/DSM1509/VOLUME.HISTORY.FILE \
    /home/guest/drmtest/prepare/tservr/DSM1509/volhistory.txtx
cp /home/guest/drmtest/prepare/plandir/DSM1509/DEVICE.CONFIGURATION.FILE \
    /home/guest/drmtest/prepare/tservr/DSM1509/devconfig.txtx
export DSMSERV_CONFIG=/home/guest/drmtest/prepare/tservr/DSM1509/dsmserv.optx
export DSMSERV_DIR=/home/guest/drmtest/prepare/tservr/DSM1509

 # Create and format log and database files.
/home/guest/drmtest/prepare/plandir/DSM1509/LOGANDDB.VOLUMES.CREATE 2>&1 \
| tee /home/guest/drmtest/prepare/plandir/DSM1509/LOGANDDB.VOLUMES.CREATE.log

 # Install the log and database files.
/home/guest/drmtest/prepare/plandir/DSM1509/LOGANDDB.VOLUMES.INSTALL 2>&1 \
| tee /home/guest/drmtest/prepare/plandir/DSM1509/LOGANDDB.VOLUMES.INSTALL.log

 # Restore the ADSM server database to latest version backed up per the
 # volume history file.
dsmserv restore db todate=09/03/1995 totime=12:05:28

 # Start the server.
nohup dsmserv &
echo Please start new ADSM server console with command dsmadmc -CONSOLE.
read pause

 # Tell ADSM Server these copy storage pool tapes are available for use.
 # Recovery Administrator: Remove any volumes not obtained from the vault.
dsmadmc -id=$1 -pass=$2 -serv=$3 -ITEMCOMMIT \
  -OUTFILE=/home/guest/drmtest/prepare/plandir/DSM1509/COPYSTGPOOL.
  VOLUMES.AVAILABLE.log \
     macro /home/guest/drmtest/prepare/plandir/DSM1509/COPYSTGPOOL.
     VOLUMES.AVAILABLE

 # Volumes in this macro were not marked as 'offsite' at the time
 # PREPARE ran. They were likely destroyed in the disaster.
 # Recovery Administrator: Remove any volumes that were not destroyed.
dsmadmc -id=$1 -pass=$2 -serv=$3 -ITEMCOMMIT \
  -OUTFILE=/home/guest/drmtest/prepare/plandir/DSM1509/COPYSTGPOOL. \
  VOLUMES.DESTROYED.log
     macro /home/guest/drmtest/prepare/plandir/DSM1509/COPYSTGPOOL. \
     VOLUMES.DESTROYED

 # Recovery administrator: Delete entries for any primary volumes listed
 # here that were not destroyed.
dsmadmc -id=$1 -pass=$2 -serv=$3 -ITEMCOMMIT \
  -OUTFILE=/home/guest/drmtest/prepare/plandir/DSM1509/PRIMARY. \
  VOLUMES.DESTROYED.log
     macro /home/guest/drmtest/prepare/plandir/DSM1509/PRIMARY. \
     VOLUMES.DESTROYED
```

The RECOVERY.VOLUMES.REQUIRED provides a list of database and storage pool backups needed to recover the server. The locations of the volumes are also listed in this file.

```
Volumes required for data base restore

 Location = fedex
  Device Class = LIB8MM
  Volume Name =
   TPBK50
 Location = ironmnt
  Device Class = LIB8MM
  Volume Name =
   TAPE1P

Volumes required for storage pool restore

 Location = fedex
  Copy Storage Pool = CSTORAGEPF
  Device Class = LIB8MM
  Volume Name =
   DBTP02

 Location = ironmnt
  Copy Storage Pool = CSTORAGEPF
  Device Class = LIB8MM
  Volume Name =
   TPBK07

  Copy Storage Pool = CSTORAGEPFA
  Device Class = LIB8MM
  Volume Name =
   TPBK99
```

5.6 Media Status

Disaster recovery media movement is performed using the move drmedia command. The media volumes can have six different statuses.

The statuses possible for the media volumes are listed below:

- MOUNTABLE

  The volumes created by DRM are on-site and available for ADSM. Changing the status to NOTMOUNTABLE is done with the command:

  ```
  move drmedia * wherestate=notmountable
  ```

- NOTMOUNTABLE

  The volumes created by DRM are on-site and not available for ADSM. Changing the status to COURIER is done with the command:

  ```
  move drmedia * wherestate=notmountable
  ```

- COURIER

The volumes created by DRM are handed over to the courier to be taken off-site.

Changing the status to VAULT is done with the command:

```
move drmedia * wherestate=courier
```

- VAULT

  The volumes created by DRM are at the off-site location. If the volumes contains invalid data and the number of expiration days have passed, the status is automatically updated to VAULTRETRIEVE.

- VAULTRETRIEVE

  The volumes created by DRM are off-site but must be returned to the on-site location because they no longer contain valid data. Changing the status to COURIERRETRIEVE is done with the command:

  ```
  move drmedia * wherestate=vaultretrieve
  ```

- COURIERRETRIEVE

  The volumes flagged VAULTRETRIEVE are handed over to the courier to be brought on-site. After arriving on-site, they can be reused. Deleting the volumes, thereby freeing them for reuse, is done with the command:

  ```
  move drmedia * wherestate=courierretrieve
  ```

In all of the above move commands, the * can be replaced with the volume name found by using the Query MEDia command.

Chapter 6. Disaster Recovery Scenarios

This chapter will demonstrate recovery from different kind of disasters. The scenarios covered will include the loss of the database, the recovery log, various media and the entire server.

6.1 Test Environment

The test environment used in these scenarios was comprised of the following hardware and software:

6.1.1 Machine Belay

An RS/6000 model 530 used as both a server and a client.

6.1.1.1 Disks

The following disks were installed:

Name	Size (MB)	Volume Group	Address
hdisk0	670	rootvg	00-08-00-00
hdisk1	320	rootvg	00-08-00-01
hdisk2	400	rootvg	00-08-00-02
hdisk3	400	none	00-08-00-03

6.1.1.2 Tapes

The following tapes were installed:

Name	Type	Address
rmt0	8mm	00-08-00-06

6.1.1.3 SCSI Adapters

The following SCSI adapters were installed:

Name	Address
scsi0	00-07
scsi1	00-08

6.1.1.4 ADSM

The following ADSM configuration was used:

Entity	Size (MB)	Path
Data Base	5	/usr/lpp/adsmserv/bin/db.dsm
Recovery Log	9	/usr/lpp/adsmserv/bin/log.dsm

6.1.1.5 File System Location

The AIX operating system and LPPs were located on hdisk0 and hdisk1. The following file systems were located on hdisk2:

File System	Size (MB)	Mount Point
/dev/dblogcopy	40	/usr/lpp/adsmserv/bin/dblogcopy
/dev/adsmstg	120	/usr/lpp/adsmserv/bin/backuppool

6.2 Database Failure

This section will look at recovery from database failures, both with mirroring active and without.

6.2.1 Mirroring Active

For this scenario, a copy of the database, /usr/lpp/adsmserv/bin/db.dsm, has been created called /usr/lpp/adsmserv/bin/dblogcopy/db.copy. This was done by issuing the following commands:

```
> dsmfmt -m -db /usr/lpp/adsmserv/bin/dblogcopy/db.copy 5
define dbc /usr/lpp/adsmserv/bin/db.dsm /usr/lpp/adsmserv/bin/dblogcopy/db.copy
```

The output of Q DBV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Sync'd  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/db.dsm             rv/bin/dblogco-                            ined
                           py/db.copy
```

Now, the primary database, /usr/lpp/adsmserv/bin/db.dsm, is removed by simply deleting it from AIX. No problems will occur with the working of the server because the copy of the database will be automatically utilized instead. Only after stopping and starting the server will a message be generated to indicate that the primary database is offline. The system will display messages similar to the following:

```
ANR7801I Subsystem (master) PID is 3572.
ANR0900I Processing options file dsmserv.opt.
ANR0990I ADSM server restart-recovery in progress.
ANR9999D blkdisk.c(959): Error opening disk /usr/lpp/adsmserv/bin/db.dsm.
Explanation (open error): A file or directory in the path name does not exist
ANR0214W Database volume /usr/lpp/adsmserv/bin/db.dsm is in the offline state -
VARY ON required.
ANR0200I Recovery log assigned capacity is 8 megabytes.
ANR0201I Database assigned capacity is 4 megabytes.
```

The output of Q DBV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Off-L-  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/db.dsm     ine     rv/bin/dblogco-                            ined
                           py/db.copy
```

To recreate a new copy of the database and delete the offline database, issue the following commands:

```
> dsmfmt -m -db /usr/lpp/adsmserv/bin/dblogcopy/db.copy2 5
define dbc /usr/lpp/adsmserv/bin/dblogcopy/db.copy /usr/lpp/adsmserv/bin
/dblogcopy/db.copy2
delete dbv /usr/lpp/adsmserv/bin/db.dsm
```

The output of Q DBV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Sync'd  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/dblogco-           rv/bin/dblogco-                            ined
 py/db.copy                py/db.copy2
```

If it is required to restore the situation to the state that it was before the failure of the db.dsm database, the following commands must be issued:

```
> dsmfmt -m -db /usr/lpp/adsmserv/bin/db.dsm 5
define dbc /usr/lpp/adsmserv/bin/dblogcopy/db.copy /usr/lpp/adsmserv/bin/db.dsm
vary off /usr/lpp/adsmserv/bin/dblogcopy/db.copy2
delete dbv /usr/lpp/adsmserv/bin/dblogcopy/db.copy2
```

And then remove the file /usr/lpp/adsmserv/bin/dblogcopy/db.copy2. The output of Q DBV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Sync'd  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/db.dsm             rv/bin/dblogco-                            ined
                           py/db.copy
```

6.2.2 Mirroring Inactive

If mirroring has not been implemented for the database, it will be necessary to take full and incremental backups of the database. First, a device class must be designated as the backup destination. A file has been defined with the name dbbackupname for this purpose. The definition is done by the command:

```
def dev dbbackupname devt=file
```

The database backup can be initiated in two ways, both of which are described below:

1. DBBACKUPTRIGGER

   Set the DBBACKUPTRIGGER by issuing the command:

   ```
   def dbb dev=dbbackupname logf=70
   ```

   As a result of doing this, a backup of the database is made every time the recovery log reaches 70 percent utilization, and the backup is made to a device called dbbackupname.

2. Manually

   It is possible to manually initiate the backups with the command:

   ```
   ba db dev=dbbackupname t=f   (for a full backup)
   ba db dev=dbbackupname t=i   (for an incremental backup)
   ```

For both methods, it is advisable to set the logmode to roll-forward with the command:

```
set logm r
```

This mode allows restore to the most current state, just before the failure occurred, by using the most recent database backup series and recover log records.

If the database is now removed by deleting the database file, the following messages will be observed when the server is restarted:

```
ANR7800I DSMSERV generated at 16:17:16 on Oct 17 1995.

ADSTAR Distributed Storage Manager for AIX-RS/6000
Version 2, Release 1, Level 0.1/0.1

Licensed Materials - Property of IBM

5765-564 (C) Copyright IBM Corporation 1990, 1995. All rights reserved.
U.S. Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corporation.

ANR7801I Subsystem (master) PID is 3389.
ANR0900I Processing options file dsmserv.opt.
ANR0990I ADSM server restart-recovery in progress.
ANR9999D blkdisk.c(959): Error opening disk /usr/lpp/adsmserv/bin/db.dsm.
Explanation (open error): A file or directory in the path name does not exist
ANR0259E Unable to read complete restart/checkpoint information from any
database or recovery log volume.
```

A new database must now be formatted using the following command:

```
> dsmfmt -m -db /usr/lpp/adsmserv/bin/db.dsm 5
```

The database can now be restored using the following command:

```
> dsmserv restore db
```

This will restore the database to its most current state. The server will need to be restarted subsequently.

6.3 Recovery Log Failure

This section will look at recovery from recovery log failures, both with mirroring active and without.

6.3.1 Mirroring Active

For this scenario, a copy of the recovery log, /usr/lpp/adsmserv/bin/log.dsm, called /usr/lpp/adsmserv/bin/dblogcopy/log.copy, has been created. This was done by issuing the following commands:

```
> dsmfmt -m -log /usr/lpp/adsmserv/bin/dblogcopy/log.copy 9
define logc /usr/lpp/adsmserv/bin/log.dsm /usr/lpp/adsmserv/bin/dblogcopy
/log.copy
```

The output of Q LOGV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Sync'd  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/log.dsm           rv/bin/dblogco-                             ined
                          py/log.copy
```

Now, the primary recovery log, /usr/lpp/adsmserv/bin/log.dsm, is removed by simply deleting it. There will not be any problems in the working of the server because the copy of the recovery log will be automatically utilized instead. Only after stopping and starting the server will messages be seen indicating that the primary recover log is offline. The screen will show messages similar to the following:

```
ANR7801I Subsystem (master) PID is 11645.
ANR0900I Processing options file dsmserv.opt.
ANR0990I ADSM server restart-recovery in progress.
ANR9999D blkdisk.c(959): Error opening disk /usr/lpp/adsmserv/bin/log.dsm
Explanation (open error): A file or directory in the path name does not exist
ANR0215W Recovery log volume /usr/lpp/adsmserv/bin/log.dsm is in the offline
state - VARY ON required.
ANR0200I Recovery log assigned capacity is 8 megabytes.
ANR0201I Database assigned capacity is 4 megabytes.
```

The output of Q LOGV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Off-L-  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/log.dsm    ine     rv/bin/dblogco-                             ined
                           py/log.copy
```

To recreate a copy of the recovery log and to delete the offline copy, issue the following commands:

```
> dsmfmt -m -log /usr/lpp/adsmserv/bin/dblogcopy/log.copy2 9
define logc /usr/lpp/adsmserv/bin/dblogcopy/log.copy /usr/lpp/adsmserv/bin
/dblogcopy/log.copy2
delete logv /usr/lpp/adsmserv/bin/log.dsm
```

The output of Q LOGV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Sync'd  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/dblogco-            rv/bin/dblogco-                           ined
 py/log.copy                py/log.copy2
```

If it is required to restore the situation to the state that it was in before the failure of the log.dsm recovery log, the following commands must be issued:

```
> dsmfmt -m -log /usr/lpp/adsmserv/bin/log.dsm 9
define logc /usr/lpp/adsmserv/bin/dblogcopy/log.copy /usr/lpp/adsmserv/bin
/log.dsm
vary off /usr/lpp/adsmserv/bin/dblogcopy/log.copy2
delete logv /usr/lpp/adsmserv/bin/dblogcopy/log.copy2
```

Finally, remove the file /usr/lpp/adsmserv/bin/dblogcopy/log.copy2. The output of Q LOGV now looks like:

```
Volume Name       Copy    Volume Name       Copy    Volume Name       Copy
(Copy 1)          Status  (Copy 2)          Status  (Copy 3)          Status
----------------  ------  ----------------  ------  ----------------  ------
/usr/lpp/adsmse-  Sync'd  /usr/lpp/adsmse-  Sync'd                    Undef-
 rv/bin/log.dsm             rv/bin/dblogco-                           ined
                            py/log.copy
```

6.3.2 Mirroring Inactive

The recovery of a damaged recovery log without mirroring is much more difficult than with mirroring. The following files must be available:

- A backup of the volume history file
- A backup of the device configuration file
- A backup of the database.

The server options file should also be updated with the VOLUMEHistory and the DEVCONFIG parameters.

In this test case, the following files are used.

Volume history backup file /usr/lpp/adsmserv/bin/dblogcopy/volhistbackup

Device configuration backup file /tmp/devconfig.out

Database backup device dbbackupname

The following two entries were made in the server options file:

VOLUMEH /usr/lpp/adsmserv/bin/dblogcopy/volhistbackup

DEVCONFig /tmp/devconfig.out

The volume history file backup is made with the command:

```
ba volh f=/usr/lpp/adsmserv/bin/dblogcopy/volhistbackup
```

The device configuration file backup is made with the command:

```
ba devconf f=/tmp/devconfig.out
```

To simulate the damaging of the recovery log, the entire file was simply deleted and the ADSM server stopped and restarted. Messages similar to the following will be seen when the server restarts:

```
ANR7801I Subsystem (master) PID is 11543.
ANR0900I Processing options file dsmserv.opt.
ANR0990I ADSM server restart-recovery in progress.
ANR9999D blkdisk.c(959): Error opening disk /usr/lpp/adsmserv/bin/log.dsm.
Explanation (open error): A file or directory in the path name does not exist.
ANR0259E Unable to read complete restart/checkpoint information from any
database or recovery log volume.
```

To recover from this disaster, the following commands need to be issued:

1. Dump the database.

   ```
   > dsmserv dumpdb dev=dbbackupname
   ```

   The output will look similar to the following:

   ```
   ANR7801I Subsystem (master) PID is 11546.
   ANR0900I Processing options file dsmserv.opt.
   ANR0990I ADSM server restart-recovery in progress.
   ANR9999D blkdisk.c(959): Error opening disk /usr/lpp/adsmserv/bin/log.dsm.
   Explanation (open error): A file or directory in the path name does not exist.
   ANR0215W Recovery log volume /usr/lpp/adsmserv/bin/log.dsm is in the offline
   state - VARY ON required.
   ANR0200I Recovery log assigned capacity is 8 megabytes.
   ANR0201I Database assigned capacity is 4 megabytes.
   ANR4000I DUMPDB: Database dump process started.
   ANR8340I FILE volume /usr/lpp/adsmserv/bin/17679102.DMP mounted.
   ANR1360I Output volume /usr/lpp/adsmserv/bin/17679102.DMP opened (sequence
   number 1).
   ANR1361I Output volume /usr/lpp/adsmserv/bin/17679102.DMP closed.
   ANR4031I DUMPDB: Copied 38 database pages.
   ANR4033I DUMPDB: Copied 3 bit vectors.
   ANR4034I DUMPDB: Encountered 0 bad database pages.
   ANR4036I DUMPDB: Copied 214 database entries.
   ANR4037I DUMPDB: 28 Kilobytes   copied.
   ANR4001I DUMPDB: Database dump process completed.
   ```

   Write down the volume name /usr/lpp/adsmserv/bin/17679102.DMP, since this will be required for the loaddb step.

2. Format a new log volume.

   ```
   > dsmfmt -m -log /usr/lpp/adsmserv/bin/log.dsm 9
   > dsmserv install 1 /usr/lpp/adsmserv/bin/log.dsm 1 /usr/lpp/adsmserv/bin/db.dsm
   ```

The output will look similar to the following:

```
ANR7801I Subsystem (master) PID is 8226.
ANR0900I Processing options file dsmserv.opt.
ANR0300I Recovery log format started; assigned capacity 8 megabytes.
ANR0301I Recovery log format in progress; 4 megabytes of 8.
ANR0301I Recovery log format in progress; 8 megabytes of 8.
ANR0302I Recovery log formatting took 9616 milliseconds.
ANR0303I   Format rate:         213.0  pages/second.
ANR0304I   Page service time:     4.7  ms.
ANR0305I Recovery log format complete.
ANR0306I Recovery log volume mount in progress.
ANR0353I Recovery log analysis pass in progress.
ANR0354I Recovery log redo pass in progress.
ANR0355I Recovery log undo pass in progress.
ANR0352I Transaction recovery complete.
ANR0992I ADSM server installation complete.
```

3. Reload the database.

```
> dsmserv loaddb dev=dbbackupname vol=<volumename>
```

Where <volumename> is the name noted in the previous step, /usr/lpp/adsmserv/bin/17679102.DMP, in this case. The output will be similar to the following:

```
ANR7801I Subsystem (master) PID is 12073.
ANR0900I Processing options file dsmserv.opt.
ANR0990I ADSM server restart-recovery in progress.
ANR0200I Recovery log assigned capacity is 8 megabytes.
ANR0201I Database assigned capacity is 4 megabytes.
ANR0306I Recovery log volume mount in progress.
ANR0353I Recovery log analysis pass in progress.
ANR0354I Recovery log redo pass in progress.
ANR0355I Recovery log undo pass in progress.
ANR0352I Transaction recovery complete.
ANR4003I LOADDB: Database load process started.
ANR8340I FILE volume /usr/lpp/adsmserv/bin/17679102.DMP mounted.
ANR1363I Input volume /usr/lpp/adsmserv/bin/17679102.DMP opened (sequence
number 1).
ANR4038I LOADDB: Loading database information dumped on 11/29/95 at 15:05:03.
ANR1364I Input volume /usr/lpp/adsmserv/bin/17679102.DMP closed.
ANR4032I LOADDB: Copied 38 database records.
ANR4033I LOADDB: Copied 3 bit vectors.
ANR4035I LOADDB: Encountered 0 bad database records.
ANR4036I LOADDB: Copied 214 database entries.
ANR4037I LOADDB: 28 Kilobytes   copied.
ANR4004I LOADDB: Database load process completed.
ANR4405I LOADDB: Loaded an inconsistent dump image - a database audit (AUDITDB)
IS REQUIRED.
```

4. Audit the database.

```
> dsmserv auditdb fix=yes
```

The output will be similar to the following:

```
ANR7801I Subsystem (master) PID is 10549.
ANR0900I Processing options file dsmserv.opt.
ANR0990I ADSM server restart-recovery in progress.
ANR0200I Recovery log assigned capacity is 8 megabytes.
ANR0201I Database assigned capacity is 4 megabytes.
ANR0306I Recovery log volume mount in progress.
ANR0353I Recovery log analysis pass in progress.
ANR0354I Recovery log redo pass in progress.
ANR0355I Recovery log undo pass in progress.
ANR0352I Transaction recovery complete.
ANR4140I AUDITDB: Database audit process started.
ANR4075I AUDITDB: Auditing policy definitions.
ANR4040I AUDITDB: Auditing client node and administrator definitions.
ANR4135I AUDITDB: Auditing central scheduler definitions.
ANR2833I AUDITDB: Auditing license definitions.
ANR4136I AUDITDB: Auditing server inventory.
ANR4137I AUDITDB: Auditing inventory file spaces.
ANR4307I AUDITDB: Auditing inventory external space-managed objects.
ANR4138I AUDITDB: Auditing inventory backup objects.
ANR4139I AUDITDB: Auditing inventory archive objects.
ANR4310I AUDITDB: Auditing inventory space-managed objects.
ANR4230I AUDITDB: Auditing data storage definitions.
ANR4264I AUDITDB: Auditing file information.
ANR4265I AUDITDB: Auditing disk file information.
ANR4266I AUDITDB: Auditing sequential file information.
ANR4256I AUDITDB: Auditing data storage definitions for disk volumes.
ANR4263I AUDITDB: Auditing data storage definitions for sequential volumes.
ANR6646I AUDITDB: Auditing machine and recovery media definitions.
ANR4210I AUDITDB: Auditing physical volume repository definitions.
ANR4141I AUDITDB: Database audit process completed.
```

5. Restart the server.

   The ADSM server can now be restarted.

Remember that the time taken for the dump, load and audit steps will be dependent upon the size of the database and can take considerable amounts of time. It is therefore strongly recommended that at least one mirrored copy of the recovery log be maintained.

6.4 Media Failure

For this scenario, a new storage pool, called backuptest, and a copy storage pool, called backuptestcopy, have been created.

The backuptest storage pool has five volumes as shown below:

```
            Volume Name: /vol1/vol1
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 0.4
          Volume Status: On-Line

            Volume Name: /vol2/vol2
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 0.4
          Volume Status: On-Line

            Volume Name: /vol3/vol3
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 0.4
          Volume Status: On-Line

            Volume Name: /vol4/vol4
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 0.4
          Volume Status: On-Line

            Volume Name: /vol5/vol5
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 0.4
          Volume Status: On-Line
```

The backuptest copy storage pool has one volume as shown below:

```
            Volume Name: TEST
      Storage Pool Name: BACKUPTESTCOPY
      Device Class Name: 8MILL
Estimated Capacity (MB): 0.0
                  %Util: 0.0
          Volume Status: Empty
```

After having made a client backup, the backuptest volumes looked as follows:

```
            Volume Name: /vol1/vol1
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 13.7
          Volume Status: On-Line

            Volume Name: /vol2/vol2
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 3.1
          Volume Status: On-Line

            Volume Name: /vol3/vol3
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 3.1
          Volume Status: On-Line

            Volume Name: /vol4/vol4
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 16.0
          Volume Status: On-Line

            Volume Name: /vol5/vol5
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 8.2
          Volume Status: On-Line
```

Now the backup of the primary storage pool, backuptest, is made to the copy storage pool, backuptestcopy, using the command:

```
ba stg backuptest backuptestcopy
```

Output of this command looked like:

```
ANR8326I 001: Mount 8MM volume TEST R/W in drive 23GIGMT0 (/dev/mt0)
of library 8MILL within 10 minutes.
ANR8335I 001: Verifying label of 8MM volume TEST in drive 23GIGMT0 (/dev/mt0)
ANR8328I 001: 8MM volume TEST mounted in drive 23GIGMT0 (/dev/mt0)
ANR1212I Backup process 9 ended for storage pool BACKUPTEST
ANR1214I Backup of primary storage pool BACKUPTEST to copy storage pool
BACKUPTESTCOPY has ended. Files Backed Up: 36, Bytes Backed Up: 442368,
Unreadable Files: 0, Unreadable Bytes: 0.
```

The copy storage pool volume now looked as follows:

```
            Volume Name: TEST
      Storage Pool Name: BACKUPTESTCOPY
      Device Class Name: 8MILL
Estimated Capacity (MB): 2,472.0
                  %Util: 0.0
          Volume Status: Filling
```

For the media failure test, volume /vol1/vol1 was deleted. A media failure will usually be noticed first by a client attempting to access the destroyed volume. The following is an example of messages that might be received:

```
ANR1307I Disk volume /vol1/vol1 varied offline.
ANR1421W Read access denied for volume /vol1/vol1 - volume offline.
ANR0542W Retrieve or restore failed for session 2 for node CLIENT (AIX) -
storage media inaccessible.
```

After the failure, the backuptest volumes looked as follows:

```
            Volume Name: /vol1/vol1
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 0.0
                  %Util: 0.0
          Volume Status: Off-Line

            Volume Name: /vol2/vol2
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 3.1
          Volume Status: On-Line

            Volume Name: /vol3/vol3
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 3.1
          Volume Status: On-Line

            Volume Name: /vol4/vol4
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 16.0
          Volume Status: On-Line

            Volume Name: /vol5/vol5
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 8.2
          Volume Status: On-Line
```

For the restore of the data that was on volume /vol1/vol1, it is necessary to allocate new space for /vol1/vol11 that has at least the same size as the destroyed /vol1/vol1 and it must then be added to the primary storage pool backuptest. The following commands will do this:

```
> dsmfmt -m -data /vol1/vol11 1
def vol backuptest /vol1/vol11
```

The backuptest volumes now looked as follows:

```
            Volume Name: /vol1/vol1
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 0.0
                  %Util: 0.0
          Volume Status: Off-Line

            Volume Name: /vol1/vol1
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 0.4
          Volume Status: On-Line

            Volume Name: /vol2/vol2
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 3.1
          Volume Status: On-Line

            Volume Name: /vol3/vol3
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 3.1
          Volume Status: On-Line

            Volume Name: /vol4/vol4
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 16.0
          Volume Status: On-Line

            Volume Name: /vol5/vol5
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                  %Util: 8.2
          Volume Status: On-Line
```

Restoring the data is performed with the command:

```
restore volume /vol1/vol1
```

The output of this command will be similar to the following:

```
ANR2114I RESTORE VOLUME: Access mode for volume /vol1/vol1 updated to
"destroyed".
ANR1232I Restore of volumes in primary storage pool BACKUPTEST started
as process 2.
ANR1254I Removable volume TEST is required for restore processing.
ANR2110I RESTORE VOLUME started as process 2.
ANR8324I 8MM volume TEST is expected to be mounted (R/W).
ANR8326I 001: Mount 8MM volume TEST R/W in drive 23GIGMT0 (/dev/mt0) of
library 8MILL within 10 minutes.
ANR8335I 001: Verifying label of 8MM volume TEST in drive 23GIGMT0
(/dev/mt0).
ANR8328I 001: 8MM volume TEST mounted in drive 23GIGMT0 (/dev/mt0).
ANR1235I Restore process 2 ended for volumes in storage pool BACKUPTEST.
ANR1240I Restore of volumes in primary storage pool BACKUPTEST has
ended. Files Restored: 8, Bytes Restored: 123691. Unreadable Files: 0,
Unreadable Bytes: 0.
ANR2208I Volume /vol1/vol1 deleted from storage pool BACKUPTEST.
```

After this, the backuptest volumes looked as follows:

```
            Volume Name: /vol1/vol1
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                 %Util: 1.6
          Volume Status: On-Line

            Volume Name: /vol2/vol2
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                 %Util: 3.9
          Volume Status: On-Line

            Volume Name: /vol3/vol3
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                 %Util: 4.7
          Volume Status: On-Line

            Volume Name: /vol4/vol4
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                 %Util: 16.8
          Volume Status: On-Line

            Volume Name: /vol5/vol5
      Storage Pool Name: BACKUPTEST
      Device Class Name: DISK
Estimated Capacity (MB): 1.0
                 %Util: 17.2
          Volume Status: On-Line
```

All of the data is once again online and available.

6.5 Complete ADSM Server Failure

This section looks at recovery from a total failure of the ADSM server. The following protective measures had been taken:

- Database backup and mirroring
- Recover log mirroring
- Primary storage pool backup to copy storage pool
- Copy storage pool volumes are off-site
- Volume history file backup
- Device configuration file backup
- Server option file backup

For this scenario, it was assumed that all disk used by ADSM, that is the disks containing the database, recovery log and primary storage pool volumes, had been lost. Recovery was accomplished as in the following scenario:

1. Restore dsmserv.opt.

   The dsmserv.opt file is restored from backup tape.

2. Allocate disk space.

   Allocate new space for the database, recovery log and storage pool volumes by issuing the following commands:

```
> dsmfmt -m -db /usr/lpp/adsmserv/bin/db.dsm 5
> dsmfmt -m -log /usr/lpp/adsmserv/bin/log.dsm 9
> dsmfmt -m -data /vol1/vol11 1
> dsmfmt -m -data /vol2/vol21 1
> dsmfmt -m -data /vol3/vol31 1
> dsmfmt -m -data /vol4/vol41 1
> dsmfmt -m -data /vol5/vol51 1
```

3. Re-install server.

   The ADSM server is then re-installed using the following command:

```
> dsmserv install 1 log.dsm 1 db.dsm
```

4. Restore volume history and device configuration files.

   The volume history file and the device configuration file are restored from backup tape.

5. Restore the database.

   The database is restored using the following command:

```
> dsmserv restore db tod=today
```

6. Restart server.

   Now, start the server and update all lost volumes that have a status of DESTROYED. The following commands illustrate this process:

```
upd vol /vol1/vol1 acc=des
upd vol /vol2/vol2 acc=des
upd vol /vol3/vol3 acc=des
upd vol /vol4/vol4 acc=des
upd vol /vol5/vol5 acc=des
```

7. Attach new storage pools.

   Attach the newly allocated storage pool volumes to the existing storage pool. The following commands illustrate this process:

```
def vol backuptest /vol1/vol11
def vol backuptest /vol2/vol21
def vol backuptest /vol3/vol31
def vol backuptest /vol4/vol41
def vol backuptest /vol5/vol51
```

8. Restore storage pool data.

9. Now, data can be restored to the storage pools from the copy storage pools. First, the status of the copy storage pools must be changed from off-site to read/write. The following commands illustrate these steps:

```
upd vol test acc=readw
restore stg backuptest co=backuptestcopy
```

10. Execute a full database backup.

    It is advisable to initiate a full database backup at this point. This can be done as follows:

```
ba db dev=dbbackupname
```

11. Allocate and attach mirrored copies.

    If there were mirrored copies of the database and the recovery log, then new space must be allocated and the new copies attached using the commands shown below:

```
> dsmfmt -m -db /usr/lpp/adsmserv/bin/dblogcopy/db.copy 5
> dsmfmt -m -log /usr/lpp/adsmserv/bin/dblogcopy/log.copy 9
def dbc /usr/lpp/adsmserv/bin/db.dsm /usr/lpp/adsmserv/bin/dblogcopy
/db.copy
def logc /usr/lpp/adsmserv/bin/log.dsm  /usr/lpp/adsmserv/bin
/dblogcopy/log.copy
```

In this scenario, it was assumed that the operating system was unaffected by the server disk disaster. The person responsible for the operating system must ensure that there is always a good backup of the operating system available in the event that the operating system is also damaged. This usually takes the form of a making regular mksysb tapes.

The additional benefits of performing this process on a regular basis are that good copies of the dsmserv.opt file, the volume history file and the device configuration file will always be available on the backup tape.

6.6 Client and Server Failure

In order to restore after a client and server disaster, the above procedure can be used to restore the server. After a successful server recovery and reinstallation of the client operating system (if necessary), the client data can be restored from the ADSM server as normal.

6.7 Important Things to Remember

The following list contains some important tips that have been learned as a result of creating the test scenarios:

- Always mirror the Recovery Log

  Recovering without a mirrored recovery log was the most difficult task in these scenarios. In addition, mirroring the recovery log does not consume much space.

- Always keep one or more copies of the following files:
  - Device configuration file
  - Volume history file
  - Server option file
- Use off-site copy storage pools

  The use of off-site copy storage pool volumes was extremely beneficial.

- Most importantly:

Do Not Forget

The better the plan, the easier it will be to restore.

Part 2. ADSM Product Interrelationships

Chapter 7. ADSM in a Heterogeneous Network

Part two of this redbook concentrates on using ADSM in a modern distributed network environment and the subsequent opportunities for integration. Other publications have highlighted the abilities of ADSM to support software products from the point of view of data management; this redbook will therefore focus on integration from the other side and, in particular, with the products supplied with SystemView. The manner in which the components of SystemView can be used as they stand today and their potential interrelationships with ADSM will be investigated.

This chapter will focus on the following issues:

- The place of ADSM in a distributed network
- Reasons for integration

The succeeding chapters will examine:

1. The SystemView suite of products in overview format
2. Where it is advantageous to develop relationships between SystemView and ADSM using tools and features that are available today
3. The integration of SystemView and ADSM using these tools
4. Example scenarios that have utilized the integrated approach with some of the SystemView product set and ADSM

7.1 ADSM and the Network Environment

ADSM is designed to manage data on behalf of client systems and applications. Clients have access to a number of services provided through client software that executes on the client machine or through an API that allows client applications to directly take advantage of the services provided by ADSM.

The following services are available to clients:

- Backup and recovery of data
- Archive and retrieve of data
- Client hierarchical storage management
- Server hierarchical storage management
- Scheduling
- Storage hardware management

All of these services can be accessed from clients running on the same machine as the server or from other machines anywhere in a network. ADSM therefore provides a complete data management service to clients (both systems and applications).

In addition, the ADSM server provides a number of mechanisms to protect itself and the data that it is managing should there be a problem with the server or server hardware. These include the ability to backup the server database and recovery log and to store copies of critical user and server data on removable

media off-site. This area of ADSM management is discussed in detail in Part 1, “Disaster Recovery” on page 1.

However, when events occur that require administrator intervention, such as simple tape mount requests or perhaps more serious storage pool full conditions, action must be taken locally at the server. In many cases the intervention required can be effected remotely by logging in to the server machine, but the point is that the server messages must be monitored, either locally at the console or from an administrator session, and then action manually taken.

There are a number of products that provide monitoring and management services at this level, such as those products included in SystemView. The next section looks at what can be achieved with a greater degree of integration between ADSM and these products.

7.2 Why Integrate ADSM and SystemView?

As was stated in the previous section, ADSM provides excellent data management services to client systems and applications, but direct management of ADSM itself must be effected manually at present. The purposes of integrating ADSM with products such as those included with SystemView is to attempt to automate monitoring and management of ADSM to a higher degree. SystemView packages together systems management products that provide a comprehensive solution for managing most aspects of a computer system from the client perspective. The focus of this section of the book is to look at achieving a higher degree of integration between ADSM and the systems management products themselves.

There are various elements involved in running any application, systems management applications included. The following list looks at some of these elements and briefly discusses potential integration with ADSM. More detail on how and why integration might be possible is provided in Chapter 9, “Integrating ADSM” on page 81.

1. System resources

   The application must share the resources of the hardware that it is executing on. The JobScheduler systems management product included with SystemView is designed to assist in efficiently managing the scheduling of applications. This is more appropriate to applications or functions that need to be initiated on a regular basis; typically, an ADSM server will be running constantly. Furthermore, ADSM provides its own internal scheduling facilities for ADSM-related functions.

2. Management

   All applications require intervention from time to time. Whether they are resource related problems, internal issues or system or network errors, events will occur that require operator intervention. Management is really a twofold process:

   - Monitoring

     Management products must monitor applications in order to detect situations which require external intervention.

   - Resolving

Once a situation has been detected that requires intervention, the management product must know what actions need to be performed and must have the capability to perform them.

Two products provided with SystemView assist in this respect:

- Trouble Ticket/6000

  Trouble Ticket allows incidents to be logged and personnel notified when required for problem resolution. There are various situations that can occur with ADSM where this would be useful. A database full situation may be resolved by increasing the size of the database, but if the disk becomes full, then escalation is necessary as more disk space or a new disk may be required.

- NetView/6000

  NetView allows applications to be monitored and based on certain criteria actions to be implemented. There are many messages generated by ADSM that require operator intervention currently, but could be automatically resolved if the situation could be detected. Integration with NetView would allow this.

This kind of integration requires interroperation between the products and is the kind of task that is covered in this book.

3. System performance

   Applications must share the hardware resources available to them and many have performance constraints placed on them. For example, with ADSM, backups are generally scheduled to complete overnight while the system and network are relatively unused. Monitoring the loading and performance of the system can be vital to detecting situations where overloading may prevent the scheduled completion of a task. Products such as SysMon and the Performance Toolbox can assist in this kind of monitoring and products like JobScheduler or LoadLeveler can be used to prevent overloading. This kind of integration is really achieved at the planning level and is not really a direct product integration.

4. Network performance

   Applications must also share the network resource available to them and backup and archive are usually heavy consumers of network resource. Products such as RMON can be used to monitor network availability and utilization to ensure that adequate resource is available to allow completion of scheduled tasks. Again, products like JobScheduler and LoadLeveler can be used to prevent overloading. As mentioned in the discussion on system performance, this kind of integration is also achieved at the planning level and therefore not really a direct product integration.

5. Product installation and update

   All products need to be installed initially and all products will require updates and possibly PTFs during their lifetimes. There are regularly new releases of ADSM client and server software, and there will normally be several server machines in a network (for availability reasons) and many client machines. The manual distribution of software and updates is a time-consuming and error-prone task, and products like NetView DM have been designed to assist in this process. This level of integration is midway between planning and direct product integration and is be covered in this book.

6. System management

   The systems that applications actually execute on also need to be managed. In a networked environment this becomes more complex, and products such as DSMIT assist in centralizing the management of systems in a network. This has an impact on ADSM from the point of view of defining the various peripheral devices that ADSM servers may require, for example. This level of integration is not a direct product-level integration; it does, however, ease the process of managing ADSM servers in a networked environment.

Thus, the integration of ADSM with the SystemView products is concerned with the higher-level management of the ADSM server itself.

7.3 Summary

ADSM provides comprehensive data management services to client systems and applications in a networked environment. However, by definition, a networked environment is complex, and management of the ADSM server itself must currently be effected manually by administrators.

The purpose of looking at possible higher levels of integration with the SystemView products is to attempt to utilize some of the sophisticated systems management capabilities of the products to assist in the process of managing ADSM, and thereby providing a completely integrated solution for systems management.

Chapter 8. Product Overviews

This chapter provides a brief overview of the SystemView products that are discussed in this redbook. It is not by its nature an exhaustive examination and should not be regarded as a reference to the functions of the products.

8.1 Products Included

The following products are discussed as with regard to being integrated with ADSM:

1. Netview Distribution Manager
2. NetView/6000
3. Trouble Ticket/6000
4. RMON
5. Distributed SMIT
6. JobScheduler
7. Performance Toolbox
8. System Monitor/6000

These products will now be discussed in overview and those functions that are relevant to the integration of ADSM into the network management environment highlighted.

8.2 Netview Distribution Manager

NetView Distribution Manager is IBM′s software product for the distribution of software and data in the network environment. It also provides change control functions. It uses the client/server model, with one central RS/6000 acting as a change-control server connected via TCP/IP to a number of NetView Distribution Manager clients, which may be AIX platforms or any other supported platform. The supported nodes must each have the relevant NetView Distribution Manager change-control client installed.

This enables software or data held on the change control server to be shipped to the clients and installed or made available for use. The focus with regard to NetView Distribution Manager is to put it together with ADSM in such a way as to permit automated shipping of ADSM code and fixes.

8.3 NetView/6000

NetView is the network management tool of choice for networks of distributed heterogeneous systems. It provides the major management functions required for such systems, including:

- Notification

  NetView provides the capability to monitor the status of applications, systems and devices and possible errors (alerts) that may occur.

- Actions

NetView can initiate actions in response to a particular situation, for example driving a shell script or executable program in response to a particular notification.

- Reporting

  NetView can generate reports of online status in real time as events occur as well as report on historical or stored information, such as configuration or connectivity information.

In order to carry out these management actions, it may be necessary for a large number of systems or a large number of applications to be involved. In this redbook, a number of scenarios are detailed which illustrate this process on a small scale. These illustrate integrating ADSM into NetView today and show how other SystemView products may be involved in the overall management, not only of ADSM but of the whole network environment.

In its basic form, NetView is made up of two types of process:

- background
- foreground

The background processes (daemons) run continuously whether the user interface is started or not, while the foreground processes may only be invoked while the end-user interface is running. The processes may be regarded as performing five main functions:

- End-user presentation

  Managing the end-user interface.

- Network topology and database operations

  Dynamically discovering the layout of the IP network and logging the information into a number of databases. These processes also maintain and monitor that information.

- Host connection and trap routing

  Providing event and trap processing functions. For example, receiving and acting upon notification of network events. These processes also provide the communications paths from the host NetViews via AIX NetView Service Point.

- End-user Interface Application Programming Interface

  Providing the route through which external applications may drive the end-user presentation environment. There are a number of API groups each dealing with different aspects of the product.

- Object database

  This contains all of the data relating to the network resources being managed. Representations of the contents of this database may be displayed through the presentation environment. It should be noted that there are API calls that interact directly with the object database without invoking the presentation environment.

In considering the integration of ADSM into SystemView, the role of NetView is seen to be crucial.

8.4 Trouble Ticket/6000

Trouble Ticket/6000 provides an integrated set of applications, including system inventory, notification and trouble ticketing. These functions enable a more proactive approach to management of the day-to-day problems encountered in a network environment. Trouble Ticket enables controlled management from problem discovery to problem closure by providing diverse analysis tools and reporting abilities.

The system inventory function allows customers to keep an up-to-date, continuously maintained database of corporate assets including detailed information on:

- Network-attached devices
- Network-available applications
- Services required to support network operations

Trouble Ticket can track the problem history of each network device and each service used by the network, thereby giving users the information needed to run the network more efficiently. Comprehensive reports allow users to quickly identify chronic trouble spots, hardware or software failures, the status of all open trouble tickets, and evaluate the reliability history of a particular vendor′s products.

Problem management information is easily stored and retrieved in Trouble Ticket′s database facility. This facility gives the sophisticated user full access to all tables to allow, for example, the definition of new fields within the database. This enables total flexibility in accessing problem information.

The flexibility of the database is reflected in the user interface which is tailorable to meet the unique needs of individual customers. When the product is used in the scenarios later in the book, every effort is made to ensure as close an adherence as possible to the default installed format of database and end user interface.

8.5 RMON Overview

RMON is a LAN management tool that is designed to be integrated with NetView. It works to monitor token-ring or Ethernet LAN segments by gathering statistics on packets or octets and error counts, for example. These statistics can be monitored and warnings generated should any thresholds be exceeded. Through integration with NetView, the NetView operator is able to make use of the RMON facilities to manage performance, operational and problem management issues, thus extending the scope of the integrated systems management functions. RMON running on AIX is also able to control RMON agents within the network to gather LAN data and funnel it to the central point of control.

8.6 DSMIT

DSMIT for AIX provides the functionality of SMIT in a homogeneous or heterogeneous distributed systems environment. Using DSMIT, clusters of resources can be managed concurrently or sequentially. DSMIT is currently supported on AIX, SunOS and HP-UX.

DSMIT provides interactive menus and dialogs that allow commands to be automatically built, executed and logged, without the user having to learn and remember the specific command-level syntax required.

The functionality available is essentially that of SMIT, providing the ability to initiate backups of files, configure devices and manage other aspects of systems, such as printing, user definition and security consideration. DSMIT is, however, extendible, providing a systems management framework that can support central initiation of tasks on remote clusters of machines.

8.7 JobScheduler

JobScheduler for AIX is a software product provided by IBM in response to the increasing demand for background work processing on open systems. As the background load has increased, it has become increasingly resource-intensive to perform manual scheduling, monitoring of the workload and evaluation of conditional execution.

JobScheduler assists in these management tasks by implementing rules and policies defined by the systems administrator and then monitoring the workload through to completion. It automatically detects jobs that have failed and can restart them or optionally take some user-defined corrective action before the restart. Daily and weekly workload execution plans and views of historical information on processing may be generated. This is in addition to the processing audit trails held in the job log and execution log.

JobScheduler enables operations staff to manage large numbers of background jobs simultaneously and helps to reduce the chances of critical job failure. Throughput and turnaround times are improved, and skilled staff may spend more time managing exceptions rather than the workload itself.

JobScheduler consists of three components:

- Manager

  The manager runs on one node in the network and is responsible for scheduling tasks, submitting them to agents distributed throughout the network and then monitoring the results.

- Agents

  The agents run on all nodes of the network that are involved in workload execution. They receive jobs from the manager and execute them, sending the results back to the manager.

- User interface

  The user interface can be run from any node. This provides the facilities needed to define and monitor the workload managed.

8.8 Performance Toolbox

Performance Toolbox (PTX) is made up of several performance programs packaged together to provide a framework for performance management that is applicable to single nodes or to a distributed network. It can be used to augment the information gathered from SNMP network managers, such as NetView. Performance Toolbox provides finely detailed views into individual node and process performance in real time. Performance tuning and analysis controls are presented to the user through an X- and Motif-compliant end-user interface.

Performance Toolbox also includes Performance AIDE (PAIDE), which is the main provider of local AIX performance statistics. This can concurrently service multiple data requests from local or remote applications and do local data filtering and alert processing.

Performance Toolbox can be used to help a variety of users from programmers to network administrators in carrying out tasks such as creating performance profiles or debugging real-time problems. The focus of interest in Performance Toolbox, however, would be in using it to generate simple SNMP traps or performance-agent exceptions for user programs in response to a predefined local condition. These conditions may also be logged along with other information by Performance Toolbox.

8.9 System Monitor/6000

Systems Monitor/6000 is an SNMP sub-agent which is installed on an RS/6000 in the network. This sub-agent extends the enterprise-specific MIBs on the workstation so that large numbers of systems management attributes can be managed in the network; in this way, system configuration, CPU utilization, user information, or other items may be managed. This allows the network operator at a central point of control to ensure that critical resources within the network are operating correctly, thereby avoiding unnecessary downtime.

Most importantly in this consideration of the place of ADSM in the network environment, System Monitor is extendible to meet specific requirements. The aim is to show how System Monitor can be used to monitor ADSM in certain situations.

By using System Monitor to manage parts of the network, users may free up the main systems management resource to concentrate on more vital or complex parts of the network. To do this, a set of dynamically configured MIB tables is used in association with powerful filtering, threshold and analysis functions. This results in a lower workload for the central network controller as only crucial data is forwarded.

Any RS/6000 in the network may be used to run the System Monitor end-user interface. From this, the user can view, install and configure all System Monitor sub-agents in the network on any supported platform.

Chapter 9. Integrating ADSM

This chapter describes the ways in which the management of ADSM can be implemented using facilities of SystemView that are available now. The relevant features are discussed, and a number of scenarios that will be used to illustrate the integration are described. The actual implementation of these scenarios is described more fully in the subsequent chapters.

9.1 Integrating ADSM with NetView Distribution Manager

NetView DM/6000 can also be integrated with NetView/6000 in the sense that NetView DM can be initiated and controlled by NetView. The NetView database can also keep track of all of the NetView DM targets. However, in these scenarios, the focus of interest is on using NetView DM to distribute and maintain ADSM client and server code, which specifically involves designing the software packages that will contain the ADSM code and installation instructions.

NetView DM basically provides the following capabilities in a client/server environment:

- Software installation
- Data distribution
- Change control

There are, therefore, two elements to NetView DM: the change control server and the change control client. The server is responsible for central management and distribution of software packages which contain code and the instructions necessary to successfully install the code on client machines. Scheduling facilities also allow the times of distributions to be set for the most convenient opportunities, such as when client systems are not in use. The client contains the necessary facilities to interpret and implement the installation instructions at the client machine.

The next two sections look in more detail at the two components and are followed by a section describing a typical scenario for the distribution and maintenance of ADSM client and server code.

9.1.1 Integration Overview

This section will look at how integration might be achieved between NetView DM and ADSM.

- NetView DM change control server

  The NetView DM change control server supports the following functions:

  – Software preparation

    The builder, or individual responsible for preparing software packages for distribution, has access to NetView DM facilities that support the creation and testing of change files for each software package that will be distributed.

    ADSM client and server software and the appropriate installation instructions can be packaged for distribution in this way.

  – Automated distribution

The automated distribution facilities allow NetView DM to be used to perform the following distribution tasks:

- Scheduling and subsequent installation of software, updates and fixes to existing software on client machines
- Restoration of software on client machines to its state prior to any update
- Complete uninstallation of obsolete software on client machines
- Activation of software at client machines through control of client operations including system rebooting
- Maintaining of comprehensive records on software packages under the control of NetView DM using a history database
- Automatic installation of current level software on new machines as they are added to the network
- The sending, receiving, deleting and transferring between clients, of data files, including documentation, dumps and user data

The installation of ADSM client and server software as well as script files to implement initial setup, such as option file customization, can be easily effected. Documentation can also be distributed in this way.

– Scheduling

Installation or update of client machines can be scheduled by the server, and this has a number of benefits:

- Software distribution can be scheduled for times when client systems are not in use.
- Software distribution can be controlled to achieve network-wide synchronization of software levels
- Security and auditability

  The centralized mechanism provides for centralized tracking and security. The client software levels and inventories can be centrally tracked and managed.

Network-wide consistency of ADSM client and server code levels can be maintained using the scheduling facilities.

– Distributed control

The server can send commands to client machines to initiate changes while the client machine is unattended. Software can thus be installed, updated, downgraded or removed without the need for operator intervention at the client.

In addition, scripts can be executed at the client to perform additional customization tasks that may need to be performed before the new software can execute. User exits can be utilized to extend the capabilities of NetView DM at the client and server, allowing even greater functionality, if required. Examples include adding customized functions to execute on receipt of requests or completion notification.

These functions can be used to initiate the upgrade or installation of ADSM client and server software at opportune moments and to start the execution of any customization scripts.

Those functions relevant to the integration with ADSM in the scenarios in this book will be further highlighted in 9.1.2, “Scenario” on page 83.

- NetView DM change control client

  As has been mentioned previously, the NetView DM change control client, or agent, is responsible for receiving and implementing requests from the NetView DM server. The client can operate in one of two modes:

  – Push mode

    In push mode, both the NetView DM client workstation and the server can initiate software distribution.

  – Pull mode

    In pull mode, only the NetView DM client can initiate distribution.

  For the purposes of the integration with ADSM, distribution will only ever be required from the NetView DM server, therefore push mode will be used.

The next section will now look at the scenario that will be used to illustrate and test the integration of ADSM with NetView DM.

9.1.2 Scenario

This scenario concentrates on using NetView DM to package ADSM client and server code along with installation instructions and scripts that will allow centralized maintenance of ADSM software. To this end, the following steps are performed:

- Packaging

  The procedures necessary to package ADSM client and server software along with installation instructions and scripts are documented.

- Distribution

  The setup and maintenance of a centralized distribution service for the ADSM client and server software are documented.

- Installation/update

  The actual processes that occur in the successful installation or update of ADSM client and server software are documented.

There is one other step that needs to be performed before the ADSM server can be operational in a new environment and that is the local device configuration. This step can be accomplished using DSMIT from a central NetView DM server though for fully automatic operations, the script generated by SMIT for the device configuration could be used by NetView DM and this mechanism is covered.

Finally, there are further levels of integration that could be attained between NetView DM and ADSM, such as the centralized management of database and recovery log backup and transfer off-site (to other backup systems); the procedures to implement these are not covered at this time.

9.2 Integrating ADSM with NetView

As was discussed in 8.3, “NetView/6000” on page 75, the integration of ADSM and NetView is perhaps the key element in integrating ADSM with SystemView. NetView itself provides the capabilities to manage many of the other program products included with SystemView, and this management capability can provide major benefits when used to manage ADSM.

The central theme to this integration is providing a mechanism to transmit notifications of relevant ADSM events to NetView and then initiating some action as a result.

9.2.1 Integration Overview

NetView may receive notifications from ADSM in the following ways:

- Directly from shell scripts creating traps from ADSM server messages.
- Indirectly by writing ADSM server-specific messages to the AIX error log and configuring those messages as alertable by the SNMP agent.
- Indirectly from System Monitor generating notifications on recognition of configured events.

The following sections look at these methods in more detail:

- Direct Trap Creation

  This is the simplest form of notification to implement. A simple shell script may be used to read the ADSM server console messages and package them each into a trap which is directed at the network controller. The amount of preparation needed is minimal, but this approach is not without its drawbacks. The advantages of the approach are:

  – It is quick and very easy to implement.
  – Maintenance is relatively simple.
  – The skills to carry out this approach are widely available.

  The disadvantages may be summarized as follows:

  – It is a crude approach which over-simplifies the running of major networks.
  – The ADSM console receives many messages that are of little consequence to the network administrator. This may lead to problems of swamping the network controllers with information that is of little use. For example, while ADSM is waiting for tape mounts, it generates a message to this effect every minute for up to an hour, by default. The network administrators are unlikely to be interested in the first thirty or so tape mount messages, but after that time, it is likely that there is a problem that must be investigated.
  – If the implementation attempts to cater to all such eventualities, it is likely to result in an unwieldy set of programs that have a high level of skill invested in them and similarly high demands on maintenance resources.

- Indirect Trap Creation

  In this case, a program (usually shell script) is used to write the contents of ADSM messages into the AIX error log. This performs the same functions as the previous case, but with the advantage that the error log functions as a repository of information for problem determination and recording.

However, the disadvantages are similar to those of the direct approach, plus:

- There is duplication of information in the error log and ADSM logs
- Writing information into the error log is more complicated and requires far more detailed preparation than the direct method.
- The degree of customization to be carried out on the receiving NetView is much greater than in the direct case.

- System Monitor

  System Monitor sub-agents can be configured and used to monitor specific items of interest in a network. For example, sub-agents on ADSM client and server systems could be customized to monitor for specific events and provide notification to NetView upon detecting them.

  Systems Monitor will be described in more detail in 9.8, “Integrating ADSM with System Monitor” on page 91.

The next section describes an example scenario that illustrates the implementation of these methods to integrate ADSM with NetView.

9.2.2 Scenario

This scenario illustrates the steps required and the tasks necessary to allow NetView to effectively manage ADSM. This involves the following:

1. Trap and forward ADSM messages

   This can be accomplished using a short script to monitor the ADSM console and forward any messages on to NetView.

2. Configure NetView to recognize ADSM traps

   NetView must then be configured to recognize and act on notifications relating to ADSM messages. There are many different messages that may require actions and a widely varying level of complexity in implementation. Two typical issues are covered:

   - Automatically restarting a terminated server

     The ADSM server may halt for a number of reasons, and in most cases, it will not be deliberate. It is possible to arrange for NetView to detect this condition and restart the server.

     This situation is a useful example because it will illustrate the process necessary to start an ADSM server in the background.

   - Responding to a failure to mount a tape

     When ADSM is backing up data, the most common failure is that caused by a failure to mount a requested tape. In this scenario, two elements of SystemView are integrated with ADSM to give efficient response to such a failure.

     These are:

     - NetView - to give the network controller a concise view of status
     - Trouble Ticket - to give immediate logging of an error condition and tracking of responses.

     If one were to forward all tape mount messages that ADSM produces directly to a NetView display, the operator would quite quickly become swamped with useless information. Each tape mount request generates a

new message every minute until the tape is mounted or timed out, and the default timeout is to allow up to 60 minutes before timeout. The network operator might therefore be faced with a display of up to 60 messages for every tape mount requested by ADSM. If each ADSM server had, say, two backup devices and there are two servers in the network, then the NetView operator console could receive over 100 messages during an overnight run, without there being any real problem to report.

Clearly, if operator resource is to be used efficiently, then the aim should be to produce:

a. Notification at the NetView console that a mount has been requested at a server.

b. Notification that a mount request has not been fulfilled for a certain length of time and is now urgent.

c. Notification that a mount request has timed out before being fulfilled.

Further, once the operator is notified that an outstanding mount request is now urgent, he no longer needs to see the first notification that a mount request has been generated; it should therefore be removed from his display. The important information from his point of view is that there is an outstanding mount request that must be fulfilled within a short space of time. That information is contained in the latest notification; there is no need to reference the previous notification.

When ADSM generates a message to the effect that a tape has been mounted in response to a request, the network operator would prefer not to see that message; it is an ordinary event that should occur in the course of business at regular intervals. The action required on receipt of that message is in fact to delete any messages already displayed that relate to that particular tape mount request. Normally, this is what the operator would do; here that process will be automated.

The objectives of this scenario are to suppress all tape mount messages transmitted to the network operator except:

- The first such message - which informs the operator that a mount request has been generated
- The second message will remove the first from the NetView display.
- When a tape mount is fulfilled, that should remove all outstanding notifications from the operator display. It will not itself be displayed.
- When a tape mount times out, then:
  a. All previous notifications relevant to that mount request should be removed from the display
  b. A Trouble Ticket incident record is opened to track the problem.
  c. The NetView operator is informed that a mount request has timed out.

These two examples illustrate the bulk of the type of configuration and setup that needs to be done to enable a reasonable degree of integration between ADSM and NetView.

9.3 Integrating ADSM with Trouble Ticket

Trouble Ticket is an application that can be used in tandem with NetView. It is a client/server based application, with one server containing the database and a number of clients accessing the information from the server.

The problem management facilities provided include:

- Incident reporting
- Trouble ticketing
- Automatic notification of appropriate personnel
- Automatic escalation
- History of actions performed in resolving a problem

Since these facilities seem to provide a far greater degree of function than is actually required, the aim is to keep the use of Trouble Ticket as simple as possible.

9.3.1 Integration Overview

Trouble Ticket incidents can be created automatically or by users. In these scenarios, incidents will be created automatically. In a large organization, there may a number of levels of problem management, each of which requires some sort of escalation; this escalation may be provided automatically by Trouble Ticket. In these examples it will be presumed that there are only two levels of problem management and therefore only one escalation to be implemented.

An incident in these examples is a record of an event that has been created by NetView. A trouble ticket may be created from one or more incidents in order to track the underlying problem from discovery to resolution.

9.3.1.1 Incident Reports

In this consideration of Trouble Ticket in support of ADSM, an incident is the occurrence of an exception or problem situation within the ADSM environment. These situations will first be evident from ADSM messages which are notified to NetView; NetView may then automatically create a Trouble Ticket incident report.

All Trouble Ticket incident reports are accessed through the Incident Report menu. An incident report holds information about the exception that it relates to, such as:

- Trouble ticket number, if one has been assigned
- Report number
- Date on which the incident was recorded
- Summary of the facts
- Network resource reporting the incident
- Impact of this exception on that resource, if any

9.3.1.2 Trouble Tickets

Trouble tickets are in effect, groups of exceptions (incidents) that are tracked together. They are based on enterprise management policies and become the repository for all information regarding a particular problem, from first identification through to resolution. When closed (when the underlying problem has been fixed), Trouble Ticket keeps the ticket in a history database to assist in resolving similar problems in the future.

Associated with a particular trouble ticket at any time are:

- Ticket number
- Summary description
- Assigned priority
- Current escalation level
- Current status
- Name of the person to whom the ticket is currently assigned

9.3.2 Scenario

As has been described in this section on Trouble Ticket, any potential integration with ADSM will also depend on NetView. For this reason, a scenario involving Trouble Ticket is included in the section on NetView. This scenario involves extending the tape mount message example used in the NetView scenario to include Trouble Ticket.

9.4 Integrating ADSM with RMON

As described in 8.5, “RMON Overview” on page 77, RMON is also designed to be integrated with NetView. The RMON product works with RMON-compliant agents to monitor LANs and collect statistics to allow potential problems to be quickly detected and then resolved.

9.4.1 Integration Overview

In this capacity, there is no direct integration possible with ADSM. However, one of the major issues that arises with backup/archive and restore/retrieve operations, or indeed any network based client/server application, is network bandwidth. In its capacity as a LAN monitor, RMON can therefore be useful to monitor the traffic levels and status of any LANs involved in the network to ensure optimum network performance. Network performance is vital for two reasons:

1. Time constraints

   Degraded network performance can cause backups/archives to take longer than anticipated, and since most major backups are scheduled to complete overnight when network load is low, any overrun can impact network performance during critical business hours. For this reason, monitoring of network performance can be vital.

2. Event completion

   Degraded or inoperative networks may prevent the completion of backups which can have serious implications if a hardware failure then requires those backups. Early warning of network failure is therefore critical to the avoidance of this situation.

While highly important, this process does not involve any direct integration with ADSM and therefore is not covered further in this redbook.

9.4.2 Scenario

There are many potential processes that could be implemented to provide an integrated, cohesive approach to using RMON in this environment, but as stated in the previous section, these are really beyond the scope of this book.

Examples of such processes include:

- Starting RMON with JobScheduler to monitor the network during an ADSM run.
- Sending notifications to NetView of any issues.
- Utilization of NetView DM to transfer RMON data files to central location for centralized management.

9.5 Integrating ADSM with DSMIT

As described in 8.6, “DSMIT” on page 78, DSMIT provides the capabilities of the Systems Management Interface Tool (SMIT) in a distributed environment. This essentially involves providing the functions of SMIT in a distributed homogeneous or heterogeneous environment. Commands can be executed either concurrently on groups of machines or sequentially on them one at a time.

The kinds of task that SMIT assists in performing include defining and configuring devices, network configuration, printer management, and system resource management.

9.5.1 Integration Overview

DSMIT allows for the management of distributed systems from the point of view of system resources such as communications, device configuration and printing. It is also extendible, allowing other applications to make use of the easy to use interface for management or configuration purposes.

When ADSM is installed or when new devices are added, such as tapes, optical devices or libraries, they must be configured on the system, and this is most commonly done using SMIT. In a distributed environment, this same configuration could be implemented from a central point using DSMIT.

In the scenario described in 9.1.2, “Scenario” on page 83, ADSM client and server code, updates and fixes are distributed automatically from a central point. This distribution includes the necessary install instructions and scripts to automate the process. In terms of defining devices in new installations, or new devices, this process could be accomplished through DSMIT; if complete automation was required though, then a script could be written to perform the process.

9.5.2 Scenario

In the scenario described in 9.1.2, “Scenario” on page 83, DSMIT could have been used to define the devices for new ADSM installations. Examples are given instead of automating this process, and since this demonstrates the most likely level of integration with ADSM at this time, no further scenario is presented here.

9.6 Integrating ADSM with Job Scheduler

As described in 8.7, “JobScheduler” on page 78, JobScheduler provides extensive network job scheduling capabilities. It is designed to execute complex job streams anywhere in a network at the correct time and in the stipulated sequence. Execution progress is constantly monitored, and JobScheduler is capable of taking corrective actions when required and dynamically including alternate jobs.

9.6.1 Integration Overview

JobScheduler provides comprehensive scheduling services for network based environments and is specifically designed for the processing of background workloads. In this capacity, there is no direct integration possible with ADSM. Typically, the ADSM server process is required to be running constantly, and any scheduling requirements can be handled by the ADSM server.

It is possible to use JobScheduler to initiate the first execution of the ADSM server process, but this can also be effected by NetView DM, as discussed in 9.2.2, “Scenario” on page 85.

9.6.2 Scenario

Due to the lack of direct integration possibilities with ADSM, no scenario including JobScheduler is covered in this redbook.

9.7 Integrating ADSM with Performance Toolbox

As described in 8.8, “Performance Toolbox” on page 79, Performance Toolbox is a collection of tools that collectively provide advanced performance monitoring and display capabilities. In addition, it is possible to provide examples of performance characteristics recorded by Performance Toolbox that reflect unwanted operational conditions, such as performance bottlenecks. Upon recognizing a similar situation, some predefined corrective action can be initiated.

9.7.1 Integration Overview

The Performance Toolbox is really designed to assist in the monitoring of the performance characteristics of applications and the detection of performance anomalies that require correction. In this capacity, there is no real direct integration possible with ADSM. It would, however, be possible to use the Performance Toolbox to recognize situations where system load was degrading server performance and to then send notification to NetView for corrective action. Typically, a server system should be designed to be well balanced though even in the best planned situations, things can go wrong.

9.7.2 Scenario

Due to the lack of direct integration possibilities with ADSM, no scenario including Performance Toolbox is covered in this redbook.

9.8 Integrating ADSM with System Monitor

As described in 8.9, “System Monitor/6000” on page 79, System Monitor provides enhanced SNMP management capabilities through the following components:

- Mid-level manager

  The mid-level manager component provides an intermediate focal point for SNMP traffic. The mid-level manager can act as a funnel for SNMP information and can filter and perform actions based on traps received, thereby reducing the load on the central NetView manager.

- Systems information agent

  The systems information agent consists of an SNMP sub-agent that can extend the enterprise-specific MIBs on a workstation. The system information agent MIB can be extended to include application-specific information, allowing comprehensive monitoring and even execution of functions on detection of events.

9.8.1 Integration Overview

System Monitor is designed to enhance the capabilities of local SNMP agents as well as to provide distributed management of SNMP information. The former capability is of special interest in terms of potential integration with ADSM. The system information agent MIB can be extended to include ADSM-specific information which can be used to monitor and react to ADSM events.

This process would involve customization of the system information agent which can be effected by updating the command table. This procedure is similar in effect to the customization of NetView to react to notifications forwarded from ADSM as described in the NetView scenario in 9.2.2, “Scenario” on page 85.

9.8.2 Scenario

It would be possible to implement much of the function documented in the NetView scenario using the capabilities of System Monitor. Certainly, the system information agent command table could be customized to retrieve ADSM console messages and to forward relevant messages on to NetView as SNMP traps. Local execution of functions in reaction to some ADSM messages is also possible.

However, due to the greater sophistication of NetView’s trap processing capability, most message analysis and processing is best implemented at NetView, particularly in the case of multiple message processing. System Monitor could also act as a filter rather than the glue code used in the NetView scenario.

The focus of this book is primarily on integrating ADSM, and as System Monitor provides an alternate mechanism for effecting part of the management scenario described for NetView, a further scenario is not included here.

Chapter 10. NetView Scenarios

This chapter illustrates the potential integration of ADSM and NetView through the use of some example scenarios.

The first section describes some of the problems that may be encountered when designing an integration methodology and some of the techniques that may be used to circumvent them.

10.1 Processing ADSM Messages

The difficulties of integration center on the differing needs of the ADSM user and how these conflict with the needs of the administrator. The ADSM user will of course articulate a need to have an efficient, responsive and fast backup, archive and retrieval system that is available 100 percent of the time and which is transparent. The systems administrator will share the desire for efficiency, speed and responsiveness, but will be looking for certain other features in the integrated solution.

In the network environment, the administrator will be aware that there are constrained resources available to investigate the behavior of applications. It will therefore be necessary to define a solution that highlights exceptions quickly and provides an appropriate level of focus.

10.1.1 Stages of Notification

At present, the essence of integrating ADSM and SystemView is to generate messages within ADSM and to convert these into notifications which are forwarded to NetView. These notifications are then acted upon automatically by SystemView features or are brought to the attention of the administrator to be actioned.

In order to use an appropriate level of resource, the events which are notified to the network administrator should be serious enough to warrant his attention, sufficient in number or visual impact to convey quickly their importance and few enough in number to avoid swamping him with information. In order to achieve this, it is important to consider carefully each stage of implementation. In fact, the process of choosing which messages are to be trapped and forwarded to SystemView can be regarded as applying the first filter.

The first stage of a notification is determining the event set that must be trapped to ensure viable running. In determining this set the administrator is applying the first filter; he determines which messages are likely to be of interest and discards the rest. The chosen messages are then ordered into a hierarchy of importance along the lines of:

- Those messages for which some background notification of the event is required for logging purposes

  For example, the mounting of a particular tape or the messages produced during (but not those produced at the end of) database auditing. These messages are the lowest grade of notifiable messages, and their presence at the network control point may be taken to mean that the application is functioning correctly and that business is continuing as usual. Such messages will not be referred to in the normal course of events by the

administrator. They may, however, be of use in initial problem determination when a problem is discovered.

- Messages for which notification to the administrator is required immediately, for example the completion of a database audit

  The content of these messages will be addressed by the administrator quickly in the normal course of events.

- Messages for which urgent corrective action is required

  For example, if the server runs out of database space during normal operation then, correction is a matter of some importance. These events may occur outside normal business hours and yet still require a quick response. It is appropriate to define actions to be taken by SystemView automatically on receipt of such notifications.

10.1.2 Gathering Messages

Having determined which ADSM messages are to be trapped by the SNMP agent process, the method of packaging that information into traps and transmitting to NetView needs to be considered.

In the trivial case, the agent merely places the text of the ADSM message into the appropriate field of a trap using the /usr/OV/bin/snmptrap command. However, this is unlikely to be satisfactory for any but the smallest implementation. Alternatives that may be considered include:

1. Using a different specific trap-ID for each ADSM message

   This is the approach used when messages are recorded in the AIX error log and alerted by the NetView SNMP daemon.

2. Using a unique trap-ID for a range of ADSM messages

3. Allocating certain specific trap-IDs to be used by the agents of a particular host

   This approach is not considered worthwhile as NetView will report the node that it received a trap from in any case.

It is also possible to code a level of filtering into the message- gathering routines. This was of greater value before the Event Stream Enhancements of NetView V4, but could still be used today. This involves writing glue code that recognizes the significance of clusters of messages and notifies NetView via a ′pseudo-message′ describing, in one trap, the ADSM situation.

The availability of Event Stream Enhancements now simplifies the recognition of clusters if messages and places all functions related to that into NetView. Thus, the rules of cluster definition are created and applied at the NetView host rather than having disparate code that must be maintained out at each client.

Once the message-gathering routines have been created, they may be distributed and maintained using NetView Distribution Manager.

10.1.3 Sample Messages, Priorities and SystemView Responses

When the system administrator categorizes the ADSM messages as described above, he may note that many are of little importance in actually managing the system. For example, the message: `ANR0402I Session started for administrator` is usually of no significance to actually managing ADSM. It is generated every time an administrator starts an ADSM session; but the aim of integrating ADSM and SystemView is not to provide audit trails of administrator usage, but to do useful work from a centralized control point. Therefore this message is NOT selected for forwarding to SystemView.

Table 1 lists a number of messages which would generally require immediate action by the system administrator; messages of this type are the most obvious candidates for automation.

Table 1 (Page 1 of 2). ADSM Messages Requiring Immediate Action by Administrator

ADSM Message Number	Reason Message is Generated
ANR0130E	Server log space is exhausted
ANR0131E	Server database space is exhausted
ANR0132E	Server memory allocation failed
ANR0202W	Database volume varied off due to read errors
ANR0204W	Recovery log volume is varied off
ANR0205W	Recovery log volume is varied off
ANR0206W	Page write error detected on database
ANR0207E	Database page address mismatch
ANR0208W	Log volume partial write
ANR0209E	Log page address mismatch
ANR0210S	LVM internal error
ANR0211S	LVM page address mismatch
ANR7841S	Kernel memory shortage
ANR7842S	Monitor kernel extension not initialized
ANR7851S	Signal handler error
ANR7823E	Server console out of memory
ANR7836S	Server initialization terminated
ANR7838S	Server operation ended
ANR2700E	Schedule manager aborted
ANR2707E	Scheduler out of log space
ANR2708E	Scheduler out of database space
ANR0212E	Unable to read disk definition file
ANR0214W	Database volume is offline
ANR0215W	Log volume is offline
ANR0358E	Database start-up failed due to memory
ANR0359E	Database start-up failed due to restart record
ANR0360E	Database start-up failed due to database size error
ANR0361E	Database start-up failed due to page allocation

Table 1 (Page 2 of 2). ADSM Messages Requiring Immediate Action by Administrator

ADSM Message Number	Reason Message is Generated
ANR0437W	Server error - licensing
ANR0438W	Session denied - licensing
ANR0439W	Backup denied - licensing
ANR0486W	Session killed - internal error
ANR0246E	Server error reading database
ANR0249E	Server error reading log
ANR0252E	Server error writing database
ANR0254E	Server error writing log
ANR0256E	Server error writing checkpoint to database
ANR0257E	Server error writing checkpoint to log
ANR0258E	Server error writing checkpoint to database or log
ANR0485W	Session killed - low memory
ANR0521W	Transaction failed due to object size
ANR8214W	Session failed - connection refused
ANR2716E	Server cannot execute scheduled event
ANR2717E	Server cannot contact scheduler client
ANR2574W	Scheduler out of log space
ANR2575W	Scheduler out of database space
ANR2572W	Schedule prompter denied
ANR2576W	Server failed to update a past event
ANR0420W	Session refused - access disabled
ANR2708E	Scheduler out of database space
ANR9728E	Error ejecting volume from drive
ANR0664E	Media not accessible
ANR0710E	Server unable to start process

The following table lists those messages which would not normally require immediate attention.

Table 2 (Page 1 of 2). ADSM Business-as-Usual Messages

ADSM Message Number	Abbreviated Message Text
ANR8320I	Insert volume in drive of library
ANR8326I	Mount requested tape volume in xx minutes
ANR0522W	Transaction failed for session
ANR8328I	Volume mounted in drive
ANR8335I	Verifying label of volume
ANR8330I	Volume mounted - in use
ANR8329I	Volume mounted, status idle
ANR8334I	Number of volumes found

Table 2 (Page 2 of 2). ADSM Business-as-Usual Messages

ADSM Message Number	Abbreviated Message Text
ANR0400I	Session started for node
ANR0403I	Session ended for node
ANR0480W	Client session severed
ANR0481W	Client session timed out
ANR0659W	Server unable to obtain node password
ANR0350I	Recovery checkpoint started
ANR0351I	Recovery checkpoint completed
ANR0352I	Transaction recovery completed
ANR0617I	Import or export process completed OK
ANR0990I	Restart/recovery in progress
ANR0991I	Server shutdown complete
ANR0993I	Server initialization complete
ANR4000I	Database dump process started
ANR4001I	Database dump process complete
ANR0421W	Session refused - signon violation
ANR0422W	Session refused - invalid node
ANR0423W	Session refused - invalid administration ID
ANR0424W	Session refused - invalid password provided
ANR0425W	Session refused - password has expired
ANR0426W	Session refused - no open registration
ANR0429W	Session refused - maxsess exceeded
ANR0430W	Session refused - node locked
ANR0432W	Session refused - short of memory
ANR0435W	Session refused - internal error
ANR2560I	Schedule manager started
ANR2561I	Scheduler contacting node

10.1.4 Taking Action on Receipt of a Message

It is quite straightforward to put together a list of messages which must be detected in an ADSM installation. It is largely a matter of studying the messages that ADSM produces, understanding their significance and occasionally revisiting the list produced. Once the list is made, it is a trivial task to write a simple SNMP agent to detect those messages and forward them to the controlling SystemView. It should be noted that when the agent checks the content of messages, care should be taken to ensure currency of the messages being checked for; the text of ADSM messages has been known to change between releases.

Having notified the administrator, the next point to consider is which aspects of the system operation can be automated and the response appropriate to each message. The response appropriate to the receipt of a message may be one of the following:

- Ignore the message

  Some messages indicate only some minor aspect of successful operation of the server. It is appropriate therefore to ignore these individual messages.

- Analyze the Message sequence

  The NetView V4 Ruleset editor provides powerful functions to allow responses to clusters of messages to be defined. This feature is of tremendous value in managing applications that generate traps which have little individual significance but which collectively provide necessary problem determination information. For example, the following message: `ANR7838S Server operation terminated - internal error detected` usually appears sandwiched between other messages that give information about the reason for the failure.

  A good implementation of automation will recognize the significance of clusters and define rules for dealing with them.

- Record the event for future reference

  For example, the following message: `ANR0402I Session started for administrator` might be recorded for future reference. It indicates that an administrator successfully started an administrative client session with the server after supplying the correct ADSM ID and password. Therefore, there is no great significance in the event; it indicates no error and requires no action.

  Further, it is recorded in the activity log on the relevant server, which is the log which will be examined early in the problem determination procedures anyway. Hence, this message would not, in most installations, be forwarded to the SystemView control point.

- Record the event and notify some specialized personnel for future action

  This situation may come about where the administrator decides that an ADSM message is likely to be generated at a time when it is inadvisable for immediate action to be carried out.

  The nature of any backup product is such that it will perform most of its work during the night or at the weekend when there are fewer skilled staff available. Certain ADSM messages, while occurring rarely, will indicate a need for investigation of system parameters that may affect other users, or need careful consideration due to potential impact beyond ADSM. An example would be: `ANR0529W Transaction failed due to insufficient memory.`

  This message is generated when the server detects a short-of-memory situation and ends a client session. Lack of memory may indicate that the parent operating system is short of paging space, a situation that most administrators would not allow to occur from choice. Therefore, the response to this message may be to send a notification to the performance specialist and the administrator via e-mail and to log the occurrence.

- Record the event and bring it to the attention of the system administrator immediately

  This is appropriate when the message indicates a situation which is not recoverable automatically or for which an attempt at automatic recovery is not advisable. Take, for instance, the following messages:

  – ANR9999D
  – ANR0259E

ANR9999D is generated in response to disastrous situations, such as when the server is unable to find any database when it starts. ANR0259E occurs when the server is unable to find its checkpoint data at initialization.

In either case, it is unlikely that the required corrective action can be planned to such a depth and accuracy as to be able to correct the problem without human intervention. No automatically invoked procedure could actually correct the situations referred to by these messages where the cause was hardware failure.

- Record the event and obtain further information regarding it before presenting the whole to the administrator for action

  Where a message indicates a situation that required further investigation by the system administrator, it may be possible to obtain further information from the ADSM activity log to present a more coherent picture. For example, when the message: `ANR0530W Transaction failed due to server internal error` is generated, server operation continues. This enables the first stage of problem determination to be carried out automatically, before presenting the output for investigation.

- Record the event and perform some remedial action immediately

  This response is probably the most attractive aspect of integrating SystemView and ADSM. This enables the SystemView central control point to respond immediately to problems experienced by ADSM. The mechanism used is to configure NetView to carry out an action on receipt of a particular SNMP trap. That action is in the form of a executable file which communicates with the ADSM server that generated the original message to fix the problem. Once the action has been carried out, it is important that the network control point is informed to assist in problem determination if automated attempts fail.

Further examples of each of these types of response are given in more practical detail later in this publication.

10.1.5 Examples of Automatic Responses to ADSM Messages

Examples of actually defining automatic responses will be given later in this redbook. Some of the typical responses that may be appropriate for individual messages that ADSM produces are shown in Table 3:

Table 3 (Page 1 of 2). ADSM Messages and Sample Automatic Actions

ADSM Message Number	Reason for Message	Possible Automatic Action
ANR0210S	LVM internal error	Notify administrator immediately
ANR0211S	LVM page address mismatch	Notify administrator immediately
ANR7841S	Kernel memory shortage	Notify administrator immediately
ANR7842S	Kernel extension not initialized	Load the client kernel extension
ANR7836S	Server initialization terminated	Notify administrator immediately
ANR7838S	Server operation ended	Log only
ANR2700E	Schedule manager aborted	Obtain more information and notify the administrator
ANR2707E	Scheduler out of log space	Extend log space on the server
ANR2708E	Scheduler out of database space	Extend server database space
ANR0212E	Unable to read disk definition file	Notify administrator immediately

Table 3 (Page 2 of 2). ADSM Messages and Sample Automatic Actions

ADSM Message Number	Reason for Message	Possible Automatic Action
ANR0214W	Database volume is offline	Issue VARY ON for volume
ANR0215W	Log volume is offline	Issue VARY ON for volume
ANR0358E	Database start-up failed due to memory	Notify relevant specialist
ANR0359E	Database start-up failed due to restart record	Notify administrator immediately and advise contacting IBM
ANR0360E	Database start-up failed due to database size error	Notify administrator and advise contacting IBM
ANR0361E	Database start-up failed due to page allocation	Notify administrator and advise contacting IBM
ANR0437W	Server error - licensing	Audit licenses on the server
ANR0438W	Session denied - licensing	Audit licenses on the server
ANR0439W	Backup denied - licensing	Audit licenses on the server
ANR0486W	Session killed - internal error	Notify administrator immediately
ANR0246E	Server error reading database	Switch to mirror and repair
ANR0249E	Server error reading log	Use mirror and repair
ANR0252E	Server error writing database	Use mirror and repair
ANR0254E	Server error writing log	Use mirror and repair
ANR0256E	Server error writing checkpoint to database	Use mirror and repair
ANR0257E	Server error writing checkpoint to log	Use mirror and repair
ANR0258E	Server error writing checkpoint to database or log	Take emergency action to repair bad volume(s)
ANR0485W	Session killed - low memory	Send mail to specialist to get paging sorted out
ANR0521W	Transaction failed due to object size	Warn administrator of large file and turn compression on
ANR8214W	Session failed - connection refused	Check TCP/IP and scheduler daemon on client
ANR2716E	Server cannot execute scheduled event	Check TCP/IP and scheduler
ANR2717E	Server cannot contact scheduler client	Check TCP/IP and scheduler
ANR2574W	Scheduler out of log space	Extend the log
ANR2575W	Scheduler out of database space	Extend the database
ANR2572W	Schedule prompter denied	Set maxschedsessions properly
ANR2576W	Server failed to update a past event	Check if more than 1 client scheduler running at client node
ANR0420W	Session refused - access disabled	Enable the server
ANR2708E	Scheduler out of database space	Allocate more space
ANR9728E	Error ejecting volume from drive	Notify system operator
ANR0664E	Media not accessible	DEFINE VOLUME required
ANR0710E	Server unable to start process	Allocate Memory

10.1.6 How to Define Responses to ADSM Messages

The required response to a particular ADSM message varies from one installation to another. In large installations with hundreds of ADSM clients being backed up over a highly developed and automated network, the level of automatic response required will usually be higher than that for small installations where the administrative function may be less thinly spread.

In planning the configuration of SystemView components to manage ADSM events, one should consider the following factors:

- Does the message indicate that the server has failed?
- Does the server terminate when this message is generated?
- Is the response to this message the same regardless of the time of day?
- Which server functions are critical to viable operation?
- How much resource is available to configure ADSM support?
- Is automatic response required to each message?
- What level of notification is appropriate to each message?
- How often will any corrective action be run before handing control to the system administrator?
- How will the number of times a corrective action has been tried be monitored?
- Is this message produced as part of a cluster of messages, another of which is more significant than this particular message?

10.2 Generating Notifications from ADSM Console Messages

It is easy to intercept the console messages of ADSM. Messages that ADSM writes to its console (SERVER_CONSOLE) can be made available to shell scripts by starting a console mode administrative client session and piping the output into a shell script. That script then issues a relevant snmptrap command (located in /usr/OV/bin) to notify NetView of the message.

In principle, the following code will notify NetView of any ADSM console message:

```
#!/bin/ksh
# Simplest sample program to notify NetView of an ADSM message.
#
# Loops reading standard input, generating calling snmptrap to pass the
# message on to NetView. The same specific trap-id is used
# for every message.
#

# Set to the address of NetView host before execution.
NetView=

while true
do
  read adsm_msg
  /usr/OV/bin/snmptrap $NetView .1.3.6.1.4.1 hostname 6 2 0 \
               .1.3.6.1.4.1 octetstringascii "$adsm_msg"
done
```

This script, in the file /usr/local/adsm/notify_all in this example, is invoked so as to read its standard input from the standard output of an ADSM administrative client in console mode. Therefore, where the administrator's name is ADMIN and password ADMINPW, the notification script is invoked as follows:

```
dsmadmc -console -id=ADMIN -pa=ADMINPW | /usr/local/adsm/notify_all &
```

Alternatively, some administrators who prefer to maintain a history of server activity may prefer to use two processes to perform the same ultimate function. These would be:

1. The administrative client process in console mode to write to a file using the OUTFILE parameter.
2. A tail command process to continuously take the messages written to the file and process them into traps.

While some administrators have preferred this approach, there seems little to be gained from holding the ADSM messages in another file on the server system.

10.3 Configuring NetView to Manage ADSM

When integrating ADSM and SystemView, the most fundamental need is to allow ADSM to communicate with NetView. This can be done using quite straightforward shell scripts, as described elsewhere in this book.

10.3.1 NetView Tasks to Support ADSM Messages

When supporting ADSM, NetView provides the focal point for control. It is important to maintain the flow of information to the center, and to filter data wherever possible to produce valuable information. NetView features provide strong mechanisms to support that process.

The fundamental need of ADSM integration is that NetView should recognize the information being provided by glue code around ADSM. ADSM data will be transmitted across the network in SNMP trap format. The type of trap may vary according to the designs of the system administrator: only one trap may be used, or many. It is most likely that the more advanced or complex an installation, the greater the number of traps that will be employed in supporting ADSM. Each of those traps must be recognized by NetView.

10.3.1.1 Configuring Netview to Recognize ADSM Traps

To configure a NetView event corresponding to the trap to be received from the ADSM server, the following steps should be carried out.

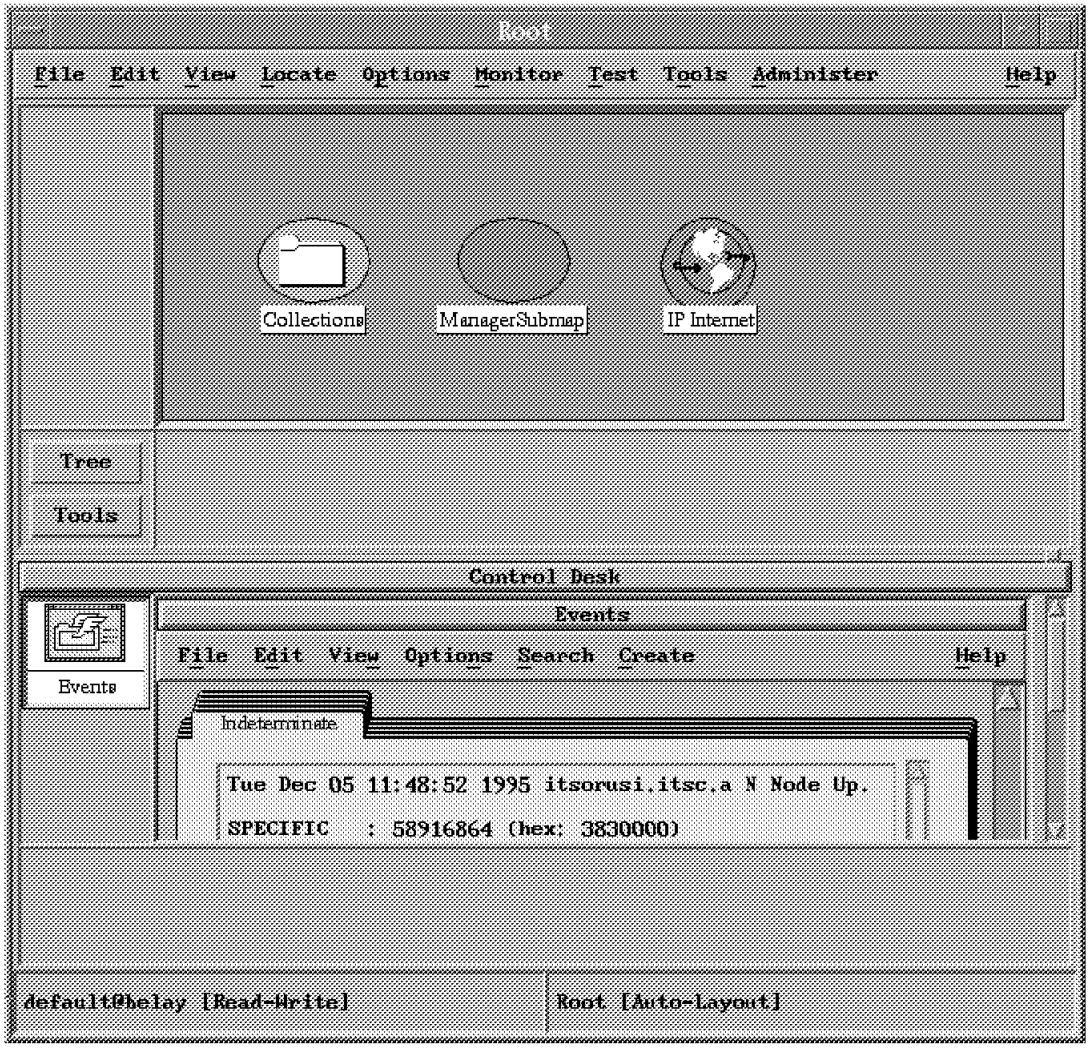


Figure 1. Selecting Trap Customization from the NetView Root Window

From the NetView root map, choose **Options**, then **Event Configuration** and **Trap Customization** (shown in Figure 1). Select the enterprise that corresponds to the event that is to be configured; in this case, an enterprise called ADSM has been defined as 1.3.6.1.4.1.51, and this has been selected. Then press the **Add...** button alongside the Event Identification list.

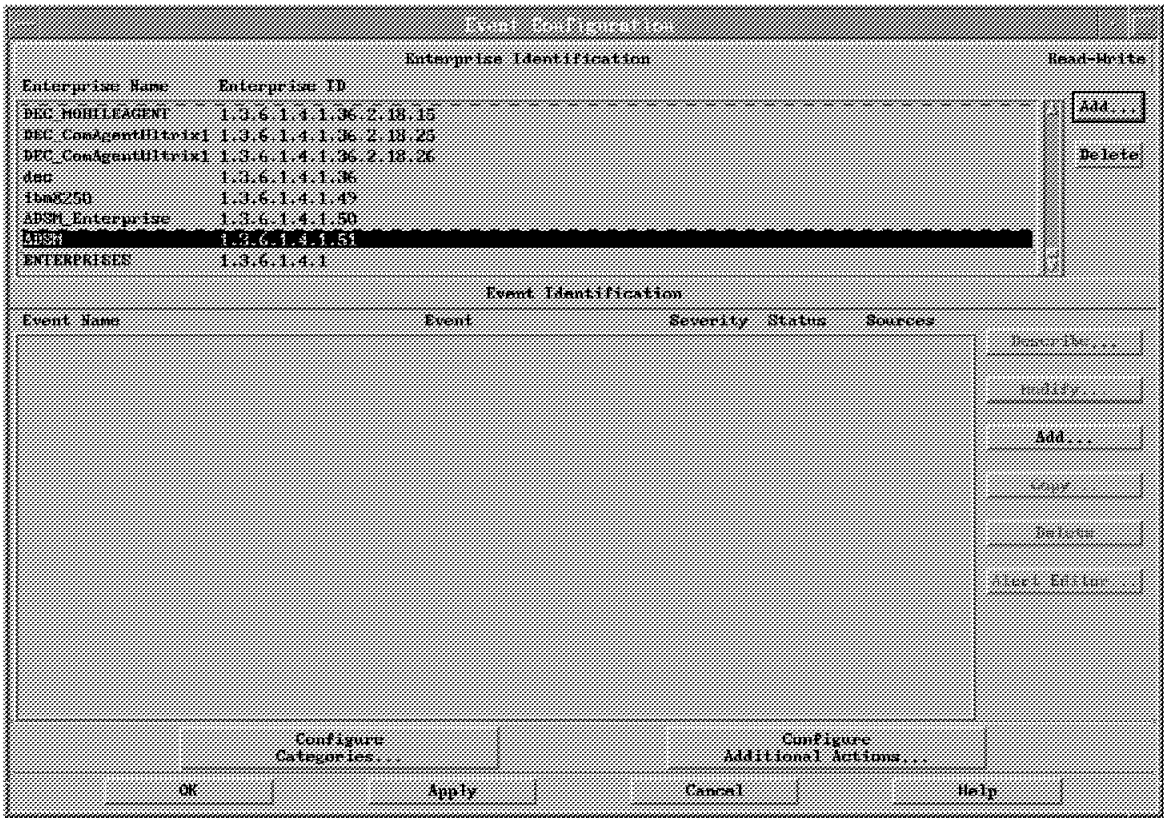


Figure 2. Event Configuration Window

This will produce the Add Event window, as shown in Figure 3 on page 105. To continue the example from above, an event may be added that corresponds to the trap generated by the shell script:

Event Name The event name field must be a unique name which NetView will use to identify the event to be configured; in the illustration, it is set to adsm_all.

Generic Trap The generic trap numbers are defined by SNMP. For the purposes of these examples, it will assumed that Enterprise-specific generic trap IDs are to be used throughout. When using enterprise-specific trap IDs, the specific trap number field must be filled in.

Event Sources Event sources refer to the nodes in the network which may send this specific trap to NetView. They may be defined by typing in the Source input field or by clicking **Add From map** and choosing nodes from a NetView graphical map display.

In the example, `belay.itsc.austin.ibm.com` is added.

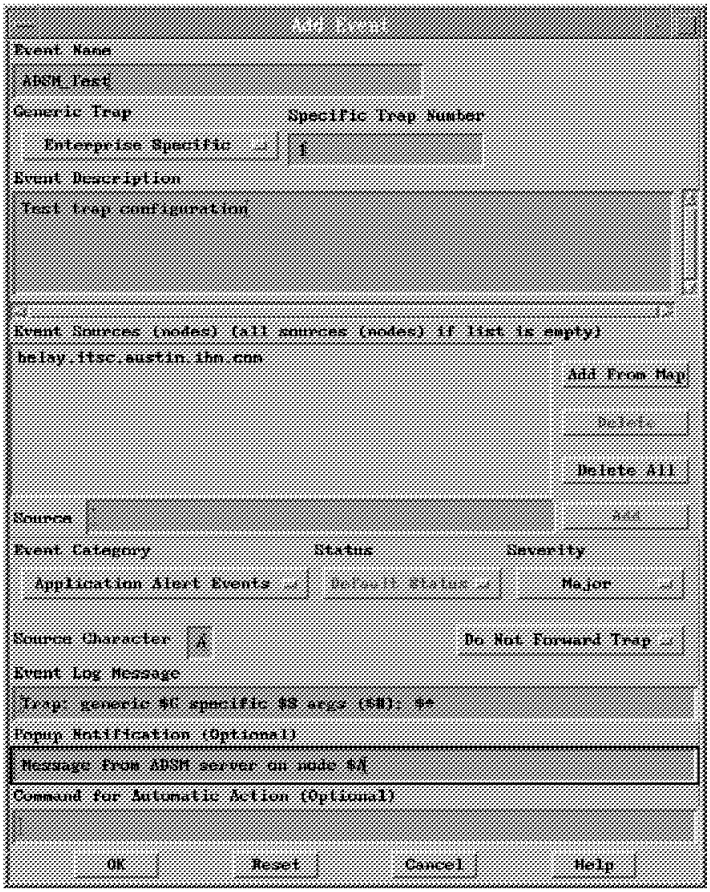


Figure 3. Adding a NetView Event

Event Category Used to select the types of events that will be considered.

Status Used to select the status of events considered.

Severity Used to select the severity of events considered.

Forward Trap The forward trap field is of use only when the trapd daemon has been configured to forward traps on to another network management station.

Event Log Message The event log message defines the text and variable for the message written to the NetView event log. The standard C language formatted output characters are recognized, plus flags to access the fields of the incoming trap. In the illustration, $G displays the generic trap, $S the specific trap, $# the number of variables in the trap, and $* displays all the variables supplied by the trap.

Popup Notification The Popup Notification field contains the text to be displayed in a pop-up window which will be displayed to the operator when this trap is received. This field may also use the variables transmitted in the trap.

Automatic Action The command for automatic action field is the full pathname of a command to be executed on the controlling NetView node in response to the received trap. This is usually an executable which can notify the operator or take some form

of recovery action. This feature will be discussed in greater detail later in this publication.

Once the NetView event has been configured, the event is updated on the Event Configuration Dialog, as shown in Figure 4.

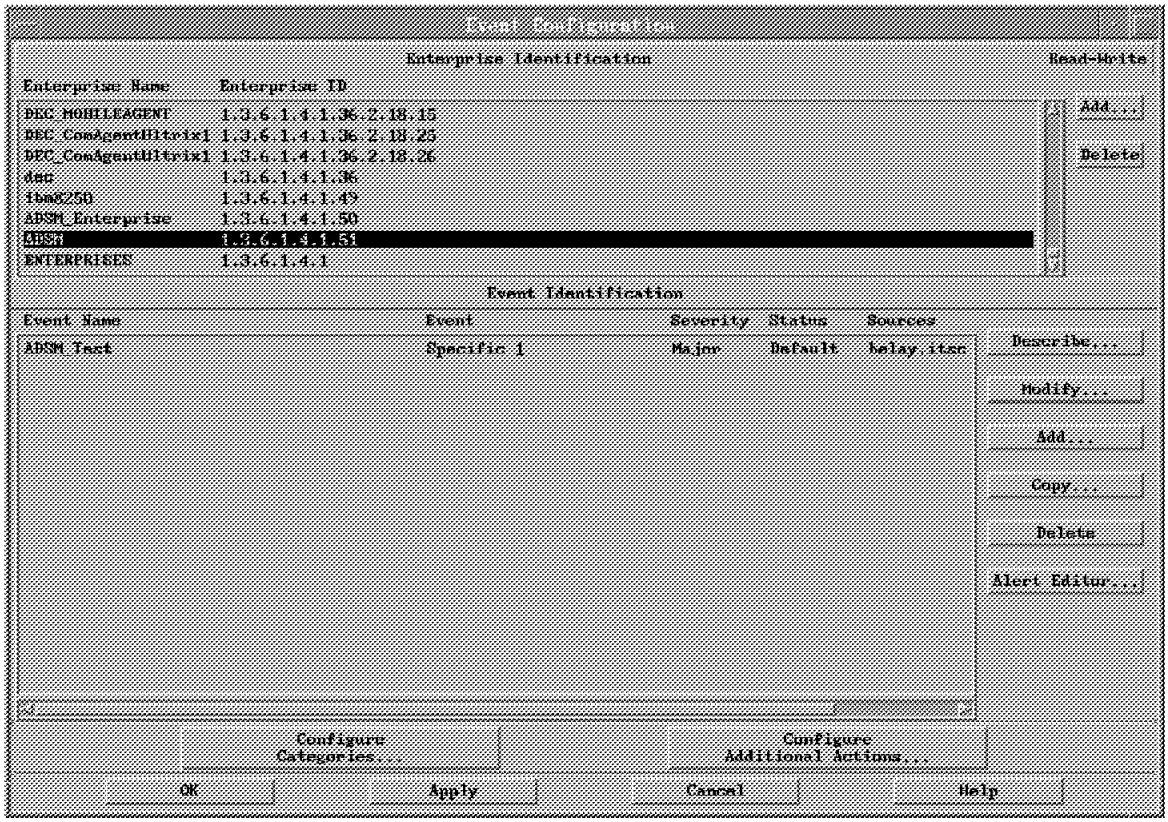


Figure 4. Updated NetView Event Configuration Window

10.3.1.2 Resulting NetView Displays

If the above example is followed, the result is a very simple tool that simply forwards all ADSM messages to NetView. This illustrates both the simplicity and elegance of the relationship and its weaknesses.

First, the beauty of the relationship between ADSM and NetView is that it can be quickly implemented and brought to a useful state. The passing of meaningful messages to a central point of network control is an essential feature of open systems distributed management. Here, the messages are passed, interpreted and brought to the attention of the network administrator by intuitive use of the menu interface and a trivial shell script.

The problem, however, lies not in what is passed to the control point, but the amount of information. If this example is tested out on a system, it will be apparent that most administrators will not be interested in all of the messages generated by ADSM. The first messages passed to NetView by the example will be the administrative client initiation messages caused by running it in the first place. This means that five messages are forwarded to NetView on starting the script. The cumulative meaning of these five messages is that one administrative

client session has been started. Each of these messages causes the network administrator to see a pop-up warning window appear on the screen.

The importance of the event being notified is not always in proportion to the importance of the event; both the administrator and the display station will be swamped with information which may or may not be of importance.

Therefore, attention must be given to defining the messages and actions to be implemented on any running system. For more information on this subject, see 10.1, “Processing ADSM Messages” on page 93.

10.3.1.3 Further NetView Customization

Now that a basic level of integration has been implemented, there are many enhancements that can be performed. Some examples include:

- Filtering console messages to produce traps
- Using color and window highlighting and beeps to get operator attention
- Using the SNMP agent to generate SNMP traps from the AIX error log
- Using the interface to Trouble Ticket

Practical examples of some of these ideas will be presented in the scenarios (10.4, “Automatically Restarting a Terminated Server” on page 109 and 10.5, “Processing ADSM Tape Mount Messages” on page 116).

10.3.2 Using the Event Stream Enhancements of NetView V4

Event Stream Enhancements (ESE) are a package of functions provided in NetView Version 4 to enhance the event management features of NetView. ESE gives the management application the following functions:

- Correlation of events

  That is, the ability to define rules to recognize that a sequence of events from a particular source may be of greater significance than the content of any single trap conveys.

- Override ability

  This enables severity and node status to be modified dynamically.

- Access to information outside the event stream

  For example, the NetView object database.

- Enhanced event processing

  The ability to access and trigger on any of the information carried by an event.

More information on the correlation of events and the ability to trigger on the information carried within an event is provided in the examples below.

10.3.3 Automation Examples

The examples given in this chapter were used in a configuration of four RS/6000s running AIX, where:

- All machines had ADSM clients installed.
- One machine had NetView for AIX server.
- All machines had NetView for AIX client.

- Two machines had ADSM for AIX servers installed.

Table Table 4 lists the messages which have been automated in this example. It is a subset of Table 1 on page 95.

Table 4 (Page 1 of 2). ADSM Messages and Sample Automatic Actions

ADSM Message Number	Reason for Message	Automated Action
ANR7838S	Server operation terminated - internal error	Notify administrator, attempt recovery
ANR7833S	Server operation aborted	Notify administrator immediately.
ANR0210S	LVM internal error	Notify administrator immediately
ANR7842S	Kernel extension not initialized	Load the client kernel extension
ANR7836S	Server initialization terminated	Notify administrator immediately
ANR7838S	Server operation ended	Log only
ANR2700E	Schedule manager aborted	Obtain more information and notify the administrator
ANR2707E	Scheduler out of log space	Extend log space on the server
ANR2708E	Scheduler out of database space	Extend server database space
ANR0212E	Unable to read disk definition file	Notify administrator immediately
ANR0214W	Database volume is offline	Issue VARY ON for volume
ANR0215W	Log volume is offline	Issue VARY ON for volume
ANR0437W	Server error - licensing	Audit licenses on the server and notify administrator of outcome
ANR0246E	Server error reading database	Switch to using mirror copy and repair failing volume
ANR0257E	Server error writing checkpoint to log	Use mirror and repair
ANR0258E	Server error writing checkpoint to database or log	Take emergency action to repair bad volume(s)
ANR0485W	Session killed - low memory	Send mail to specialist to get paging sorted out
ANR0521W	Transaction failed due to object size	Warn administrator of large file and turn compression on.
ANR8214W	Scheduler session failed - connection refused	Check TCP/IP and scheduler daemon on client
ANR2716E	Server cannot execute scheduled event	Check TCP/IP and scheduler
ANR2717E	Server cannot contact scheduler client	Check TCP/IP and scheduler
ANR2574W	Scheduler out of log space	Extend the log
ANR2575W	Scheduler out of database space	Extend the database
ANR2572W	Schedule prompter denied	Set maxschedsessions properly
ANR2576W	Server failed to update a past event	Check if more than 1 client scheduler running at client node
ANR0420W	Session refused - access disabled	Enable the server
ANR2708E	Scheduler out of database space	Allocate more database space
ANR9728E	Error ejecting volume from drive	Notify system operator

Table 4 (Page 2 of 2). ADSM Messages and Sample Automatic Actions		
ADSM Message Number	**Reason for Message**	**Automated Action**
ANR0664E	Media not accessible	DEFINE VOLUME required

10.4 Automatically Restarting a Terminated Server

It is possible to integrate ADSM and NetView such that where an ADSM server has been accidentally halted by an administrator, it is restarted without intervention. An example of how to implement this is given in this section.

This is of some interest due to the known difficulties that some users have had in starting the ADSM server in the background from a program.

The example requires the following:

- Customization of the NetView configuration file, trapd.conf, via the NetView GUI
- A shell script executed on the NetView host under the control of NetView
- A C language executable which resident on the ADSM server host to start the server
- A shell script resident on the ADSM server host which monitors ADSM messages

These items are summarized in Table 5:

Table 5. Components Used in Automating Restart of the ADSM Server		
Name	**Machine**	**Function**
/usr/local/trap_gen	ADSM Server	Shell script which monitors ADSM messages through administrative client
/usr/local/start_adsm_c	ADSM Server	C language executable that restarts the server in the background
/usr/local/start_adsm	NetView Server	Shell script called directly by NetView in response to the trap notifying of the termination of the ADSM server

10.4.1 Method

The ADSM server is established in the background in the normal way. In order to monitor the messages produced, the administrative client is started in console mode, and the output piped to a shell script which scans the messages. When messages signifying the termination of the server are recognized, a trap is sent to the NetView server using the `snmptrap` command.

NetView recognizes the trap and automatically actions the `/usr/local/start_adsm` script. The script uses `rsh` to remotely execute the `/usr/local/start_adsm_c` on the ADSM server, which re-establishes the server and the message monitoring through the administrative client.

On first inspection, it might be expected that the message-scanning routine would be written to recognize the message ANR0991I, the server termination message. However, that message will never be seen by an administrative client in the normal course of events; the ADSM processes handling the communication with the client will terminate before the message is produced.

Hence, using this approach, the message recognized and translated into a trap is ANS5103I with a return code of -50, which signifies the loss of the TCP/IP connection to the server. It is recognized that for completeness, a full application would take other actions on receipt of this message. For example, checking to see that ADSM processes have in fact terminated or trying to start another administrative client before generating the SNMP trap. The aim here is to show the way with some sample code fragments, rather than to build a fully-integrated system.

10.4.2 Sample Code

The following sample code fragments were used to produce the displays illustrated in this section.

10.4.2.1 NetView Host: start_adsm Script

The script /usr/local/start_adsm used in this example was:

```
#!/bin/ksh
# This shell script starts an ADSM server on a remote system.
# Parameter: $1 is the nodename of the remote system on which
#               the ADSM server is to be started.
#
# /usr/local/adsm/start_adsm_c is a C executable on the target system
#

rsh $1 '/usr/local/adsm/start_adsm_c &'
```

Note that in order for the rsh command to actually have the authority to execute on the remote host, a .rhosts file must be created in the root directory at the remote host, giving the NetView server root user permission to execute commands.

10.4.2.2 ADSM Server Host: start_adsm_c Executable

The file /usr/local/adsm/start_adsm_c used at the ADSM server host in this example is as follows:

```
#define ERROR   -1
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
/*
   Program to start ADSM Server on AIX .
   This program called remotely to start server after it has failed.

   Original Author: Mike Skidmore, IBM
   Date  : October 1995.
*/
int retcode, rc, sig ;
pid_t pid;
char nohup_cmd[81] =
     "/usr/bin/nohup /usr/lpp/adsmserv/bin/rc.adsmserv & ";
char monitor_cmd[181] =
```

```
     "/usr/bin/nohup /usr/bin/dsmadmc -id=admin -pa=admin -console |
/usr/local/adsm/trap_gen & ";

void trap();

main(int argc, char *argv[])
{

  /* Ensure running as root       */
  retcode=setuid(0);

  /* Start of netlsd ADSM server  */
  fflush(stdout);
  if ((pid = fork()) == 0)
  {
     /* Start child process......
        DETACH this process from ALL
        user-generated interrupts
        Disconnect from control term */
     setpgrp();

     /* Now disable all signals      */
     for (sig = 0; sig < NSIG; sig++) signal(sig, trap);

     /* Now close std-files          */
     fclose(stdin);
     fclose(stdout);
     fclose(stderr);

     /* Now call target routine in   */
     /* a subshell.............      */
     rc = execl("/usr/bin/ksh", "sh", "-c", nohup_cmd, NULL);

  }
  wait(&rc);
  /* Calculate return code        */
  rc = ((rc>>8) & 0xff);

  /* Start Monitor process again  */
  fflush(stdout);
  if ((pid = fork()) == 0)
  {

    /* Start child process......
       DETACH this process from ALL
       user-generated interrupts
       Disconnect from control term */
    setpgrp();

    /* Now disable all signals      */
    for (sig = 0; sig < NSIG; sig++) signal(sig, trap);

    /* Now close std-files          */
    fclose(stdin);
    fclose(stdout);
    fclose(stderr);

    /* Wait for server to get going */
    sleep(10);
```

```
    /* Now call target routine in
    a subshell.............      */
    rc = execl("/usr/bin/ksh", "sh", "-c", monitor_cmd, NULL);

  }
  wait(&rc);

  /* Calculate return code        */
  rc = ((rc>>8) & 0xff);
  return(rc);
}
```

Note that the program child processes perform the usual detaches and signal handling for a daemon prior to starting the ADSM server and message monitor programs. If this is not done, the ADSM server will send messages to the calling process (even through rsh) and lock it up.

10.4.2.3 ADSM Server Host: trap_gen Script

The script file /usr/local/adsm/trap_gen used on the ADSM server in this example is as follows:

```
#!/bin/ksh

NVhost="gothmog"
trap "exit" ERR
wall "Monitor Process Starting"
while true
do
  read input
  echo $input | cut -d " " -f1 | read msg_no

  # Server shutdown complete
  if [ "$msg_no" = "ANR0991I" ]
  then
    /usr/OV/bin/snmptrap $NVhost .1.3.6.1.4.1.50 hostname 6 1 0 \
                         .1.3.6.1.4.1.50 octetstringascii "$input"
  elif [ "$msg_no" = "ANS5103I" ]
  then
    retcode=$(echo $input | cut -d " " -f6)
    if [ "$retcode" -eq "-50." ]
    then
      /usr/OV/bin/snmptrap $NVhost .1.3.6.1.4.1.50 hostname 6 1 0 \
                           .1.3.6.1.4.1.50 octetstringascii "$input"
    fi
  fi
done
```

10.4.3 NetView Configuration to Support Restart of a Failed ADSM Server

To implement ADSM integration, a number of NetView tasks must be performed:

- Define a new ADSM enterprise
- Add the events necessary to support the function
- Customize those events

All of these tasks can be carried out through the NetView GUI. To access the trap customization functions, use the Options pull-down menu, choosing **Event**

Configuration, then **Trap Customization SNMP**, as shown in Figure 1 on page 103.

The event configuration window shown in Figure 1 on page 103 will be displayed.

Click the **Add** button in the top half of the window. The Add New Enterprise window, as shown in Figure 5, will open, prompting for a new enterprise ID and name.

In the example routines used in this book the enterprise name used was ADSM_Enterprise, with an enterprise ID of 1.3.6.1.4.1.50. While the particular ID and name are not important, conflicts with other names and/or IDs must be avoided.

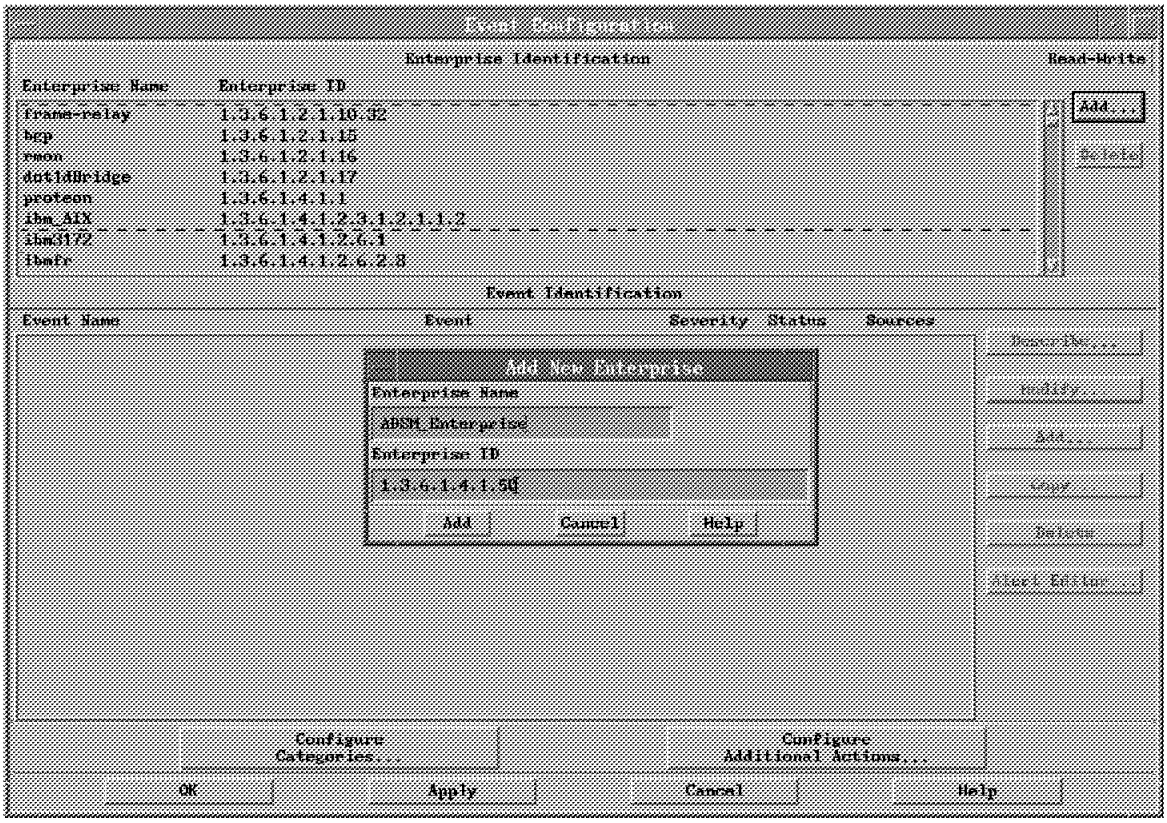


Figure 5. Adding an Enterprise

Once the new enterprise has been added, at least one new trap must be added to it. In the lower part of the event configuration window is event identification; click on **Add**, and this will bring up the Add Event Window, as shown in Figure 6 on page 114.

The trap used in this example of restarting an ADSM server that has terminated should be configured in this window as follows:

Event Name	Restart_ADSM
Generic Trap	Enterprise Specific
Specific Trap Number	1

Event Description	This event will automatically restart an ADSM server that has been accidentally halted by an administrator (optional)
Source	A list of nodes that may generate this trap (optional)
Event Category	Application Alert Events
Severity	Major
Source Character	A
Event Log Message	Defaulted
Popup Notification	ADSM Server on Node $A halted. Recovery Action started
Command for Automatic Action	/usr/local/adsm/start_adsm $A

Once the trap has been configured, the **OK** button should be clicked, returning the user to the event configuration window. Note that the changes made do not take effect until the **OK** button has also been clicked in the Event Configuration window.

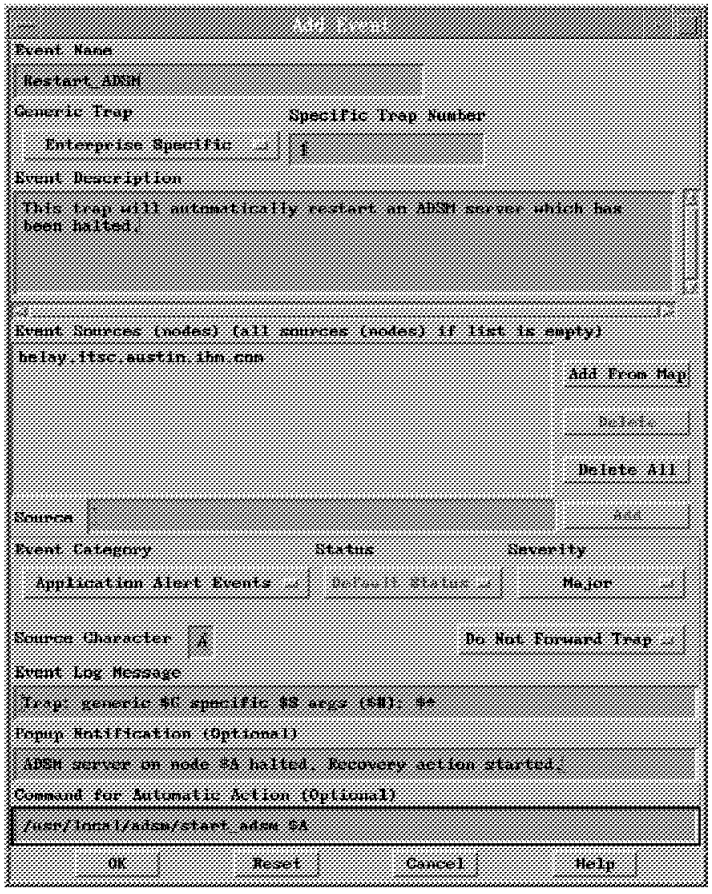


Figure 6. Sample Trap Configuration

10.4.4 When the Server is Halted

Using the samples described above, the effect of a server shutdown by an administrator at the ADSM host is that the server is unavailable for less than 30 seconds. The actual amount of time is variable, depending on host and network load. The NetView operator sees the window shown in Figure 7 almost immediately. While this window is displayed, the command for automatic action is executed on the NetView host, using rsh to restart the halted ADSM server.

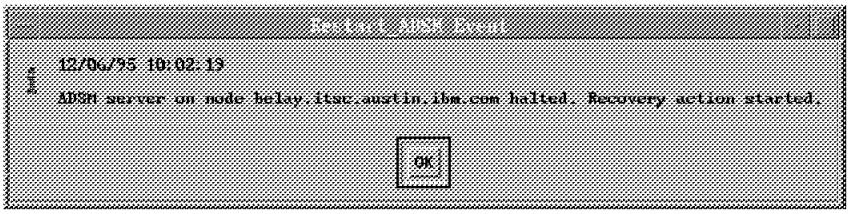


Figure 7. Event Pop-up Notification

The NetView operator must respond to the pop-up window by clicking the **OK** button. Even if the pop-up notification is not configured for an event, the operator still receives notification in the NetView events display; Figure 8 on page 116 shows the one generated by this sample.

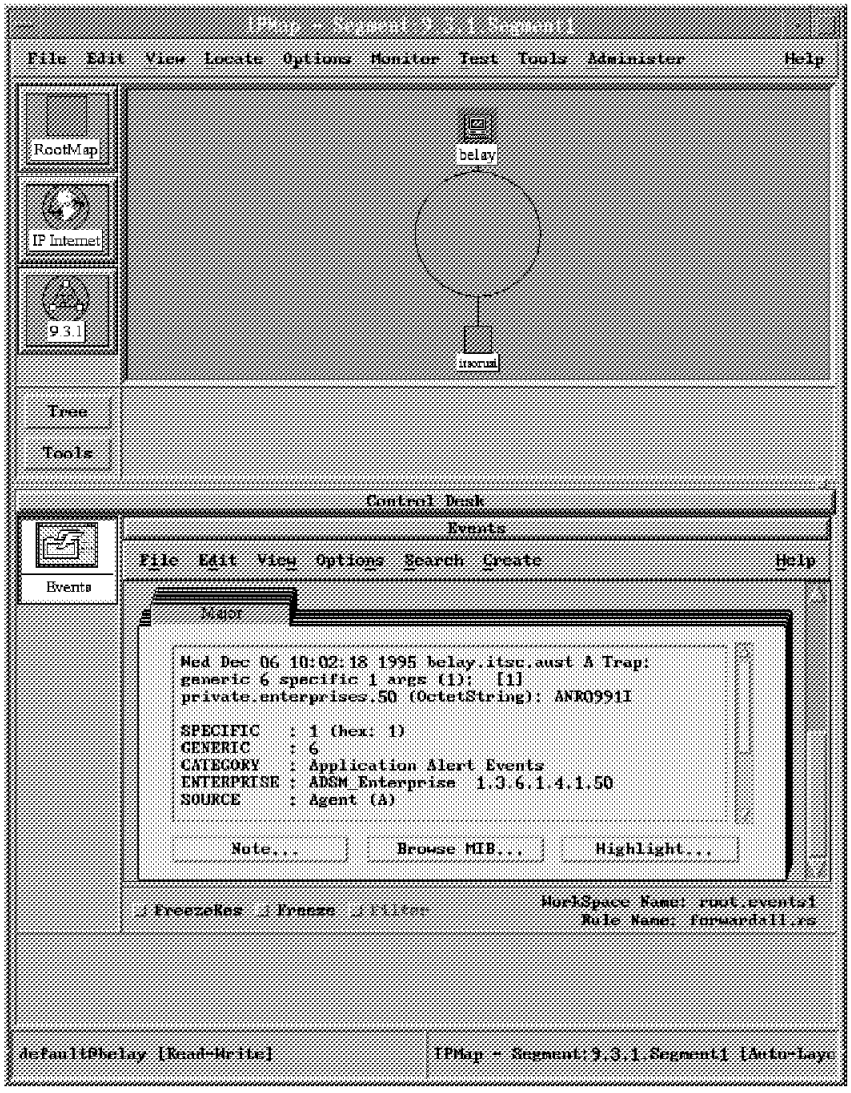


Figure 8. Root Window Event Notification

While the above example uses the most basic NetView features to manage ADSM, it may extended quite easily. The easiest alteration would be to make the command for automatic action executed by NetView on receipt of a trap:

```
rsh $A '/usr/local/adsm/start_adsm_c &'
```

This executes the remote C routine directly rather than invoking a local shell script. Remember that the command for automatic action is restricted to one line of display; for more complicated actions, the execution of a shell script is appropriate.

10.5 Processing ADSM Tape Mount Messages

Those customers that do not have automatic tape libraries probably find that missing a tape mount on a manual library is the most common cause of failed backups. Fortunately, this situation can easily be managed by using standard NetView facilities.

10.5.1 Method

In most installations, local operators have responsibility for responding to the ADSM tape mount messages. The default is for ADSM to generate a message every minute that a request remains unfulfilled for up to an hour (mountwait=60). The NetView operator at the network control center can be alerted to an unfulfilled tape mount by extending the approach used in the previous example.

The code that examines the ADSM messages is modified to recognize the tape mount messages and forward them to NetView. It may be seen as pointless to forward all such messages to NetView since the central operator will not act on most messages; concern will begin only when, say, a mount request is outstanding when there are 15 minutes of the original 60 minutes remaining. This implies some sort of filtering being applied to the message presented to the network operator.

This filtering could be carried out in the following ways:

1. Coding the message-gathering routine to recognize the time value in the message, and sending traps only when messages are received with a time of 15 minutes or less.
2. Using NetView Event Stream Enhancements to inspect all traps and filter out or override the event severity for certain traps.
3. Setting the event severity low in the first place. This would probably negate the benefit of informing the network operator as all such event messages would appear to be of about the same importance, even though those generated when short of time are, in fact, the only ones of importance.
4. Defining two different traps to be used when these messages are detected, one to be used for the bulk of the messages and one to be used for messages generated with 15 minutes or less to go.

The following sections give practical examples of:

1. Generating SNMP traps to carry information to the network operator regarding ADSM tape mount requests.
2. Selecting the appropriate requests to display to the operator using NetView V4 Event Stream Enhancements.
3. Configuring NetView to recognize and act upon the traps generated.

10.5.2 Trap Generation

The general principle of generating traps has already been discussed in 9.2, “Integrating ADSM with NetView” on page 84. In order to service ADSM tape mount requests, it will be necessary to use three types of trap:

1. Tape mount request messages

   ANR8326I - request to mount a particular tape in a drive within a timeout period.

2. Tape mounted messages

   ANR8328I - request to mount a particular tape has been satisfied.

3. Tape mount timed out messages

   ANR8351E - a tape mount request was not fulfilled within the time-out period.

The following code fragment was used to generate the traps in the laboratory:

```
# Mount request for a particular tape volume
# NB a mount request spans TWO lines!
# Create a unique-id for this mount request to be used by ESE in
# comparisons

if [ "$msg_no" = "ANR8326I" ]
then

  # Read that second line of the message
  read input_2nd_line

  # Create a unique mount-id for reference
  # Use variable to hold reference to ADSMs
  # numerical mount-id (ie the "001:" or "012:" etc
  # that is at the start of any mount request message.
  # This field used to pass more information to NetView
  # if a mount request times out.
  mountid=$(echo $input | awk '{ print $2$6$10$11 }')
  ADSM_Mountid=$(echo $input | awk ' BEGIN { mount_num = 0 }
                                          { mount_num = substr($2,1,3);
                                            print(mount_num); }')

  # Place whole unique identifier of mount (which
  # includes ADSM & AIX drive names) into an
  # environment variable; referenced later if the
  # mount times out.
  mountreq[ADSM_Mountid]="$mountid"

  # Extract the number of minutes to go from 2nd line
  time_left=$(echo $input_2nd_line | awk '{ print $3 }')

  # Now notify NetView
  /usr/OV/bin/snmptrap $NVhost .1.3.6.1.4.1.50 $hostname  6 2 0 \
                               .1.3.6.1.4.1.50 octetstringascii "$mountid" \
                               .1.3.6.1.4.1.50 octetstringascii "$time_left" \
                               .1.3.6.1.4.1.50 octetstringascii "$input"

# Mount Request Satisfied
elif [ "$msg_no" = "ANR8328I" ]
  then

    # Use sed to remove all periods from line, then awk
    # to create unique key for this mount.
    mountid=$(echo $input | sed -e s/\\./" "/g | awk '{ print $2$5$9$10 }')

    # As the mount has succeeded, the environment
    # variable set above may be reset at this point
    /usr/OV/bin/snmptrap $NVhost .1.3.6.1.4.1.50 $hostname  6 3 0 \
                                 .1.3.6.1.4.1.50 octetstringascii "$mountid" \
                                 .1.3.6.1.4.1.50 octetstringascii "$input"

# Mount Request timed out
elif [ "$msg_no" = "ANR8351E" ]
  then

    # Give the numerical part of the mount request
    # variable set above; allows reference to
    # contents of variable
```

```
    mountnum=$(echo $input | awk '{ print substr($2,1,3) }')

    /usr/OV/bin/snmptrap $NVhost .1.3.6.1.4.1.50 $hostname  6 4 0 \
                                 .1.3.6.1.4.1.50 octetstringascii "${mountreq[mountnum]}" \
                                 .1.3.6.1.4.1.50 octetstringascii "TAPE MOUNT FAILURE" \
                                 .1.3.6.1.4.1.50 octetstringascii "$input"
fi
```

In this particular method, a few points should be noted about the format of the SNMP traps that are generated. The trap generated in response to a tape mount request (ANR8326I) is of generic type 6 and specific type 2. It has three fields, all of type octetstringascii. The fields are:

1. A unique mountid.

   This is created by extracting the following fields from the ADSM-generated message:

   - Numerical mountid

     This is incremented by the server for each mount request it makes. It is a three-digit numerical field with preceding zeroes and followed by a colon. The trailing colon is not removed by this code.

   - The name of the tape being requested

     This is usually six alphanumeric characters.

   - The AIX device identifier

     The tape drive that the request is to be fulfilled on.

   The unique mountid is created for two reasons:

   a. To create a field which can be used within NetView facilities to easily test for an association between traps

      Remember that these messages may be generated in a network environment where many events occur in each hour and two ADSM messages that relate to the same event may be separated in time by up to an hour. Being able to test for an association between traps allows NetView functions to automate responses efficiently.

   b. The field provides more information to the network operator when notified

      ADSM refers to tape drives by its own names; this might be a name such as TAPEDRIVE for the unit in use. This name is unlikely to be meaningful to a network operator unfamiliar with ADSM, but he may recognize that the field /dev/mt0 refers to a UNIX device.

   Note

   There are a number of ways to create a unique mountid; this is only a sample. When defining the fields of SNMP traps for an installation, consider carefully how to create identifiers that will be unique to a particular event (like a tape mount) rather than a particular message. Some ADSM tasks may generate clusters of messages which are related.

   When defining such a field, it is probably wise not to rely entirely on the ADSM server's unique mountid (the numerical field described above) as this field is reset to 001 each time the server is started.

2. Time remaining

The amount of time left in minutes before this tape mount request will time out. This is extracted from the second line of the tape mount request.

Note

When accessed by a program via a pipe, some ADSM messages span two lines. When processing messages, be aware of the current message length limit.

3. Message text

   The complete text of the ADSM message is forwarded to NetView to assist in any first-level problem determination procedure.

The trap generated in response to a tape mount request fulfilled message (ANR8328I) is of generic type 6 and specific type 3. It has two fields of type octetstringascii, and they are:

1. The unique mountid as described above
2. The complete message text

The trap generated when a tape mount times out (ANR8351E) is of generic type 6 and specific type 4. It has three fields of type octetstringascii as follows:

1. The unique mount request described above
2. The string TAPE MOUNT FAILURE as a direct warning to the network operator
3. The complete message text

10.5.3 NetView Customization

The customization of NetView to support the ADSM mount messages relates to the three types of trap that are to be used. In the laboratory, these were defined under the enterprise name ADSM_Enterprise.

The trap to be generated in response to the mount request messages (generic type 6 and specific type 2) was defined as event `ADSM_Mount_Messages`. All fields were allowed to default, except the Event Category which was set to `Application Alert Events` and the Severity which was set to `Minor` (see Figure 9 on page 121).

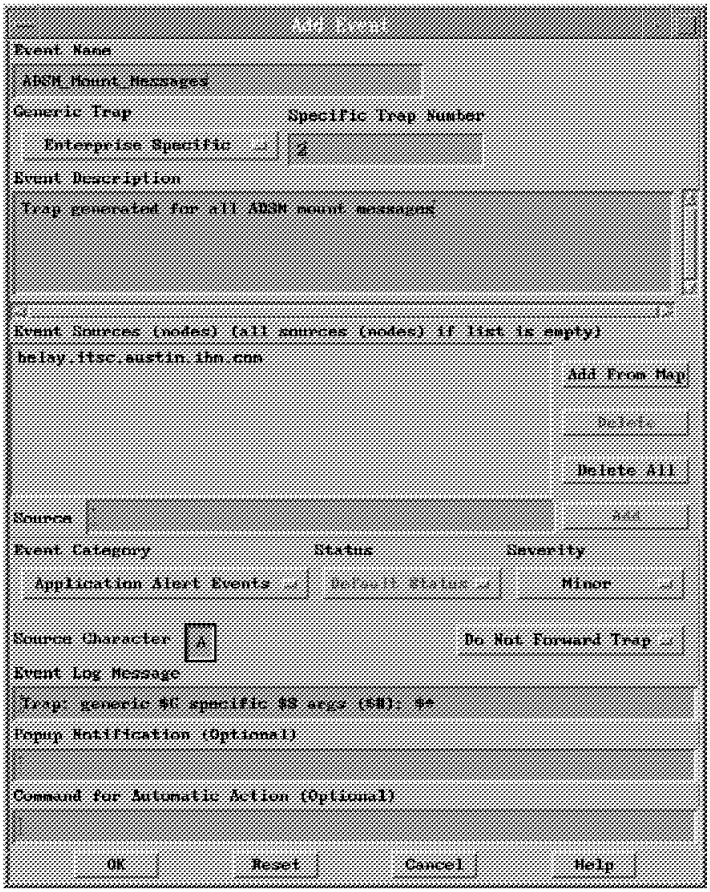


Figure 9. Definition of Trap Corresponding to ADSM Mount Request

The trap generated on fulfillment of a tape mount request (generic type 6 and specific type 3) was defined as an event called ADSM_Tape_Mounted_OK. All fields were defaulted, except the Event Category which was set to Application Alert Events and the Severity which was set to Minor, as shown in Figure 10 on page 122.

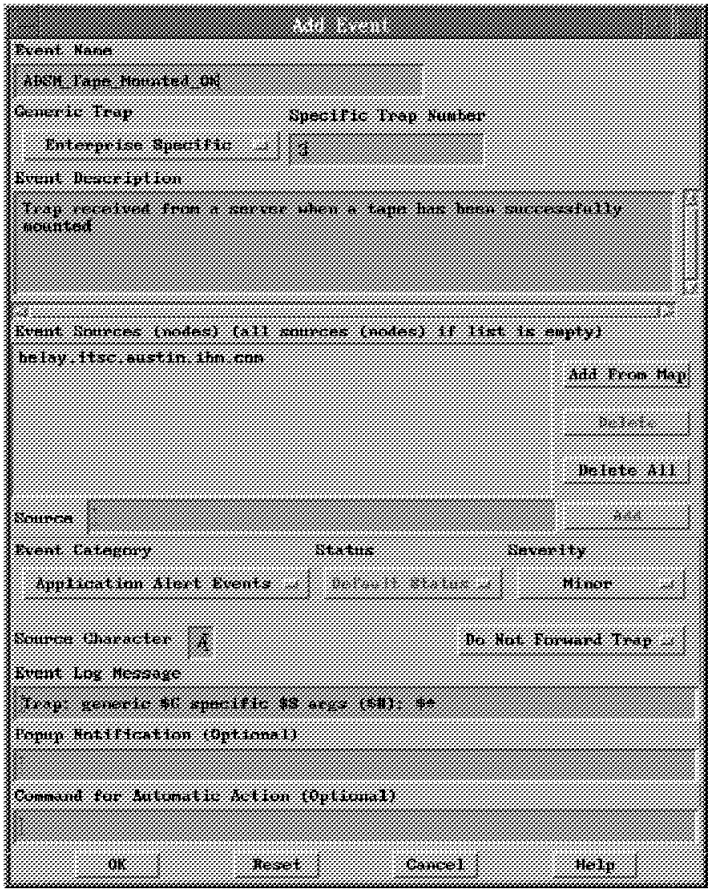


Figure 10. Definition of Trap Corresponding to ADSM Tape Mounted Message

The trap generated when a tape mount request timed out (generic type 6 and specific type 4) was defined with the event name ADSM_Tape_Mount_Failed. It is shown in Figure 11 on page 123.

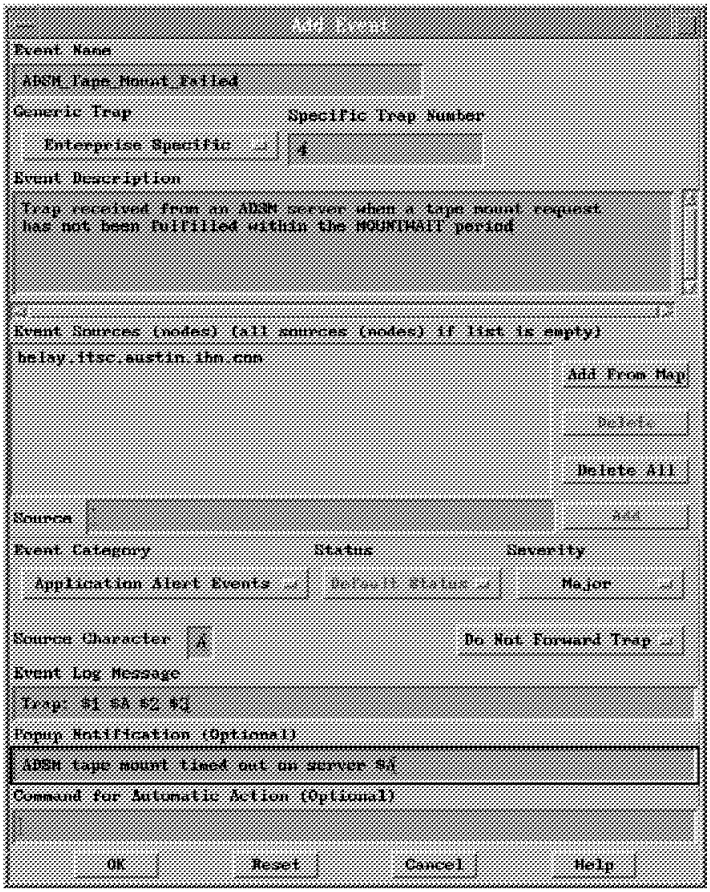


Figure 11. Definition of Trap Corresponding to ADSM Tape Mount Time Out

The Event Category was set to Application Alert Events and the Severity to Major. The Event Log Message Format was defined as `Trap: $1 $A $2 $3`. The Pop-up Notification field was set to `ADSM tape mount timed out on server $A`.

10.5.4 Trap Design

At the outset of this exercise, it was known that it would be necessary to analyze a cluster of messages developing over a period of time. Close attention was paid to the positioning of the trap fields within each trap to maintain the same information in the same positional parameter of each trap. Care taken at this stage saves time and complexity later. This example uses three traps whose parameters are as follows:

1. The first variable passed in each trap is a unique mountid created by the trap-generating program from the ADSM message.
2. The second variable is the time left before expiration of the MOUNTWAIT period or the TAPE MOUNT FAILED message once the time has expired.
3. The last variable (the second variable in the case of a successful mount) is the first line of the ADSM message.

The parameter structure is designed this way to minimize the amount of processing which must be performed by NetView.

10.5.5 NetView Event Stream Enhancements Customization

Event Stream Enhancements, available in NetView V4 is the crucial function used in processing the traps that are generated in response to the ADSM mount messages. The main features of Event Stream Enhancements are described in 10.3.2, “Using the Event Stream Enhancements of NetView V4” on page 107. These features can be used to avoid displaying events to the network operator that are insignificant, while highlighting out-of-line situations.

In this example, the situation is that a cluster of up to 60 mount request messages will be generated by ADSM at one minute intervals over a period of an hour. This cluster will be ended by either of the following messages:

- A tape mounted message (ANR8328I)
- A timeout message (ANR8351E)

The mount request events are of no concern to the network operator; they will, in the normal course of events, be fulfilled by the local system operator or by an automated tape library. As long as the network operator is aware that a tape mount has been requested, then that is sufficient. The situation becomes much more important when the request times out before being met. However, if the request is met before timing out, then the remainder of the transaction is of no importance to the network operator.

This section defines a NetView Event Stream Enhancements ruleset that will allow the network operator to see the first mount request from the ADSM server and to be notified when a request times out; it will also delete all notifications relating to fulfilled requests.

10.5.5.1 Building the Event Stream Enhancements Ruleset

The first step in building any ruleset is to determine which traps will be used and then how their fields will be matched where necessary.

Start the ruleset editor by choosing **Tools** and then **Ruleset Editor** from the NetView Root window. This will produce two windows:

1. The Templates window (Figure 12)

   The template window contains all the templates of nodes that may be added to the ruleset work area; they are used by clicking on an item in the window and dragging it into the ruleset work area.

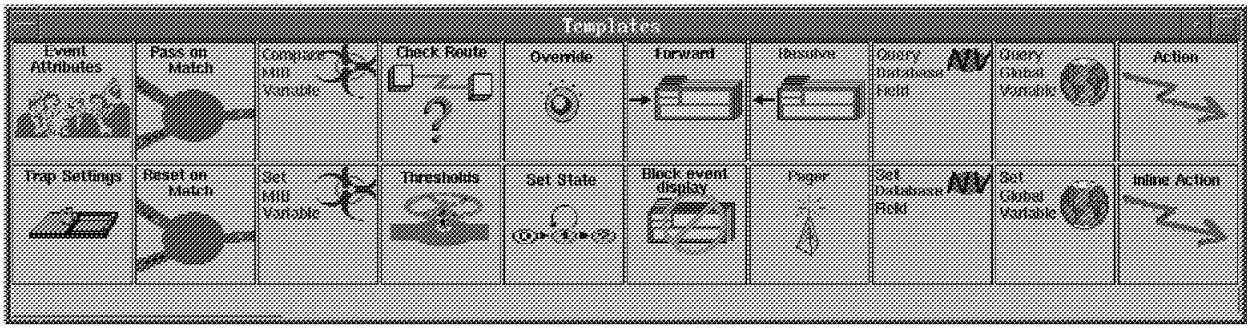


Figure 12. Ruleset Editor Templates Window

2. The rulesets work area window (Figure 13 on page 125)

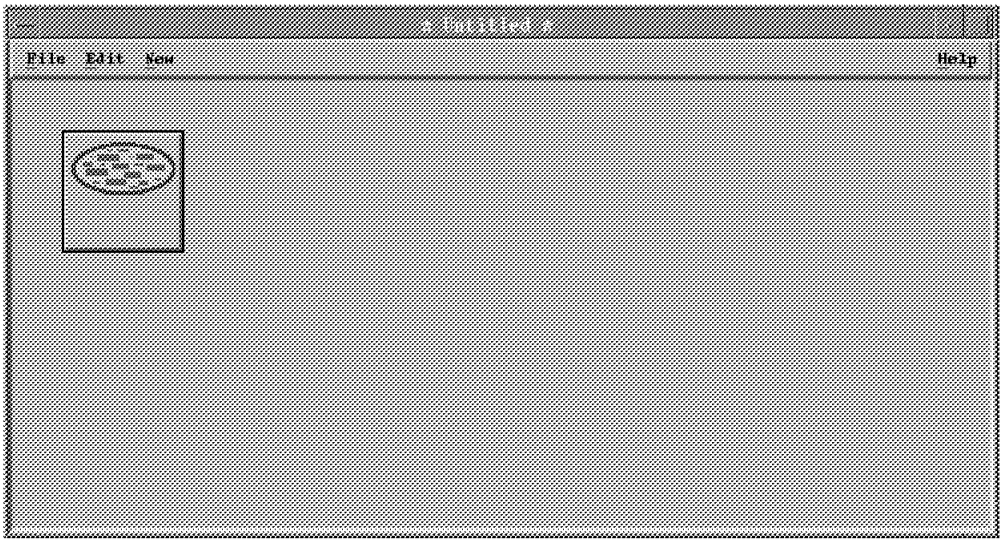


Figure 13. Rulesets Work Area Window

This example will use the following node templates:

- Trap Settings

  To specify that a particular trap be processed.

- Event attributes

  To compare the value of a field in a received trap with a literal value.

- Pass on Match

  To compare a single field across two traps.

- Override

  To change the severity assigned to an event.

- Forward

  To display an event to the operator.

- Resolve

  To delete from the operator display an event card that has been resolved or superseded.

Click on the **Trap Settings** icon in the node templates window, and drag it into the rulesets work area window using the middle mouse button. Another window will appear (the Trap Settings input window, see Figure 14 on page 126).

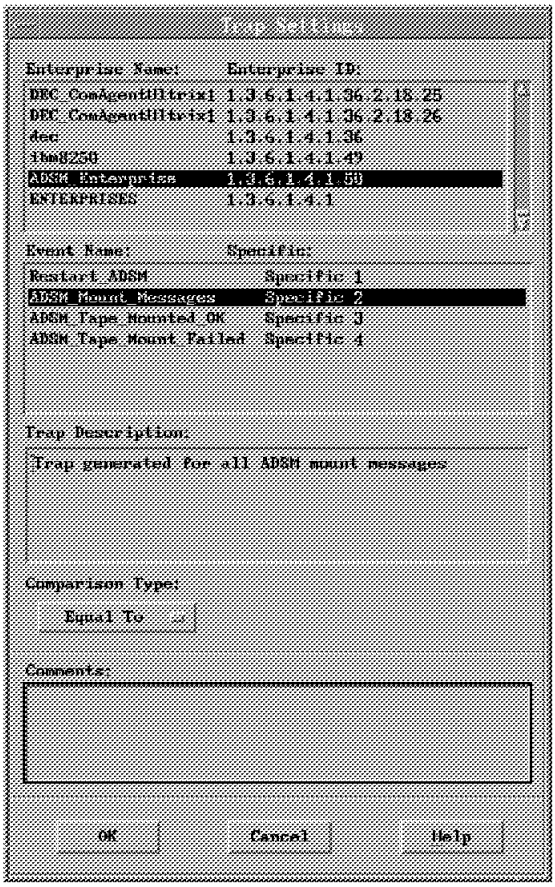


Figure 14. Entering Trap Settings for ADSM Mount Messages

This window presents a scrollable list of all traps which are currently defined to NetView. In this example, the enterprise ADSM_Enterprise was defined, with three associated traps. Select the enterprise-ID to be used in the installation and the appropriate event. This event corresponds to an ADSM mount message and is generated at one minute intervals. The Trap Description and Comments fields are optional. Comments may be entered to assist in recalling the purpose of a particular ruleset at a later date. The event name, ADSM_Mount_Messages, was used in this example.

Once all fields are filled, click on **OK**; this completes the first node of the ruleset. In order for processing to reach the newly-defined node, it must be connected to an input. The input for a ruleset is represented by the Event Stream icon in the ruleset work area. To connect the Event Stream to the Trap Settings node, click on **Edit** in the menu bar, and choose **Connect Two Nodes** from the pull-down menu. The mouse pointer will change to a connection icon. Now, click on the first node and the second node in the ruleset. An arrow will be displayed indicating the direction of flow.

Note

The flow of processing in the ruleset will be from the first node that is selected to the second. Checking the order of processing is easy; look at the direction of the arrow between the two nodes.

If the direction of flow is wrong, select **Edit** and then **Delete Line** from the pull-down menu, and click on the offending line. It will be necessary to recreate it.

In order to prevent each ADSM mount message from being displayed, relevant messages must be chosen from the stream of traps. The selection mechanism is invoked by using the Event Attributes template. Drag the Event Attributes template into the work area; it will automatically be connected by Event Stream Enhancements to the last node created. The Event Attributes window will be opened as in Figure 15. This is where the attributes of a trap that will be passed through this node for further processing in the ruleset are defined.

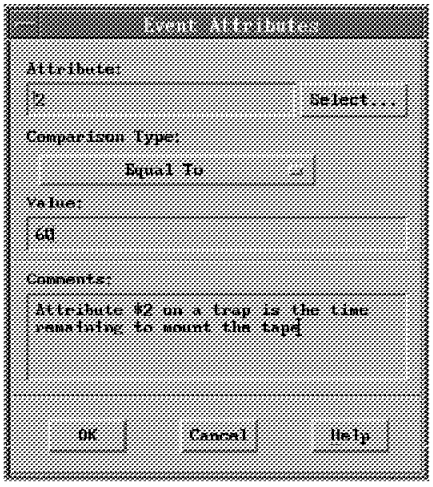


Figure 15. Setting Event Attribute Node Parameters

The object is to display the first mount request message to the network operator to keep him informed. This message will be contained in a trap which also carries the value 60 in the second variable. Hence, the Event Attributes window is filled in by selecting field 2 from the selection list and testing it against the literal value 60. Any trap reaching this node will be passed on for further processing provided the value of its second variable is 60. This node, therefore, blocks all mount messages except the first from further processing. The first message will be passed, but must be explicitly forwarded to the nvevents process for display. This is done by dragging the **Forward** template into the work area and connecting it to the event attributes icon. The input to the Forward window (Figure 16 on page 128) is an optional comment.

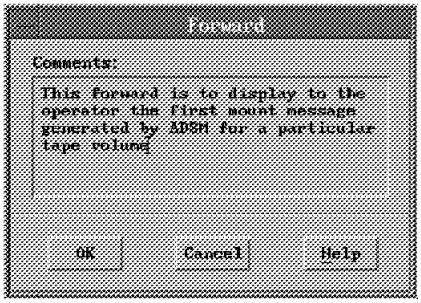


Figure 16. Forward Node Input

The ruleset should now appear as shown in Figure 17. It must now be extended to cater for the later messages. The network operator must be informed when the tape mount is unfulfilled with 10 minutes left before time-out.

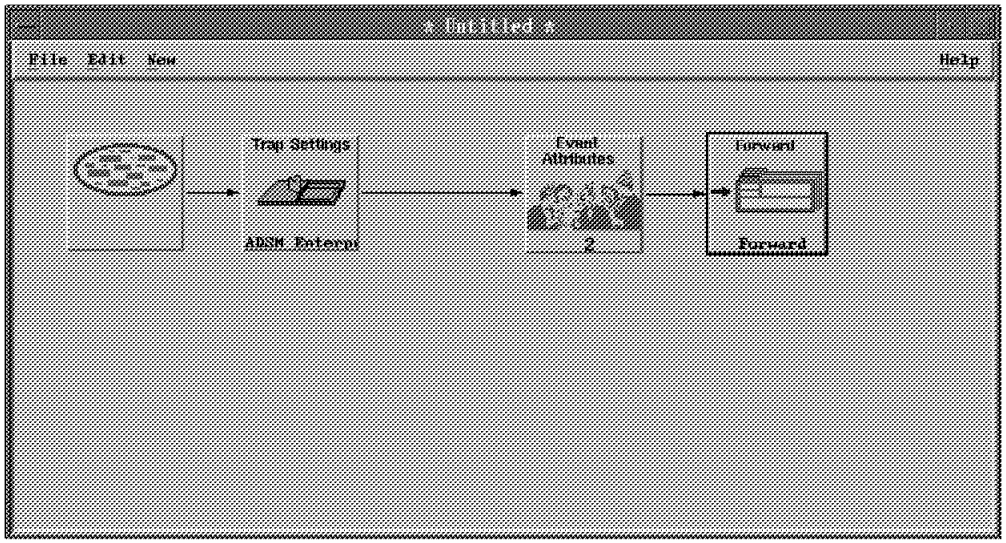


Figure 17. Ruleset After Defining Processing of ADSM_Mount_Messages

This can be done in the same ruleset. Drag a second **Event Attributes** icon into the work area connecting it so that it takes its input from the **Trap Settings** node that is already defined. Again, the Event Attributes window appears, but this time, test the same attribute - time left in which to mount the tape, the second variable in the trap - against the literal value 10. This is shown in Figure 18 on page 129.

This node will pass on for processing the event generated when there are 10 minutes left to mount a tape.

Now, it is necessary to highlight this event to the NetView operator in some way.

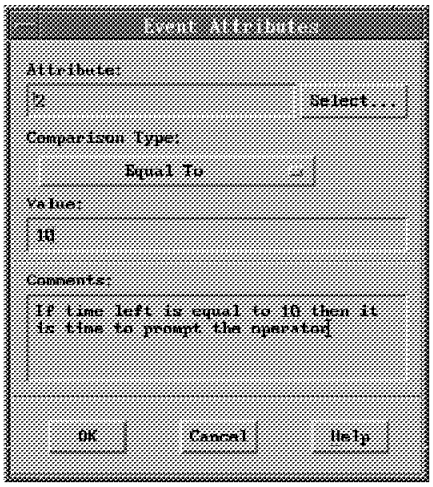


Figure 18. Setting Event Attribute Node to Detect 10 Minute Warning

This is implemented using the Override node. Drag the **Override** node template into the work area. The Override window, Figure 19, appears allowing the choice of overriding the event status or severity.

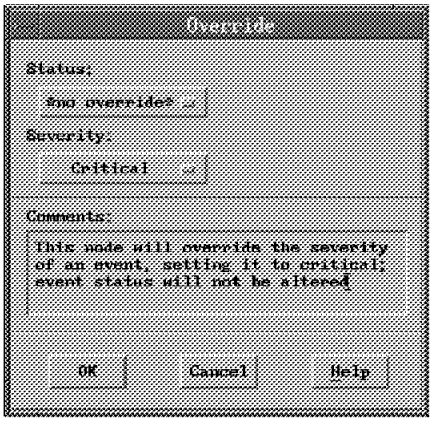


Figure 19. Defining an Override Node

This example sets the Severity to `critical`.

Note

The Override node carries with it an implicit forward (note that the implicit forward does not appear in the work area). Therefore, an Override node will automatically forward the event it has overridden, and no Forward node should be added to the ruleset after an override.

If a forward is added, then two event cards will appear on the NetView Event Control desk.

The ruleset will now display two events: one when there are 60 minutes to go before a mount request times out and one with severity set to critical when there are 10 minutes before timeout. According to the objectives, once the 10 minute message is displayed, there is no need to continue displaying the original 60

minute message, and it can be removed from the display. This is done by implementing a Pass on Match node and a Resolve node.

Dragging the **Pass on Match** template into the work area produces an input window like the one shown in Figure 20. The function of Pass on Match is to recognize clusters of incoming traps. It must be used carefully.

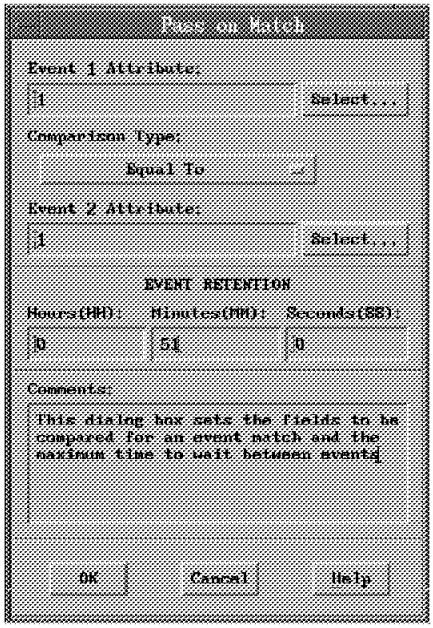


Figure 20. Setting the Parameters of the Pass on Match Node

This example could be implemented in an environment in which messages are coming from a number of sources; several mount requests might be generated in the same time period. In this case, it would not be wise to set the Pass on Match function to test the field used in the Event Attribute nodes above. This is why the shell script that generates the SNMP traps in the example always generates a mountid value and places it in the first variable of the trap. This variable is a string made up of the following:

- ADSM mount request number
- ADSM label of the tape requested
- ADSM device name
- AIX device driver name

The mountid field allows the Pass on Match node to determine immediately whether two traps are in fact associated with the same ADSM event; that is a mount of the same tape volume. Therefore, the Pass on Match node is configured to check for a match on the first variable within the two traps presented to it.

The Event Retention period is set to a period of time that is at least as long as the time period between the two messages on which the match is to be made. The period between the two messages in this case is 50 minutes; therefore the event retention period is set to at least this time.

Once comments have been added, the **OK** button can be pressed.

One other feature of Pass on Match nodes is the significance of the order in which events are presented to them. It is known that the 60 minute message will be generated before the 10 minute message; the first variable of the traps will be tested by the Pass on Match node, but it is important that the inputs are presented in the correct chronological order. This is done by connecting the Event Attribute nodes to the Pass on Match node.

When the mouse button is clicked on the Pass on Match node to connect it to another node, the Multiple Input Window shown in Figure 21 appears.

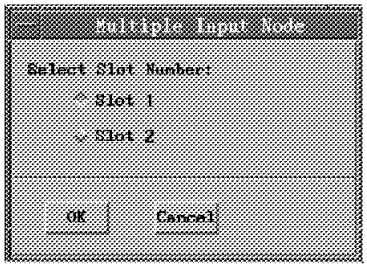


Figure 21. Multiple Input Window

This enables the user to attach the incoming events to the correct slots of the Pass on Match node. The output from the Event Attributes node checking for the trap variable being set to 60 is therefore attached to slot 1 of the Pass on Match. Implement this by clicking on the **slot 1** button on the multiple input node window when connecting the Event Attribute node to the Pass on Match node. Conversely, choose **slot 2** when connecting the other Event Attributes node (where the time left to mount the tape is 10 minutes) to the Pass on Match node.

Now, the action to take when the Pass on Match condition is met must be placed in the ruleset. The Resolve node will resolve (remove from the display) the older of two events passed to it. Now, drag a **Resolve** node into the work area, and connect the output of the Pass on Match node to it. The Resolve window (Figure 22) appears; the only input to this window is a user-defined comment.

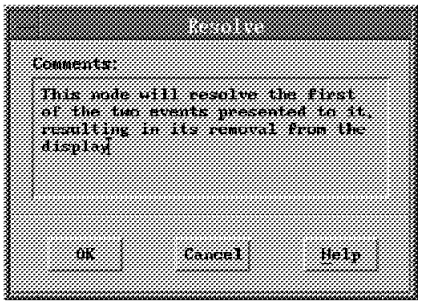


Figure 22. Resolve Node Input

Note

The event that is resolved is always the one that was presented to the Pass on Match node on slot 1. Always ensure that the event to be removed from the display is passed to the Pass on Match node via slot 1.

The ruleset work area now looks like Figure 23 on page 132. This rule will display the first event - the mount message with 60 minutes to time out - and the

message generated with 10 minutes to time out. Events referring to the same mount request, that is having the same mountid in the first field of the trap, are not displayed. When the 10 minute message arrives, the 60 minute message is resolved and deleted from the display, and the 10 minute message is displayed with a severity of critical.

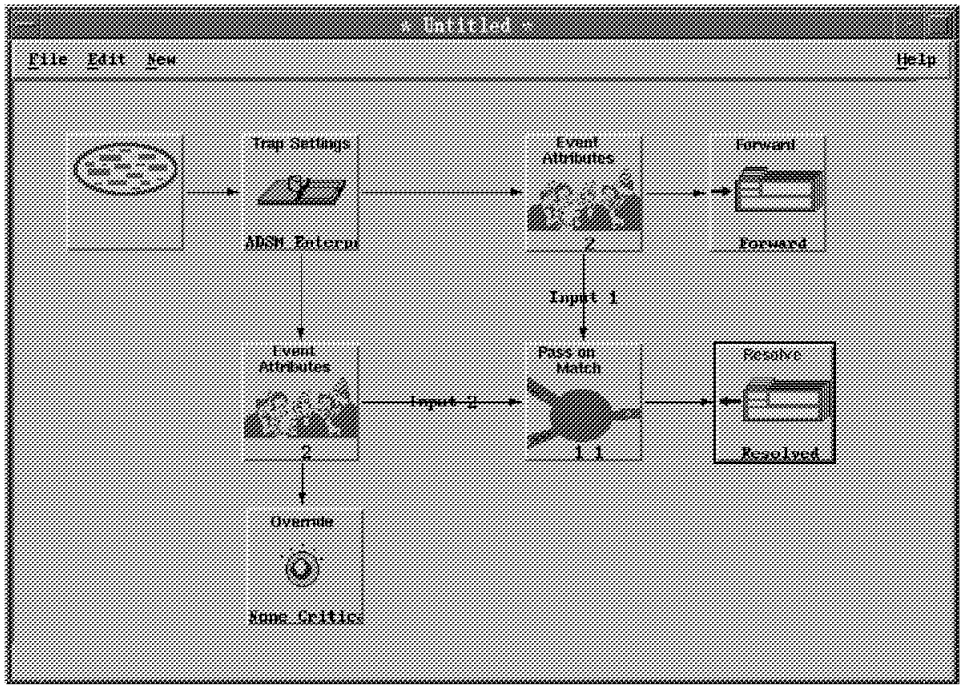


Figure 23. Ruleset After Defining Resolve Processing

When the requested tape is mounted in the drive, message ANR8328I is generated. This is converted into a trap of specific type 3 by the program reading the ADSM messages. It should be noted that this message does not give all of the fields required to fill the unique mountid; this has been worked around in the trap-generating code. The impact of a tape mounted message on the ruleset under construction must be considered. It will always arrive after a tape mount request message (either the 60 minutes or the 10 minutes to timeout message), and its processing in the ruleset must delete the previous event card from the nvevents display. The way that this is achieved in a concise manner shows the power of Event Stream Enhancements and the importance of planning the format of the traps used.

The current ruleset is modified by dragging another **Trap Settings** node into the work area. It is attached to the Event Streams node and configured through the Trap Settings window to pass specific type 3 events. These are ADSM_Tape Mounted_OK events, as shown in Figure 24 on page 133.

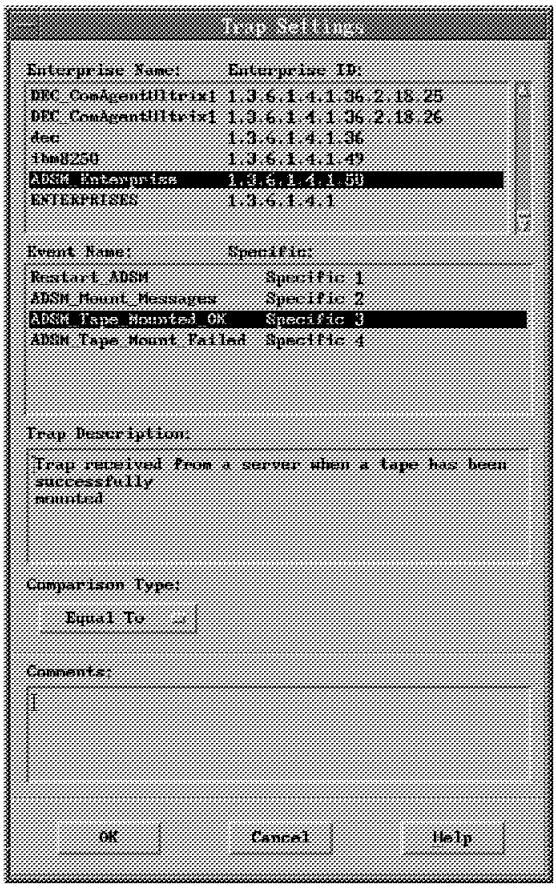


Figure 24. Trap Settings Node: Selecting the Specific Trap 3

The output from this node is then connected to the input of the original Pass on Match node on slot 2. The Event Attributes node for events with 10 minutes to timeout is then modified to pass processing onto the input of the same Pass on Match on slot 1, in addition to the already defined output to slot2. This ensures that when an ADSM_Tape Mounted_OK event arrives, it will resolve (and therefore delete) the previously displayed event, whether it was the 60 or the 10 minute message.

Do note that this processing can be carried out by such a concise ruleset only because the variable that the Pass on Match node inspected carried the same information, even though the trap-specific ID was different. This was considered in planning.

No further changes are required to support the tape mounted message. Hence, the arrival of a tape mounted message will delete whichever of the two previous messages is displayed at the time. It will not itself be displayed; it is not connected to a Forward node in the ruleset, and there is no requirement to display such events to the network operator.

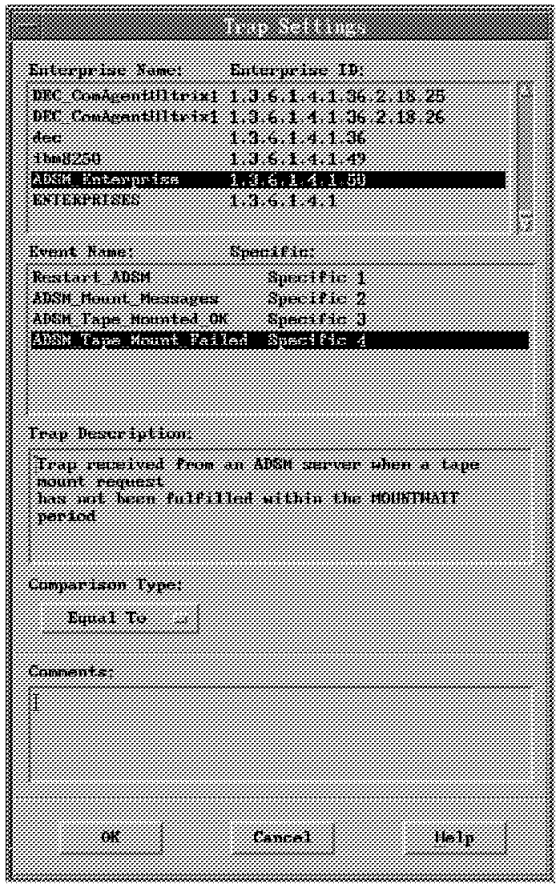


Figure 25. The Completed Ruleset for Managing ADSM Tape Mount Messages

Those occasions when the tape mount request times out are dealt with in a similar way to the successful tape mount event. In this example, the timeout event can only occur while the 10 minute warning event is being displayed. Once the mount has timed out, the objective is to warn the operator and remove the 10 minute warning event. This is accomplished by dragging another **Trap Settings** node into the work area and configuring it as an ADSM_Tape_Mount_Failed (specific type 4) event as in Figure 26 on page 135.

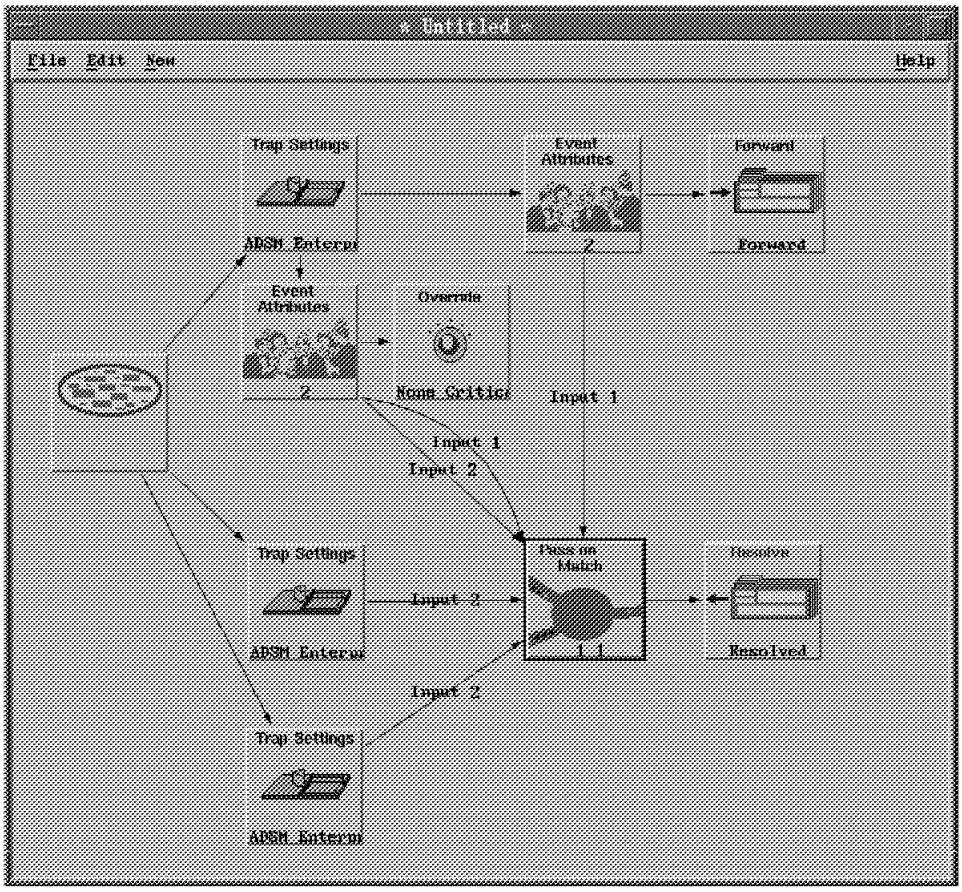


Figure 26. Trap Settings: Configuring Specific Trap for Timed-Out Mounts

Connect this node to take its input from the Event Streams node and to direct its output also to the slot 2 input of the Pass on Match node. The ruleset is now complete, as shown in Figure 25 on page 134. The result is that the currently displayed event is resolved and that although the ADSM_Tape_Mount_Failed event is not displayed, the optional pop-up does take effect.

10.5.6 When a Tape Mount Request Occurs

When an ADSM tape mount request occurs, the operator is prompted with the initial message that a tape needs to be mounted. With 10 minutes remaining, the operator will be prompted with this fact and the first message removed (if the operator has not already done so).

If the tape is changed in this time period, then the operator will receive the successful mount message; otherwise, at the expiry of the MOUNTLIMIT, the tape mount failed message will be presented.

Chapter 11. Trouble Ticket Scenarios

This chapter will look at some scenarios involving ADSM, NetView and Trouble Ticket.

11.1 Interfacing to Trouble Ticket

Once an ADSM event has been determined to have failed and the network operator informed, some form of problem management should be applied. As explained in 8.4, “Trouble Ticket/6000” on page 77, Trouble Ticket for AIX fulfills the need for problem management functions; it can also be easily associated with NetView for AIX.

In this scenario, a newly installed Trouble Ticket subsystem was loaded with the demonstration database supplied with the product. This was done by running the following command:

```
> /usr/bin/nx_load -f/usr/lpp/tt6000/samples/data/demo.dat
```

This populates the Trouble Ticket database with sample profiles for most of the functions that will be used in passing the ADSM_Tape_Mount_Failed (see 10.5.3, “NetView Customization” on page 120) event into the problem management function. These samples may be used as supplied in some cases, but in most cases, the samples loaded will be used for guidance in creating installation-specific database entries. For example, the sample contacts refer to fictitious persons; customization of a real system would involve defining local problem manager entries and so on.

In order to pass events from NetView to Trouble Ticket, the Collect NetView for AIX events field must be set to Yes during the configuration of Trouble Ticket. To check the value of this field. execute `smit` and take the following options:

1. *Communications Applications and Services*
2. *Trouble Ticket for AIX*
3. *Configure*
4. *Set Options for Trouble Ticket for AIX*

This dialog window will be similar to the one shown in the following screen:

```
                        Configure Trouble Ticket for AIX

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
  Server/Client                                       Server                 +
  Server name                                        [belay]
  Collect NetView for AIX events                      Yes                    +
  Trouble Ticket for AIX database                     Proprietary            +
  Database name or directory                         [/usr/lpp/tt6000/site/d>
  Sybase server name                                 []
  Trouble Ticket for AIX log directory               [/usr/lpp/tt6000/log
  Email user name                                    []
  Mail spool directory                               [/usr/spool/mail]
  Semaphore / Shared Memory base                     [0x4E580000]         X
  Reinitialize database                               No                    +



F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Once Trouble Ticket for AIX is installed, a real-world site is likely to need to populate the following database tables with information relevant to their operation:

- Contact Lists

  These are:

  1. Locations

     Each record gives the physical location of a resource. The description may be as accurate or vague as the Trouble Ticket administrator prefers.

  2. Sites

     A site record is a grouping of locations. For example where a company has several buildings in the same town.

  3. Organizations

     Each record describes a department or division to which Trouble Ticket responsibility may be assigned. For example Help Desk or Network Control.

  4. Individual Contacts

     These records relate to individuals who are important to some aspect of the problem management process.

  5. Vendors

     Each vendor record identifies a company that provides some form of software service or hardware to this organization.

- Resource Inventory

  These can be:

  1. Network resources

     Information about network entities. A restricted form of this may be imported from NetView via automatic discovery.

2. Models

   Detailed information about the different types (hence models) of resource owned by the organization.

For a complete discussion of Trouble Ticket for AIX configuration, see *Network Problem Management Examples Involving AIX Trouble Ticket/6000 - GG24-4014-00*. This discussion will be restricted to that part of Trouble Ticket configuration required to create incidents in response to the messages originating from ADSM and escalated to a reasonable level.

11.2 Trouble Ticket Notification

In order to demonstrate a method of integration of ADSM with Trouble Ticket, the aim here is to automatically generate a trouble ticket from the ADSM messages passed to NetView in previous scenarios. To create Trouble Ticket Incident Reports directly from NetView traps, the following definitions should be made in Trouble Ticket:

- Notification Rules
- Notification Methods
- Message Templates
- Incident Filter Rules

These functions are all accessible from the Trouble Ticket for AIX Problem Management window, as shown in Figure 27. To access them, use the *Administration* pull-down menu.

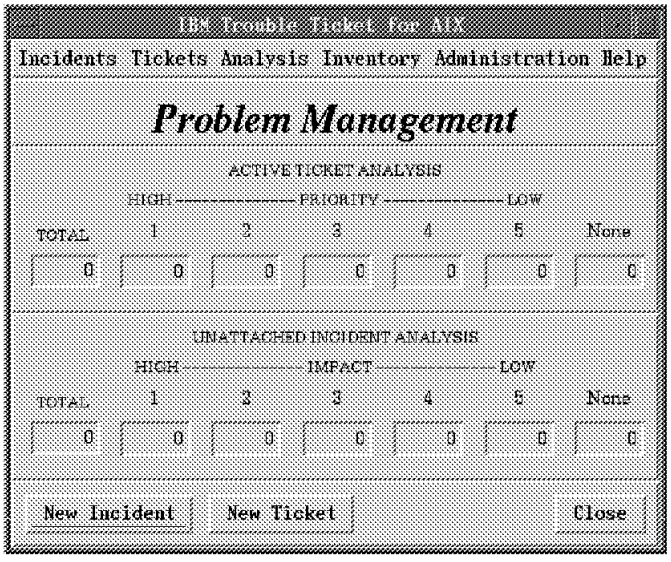


Figure 27. Accessing Notification Configuration Functions of Trouble Ticket

11.2.1 Defining Incident Filters

Incident filters are generally the mechanism used for suppressing the processing of certain incidents by Trouble Ticket. They enable the system administrator to determine the level of response for certain types of event messages. That is to say, what sort of pattern an incoming trap must exhibit before being translated into an incident report. In this example, the areas of interest are those events that are passed from NetView which relate to an ADSM error.

In the current scenario, it may take up to an hour for the error that causes concern to be generated. In that time, any number of other events may be reported by NetView. Some of these other events will be insignificant. For example, certain devices in the network may generate message just because they have been switched on. This type of message does not indicate an error or outage, so it is appropriate to screen these events via incident filters.

To pursue the example, an incident filter will be defined that will generate an incident report for every ADSM_Tape_Mount_Failed event reported by NetView.

From the Trouble Ticket Problem Management window, select the **Incident Filter Rules** option from the Administration pull-down menu. The Incident filter rules window (Figure 28) will be displayed.

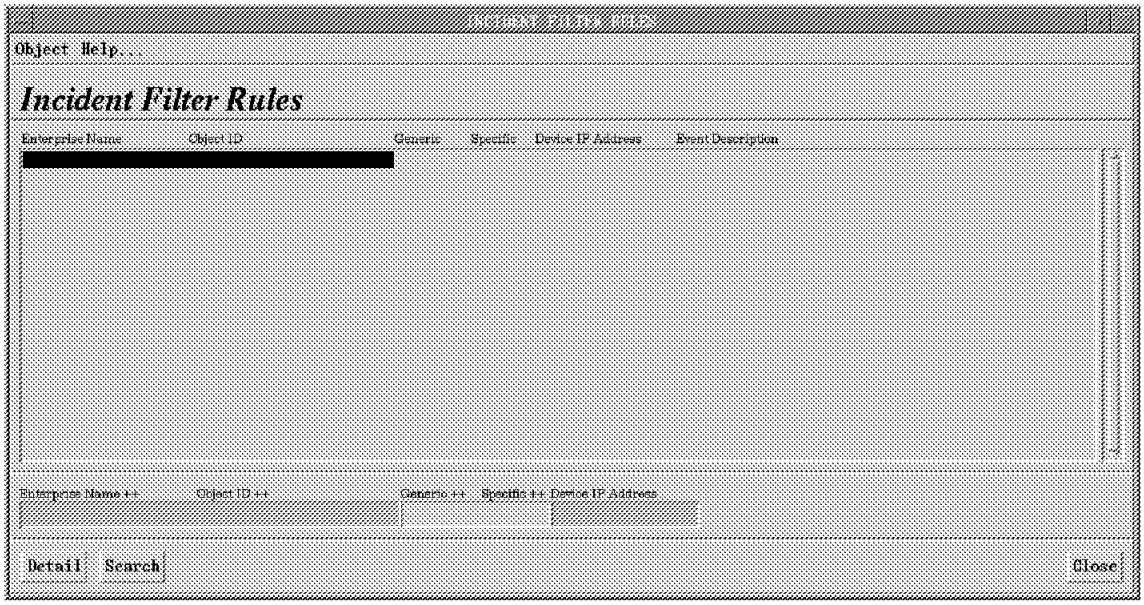


Figure 28. Incident Filter Rules Window: Creating a Rule

Take the **New** option from the Object pull-down menu; the Incident Filter Detail window is displayed, as in Figure 29 on page 141.

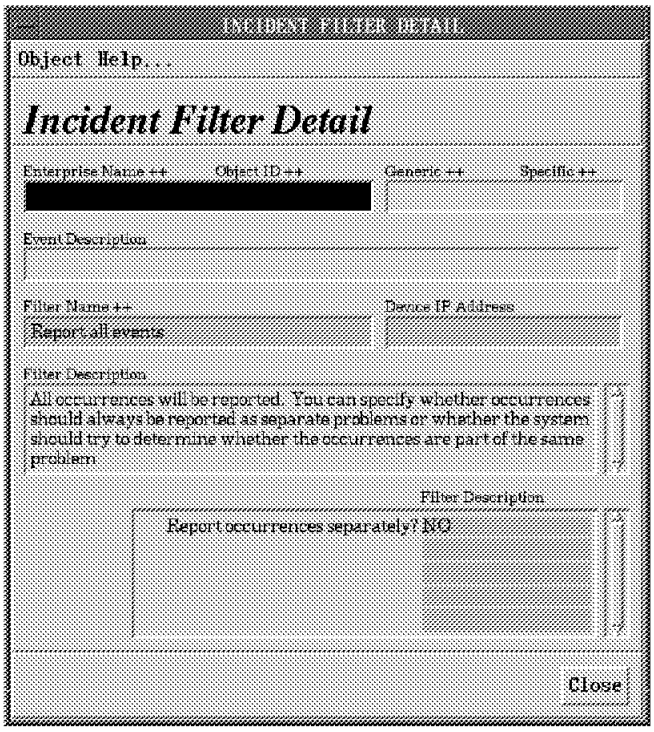


Figure 29. The Incident Filter Detail Window

This window allows either direct input or assisted input by double-clicking on the input field to see the range of available values. Fields which have a double plus sign (++) following their name in the window are required input fields; all of these have assisted input available.

Figure 30 shows the Enterprise selectable list window that is displayed by double-clicking in the **Enterprise-Name** field of the Incident Filter Detail window.

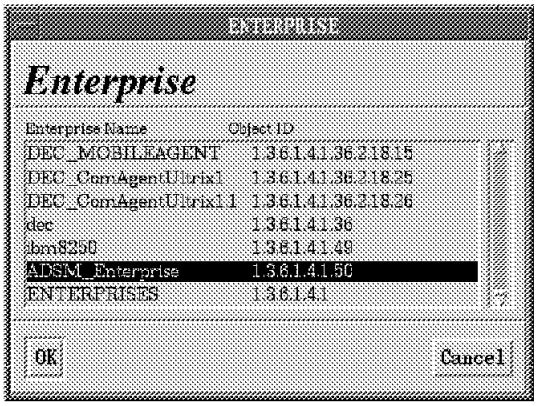


Figure 30. Enterprise Window - Selecting Incident Filter Details

Double-clicking in the **Generic** field brings up the Trap window (Figure 31 on page 142). Selection of a specific trap in this window also fills in the Trap Description field of the Incident Filter Detail window.

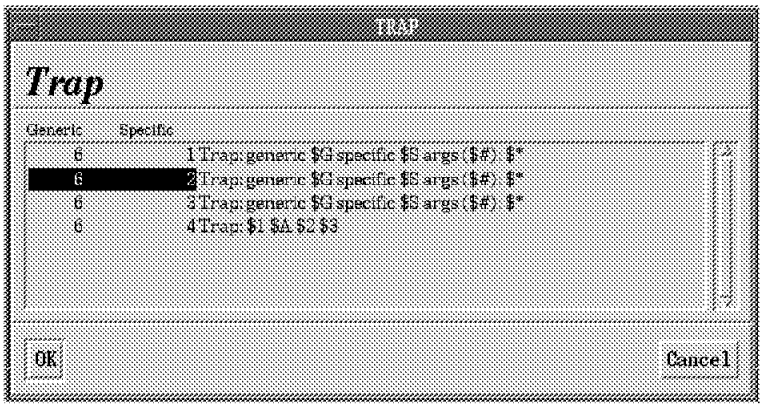


Figure 31. Trap Window: Selecting the Specific Trap

As detailed above, the specific trap used by an ADSM_Tape_Mount_Failed event is generic type 6 and specific type 4.

Once the enterprise name, object ID and trap numbers have been filled in, double-click on the **Filter Name** field. This will open the Filter List window, which details the four available filters, as shown in Figure 32.

Figure 32. The Filter List Window

`Report All Events` is selected in this example; other values can be experimented with.

Finally, set the Report occurrences separately field to `YES`, then close the Incident Filter Detail window; at the prompt, `Save Changes?`, click on the **Save** button. This completes the Incident Filter Rule preparation for the example. The Incident Filter Rules window should now appear as in Figure 33 on page 143, displaying the rule that has just been defined.

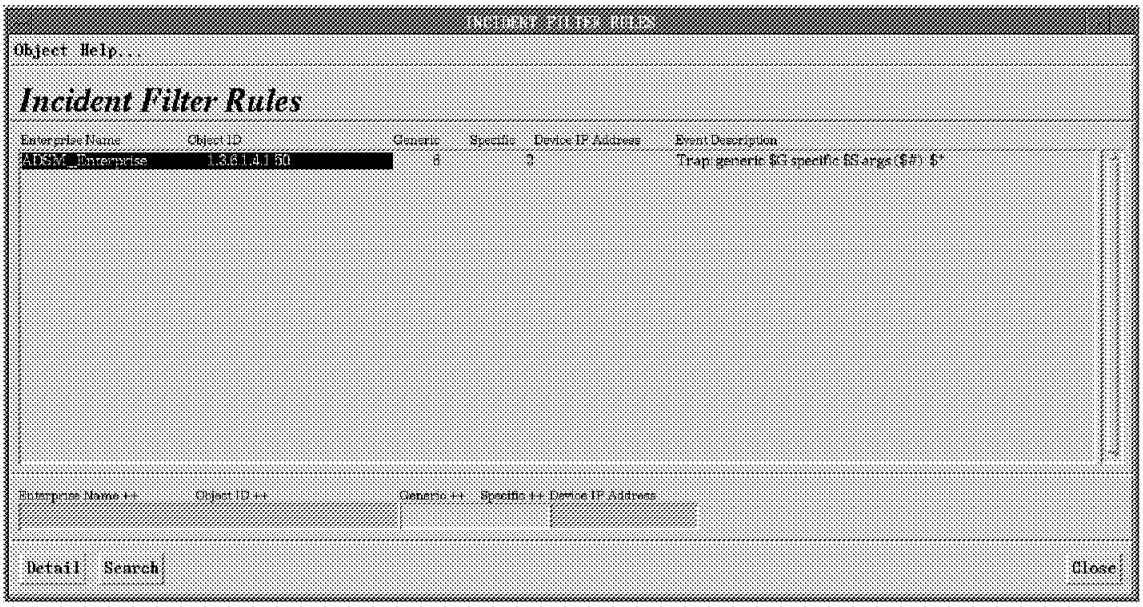


Figure 33. Completed Incident Filter Rules Window

11.2.2 Configuring Notification Methods

Trouble Ticket provides a tailorable notification process that allows the administrator to define rules for the notification of all those involved in the process of resolving a problem. The notification rules are made up of the following:

- Notification Methods

  The means of delivering Trouble Ticket messages through the notification process.

- Message Templates

  Define the formats of notification messages sent by Trouble Ticket.

- Notification Rules

  Implement the notification process.

In the standard Trouble Ticket for AIX installation, an incoming event with a filter defined as above will remain as an incident until picked up by the administrator. The administrator may then choose to open a trouble ticket and associate one or more incidents with it. Once a trouble ticket is opened, then the rules of notification and escalation defined within Trouble Ticket for AIX will come into play, guiding the problem through the resolution process.

However, this example would not be properly complete if this were the case; it would be generally expected for an ADSM mount message to be generated during the night, when the Trouble Ticket administrator is probably absent. One would prefer then to automatically create a trouble ticket from the incoming event and to direct it to an department that is available throughout the day, perhaps the helpdesk function.

Fortunately, IBM provides the means to automatically create a trouble ticket from an event with Trouble Ticket for AIX. The program in

/usr/lpp/tt6000/samples/ntf_meth/tt_script can be used to automatically generate a trouble ticket from an incident.

In the incident filter rule definition above, it was defined that each ADSM_Tape_Mount_Failed incident will be reported separately by Trouble Ticket. Now if the Notification Setup is configured to use the sample program, trouble tickets will be generated automatically for each ADSM tape mount that times out.

First, copy the sample program to another location and make it executable.

```
> cp /usr/lpp/tt6000/samples/ntf_meth/tt_script /usr/local/tt6000/tt_script
> chmod 755 /usr/local/tt6000/tt_script
```

Then select **Administration**, then **Notification Setup** and finally **Notification Methods** from the menu bar of the Trouble Ticket Problem Management window. On the Notification Methods window that is displayed, select **Object** and then **New** from the pull-down menu. The Notification Methods Detail window is displayed as shown in Figure 34.

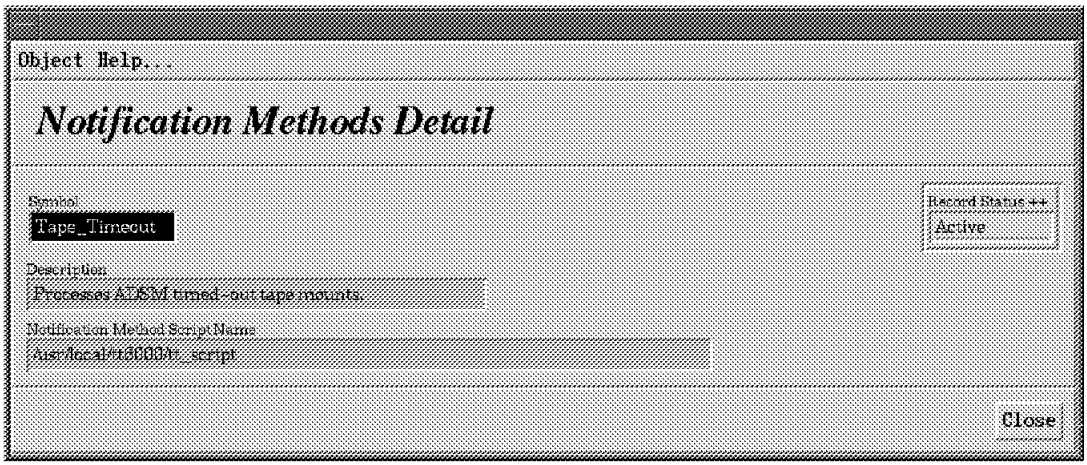


Figure 34. Notification Methods Detail Window

Fill in the fields as follows:

Symbol — A string to identify the item. In the laboratory `Tape_Timeout` was used.

Description — More text to further identify or document the method of notification. In the laboratory this was set to `Processes ASDM timed-out tape mounts`.

Notification Method Script Name — This is the full path name of the shell script that implements the notification method. This is set to the pathname of the shell script that will convert an incoming incident into a trouble ticket. In the laboratory this was set to `/usr/local/tt6000/tt_script`.

On completion, press the **Close** button on the window. This will bring back the Notification Methods window, as shown in Figure 35 on page 145. Don't forget to save the new method at the prompt. Close the Notification Methods window.

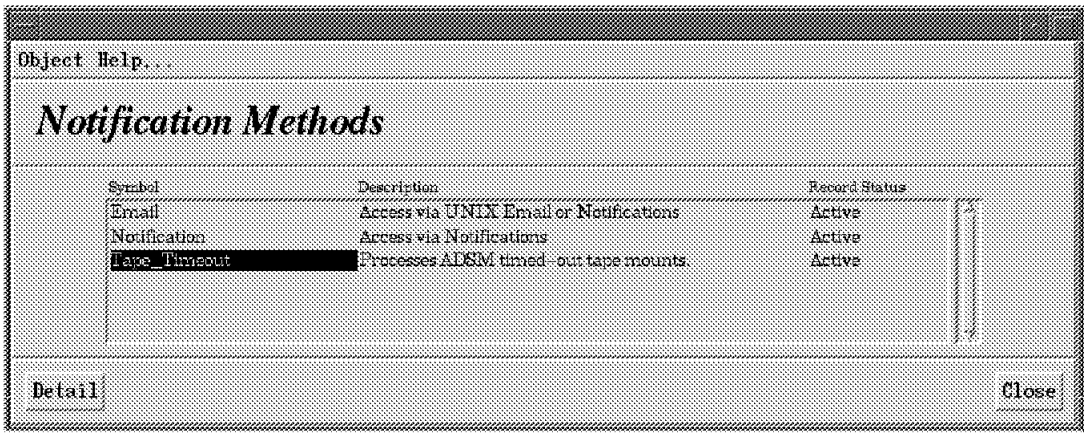


Figure 35. Completed Notification Method Window

The administrator may at this point choose to produce Message Templates and Notification Rules. First, create the Message Template.

From the Problem Management menu bar, take the following options:

1. **Administration**
2. **Notification Setup**
3. **Message Templates**

The Message Template List window will appear; click on **Object** on the menu bar, and select **New** from the pull-down menu. The Message Template Detail window is displayed; it is similar in appearance to the Notification Methods Detail window used earlier. Provide the following information:

1. A Template name

   ADSM_Tape_Timed_Out was used in the laboratory.

2. The Message Header Template

   A number of variables are available to use here; in this example, the value REF_ID - ADSM Tape mount timed out was used.

Close the window, saving changes. The Message Template List window now appears as shown in Figure 36 on page 146.

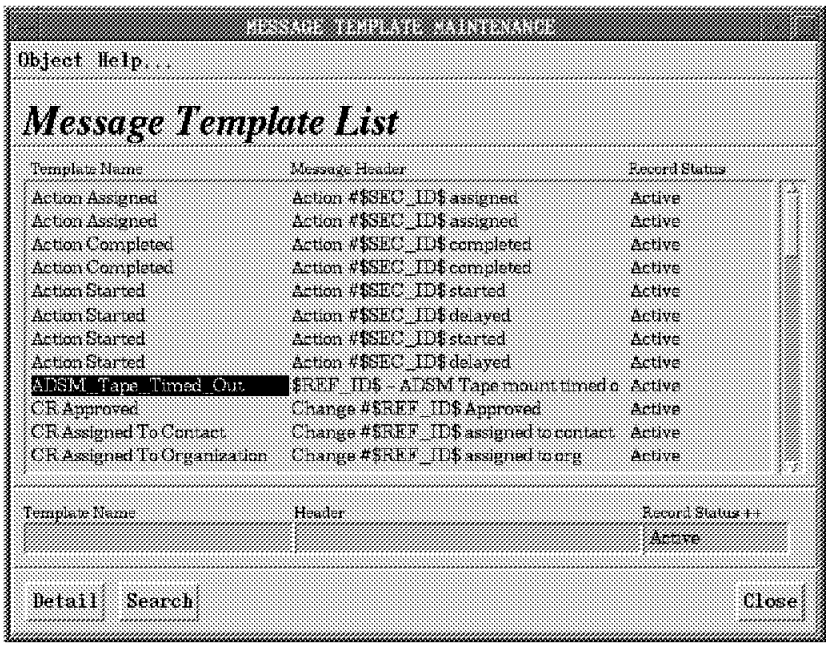


Figure 36. Completed Message Template List Window

Now close the window, and select the following from the Problem Management main menu:

1. **Administration**
2. **Notification Setup**
3. **Notification Rules**

Again select **Object** and then **New** from the menu-bar. The Notification Rule Detail window appears. Set the fields as follows:

Event Triggering Notification — Set to `Incident Report Created` (this may be chosen from the Notification Events List by double-clicking on the field).

Impact Level — This was set to 5 in the laboratory; this is significant for the operation of the /usr/local/tt6000/tt_script sample executable.

Notification Urgency — This was set to `Emergency` in the laboratory.

Where to get Recipient — This must be set to `Specific Script`.

Script Name — This must correspond with the Notification Method defined earlier. In this case, it is `Tape_timeout`.

Message Template Name — Should be set to the message template defined above; in this case, it is `ADSM_Tape_Timed_Out`. Entering this into the input field or selecting it from the list window will also fill in the Message Header field automatically.

Message Text Field — This field is optional.

Figure 37 on page 147 shows the sample Notification Rule Detail created in the laboratory. Close the window, and save the changes.

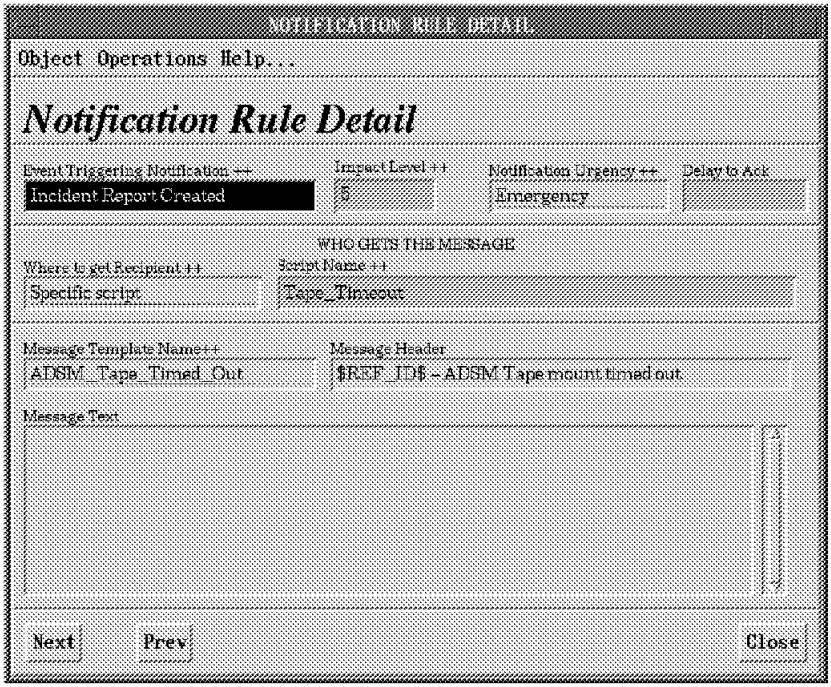


Figure 37. Notification Rule Detail window

When the configuration actions described above have been carried out, Trouble Ticket will automatically generate a trouble ticket from the ADSM_Tape_Mount_Failed trap via NetView. If a test is carried out, the Trouble Ticket Problem Management window will increment the unattached incident analysis counter for impact three on receipt of the event, and shortly after, decrement that count and increment the active ticket analysis count. This signifies that the trouble ticket has been created from the ADSM-generated event.

Chapter 12. NetView DM Scenarios

This chapter looks at several scenarios using NetView DM to distribute, install and configure ADSM.

12.1 Scenario Overview

This scenario will document an example of using NetView DM to package and distribute the ADSM client and server code to machines in a distributed network. This will involve three specific areas:

- Packaging

  The procedures required to package the ADSM client and server code, along with the necessary installation instructions, for distribution.

- Distribution

  The procedures required to configure NetView DM for the distribution of the client and server code.

- Installation/update

  The procedures necessary to actually install or update the client systems with the new ADSM code levels.

The next section looks at how to go about setting up NetView DM for these tasks.

12.2 Method

Regardless of the contents of the distributions that will be effected from NetView DM, the process that needs to be followed is essentially the same. The following steps need to be performed:

- Define targets

  The target systems that will receive (or initiate) distributions need to be defined to the NetView DM server system. These systems can be on the same physical network as the server, a TCP/IP network, for example, in which case they are known as local nodes, or they can be anywhere in a wide area network composed of many interconnected network types, in which case they are known as remote nodes. In addition, the target systems can be of many different operating system types, including:

  - AIX
  - OS/2
  - HP-UX
  - Microsoft Windows
  - DOS
  - SUN OS
  - Sun Solaris

  The process of defining the target nodes is very simple for local nodes. The following definitions must be made:

  1. Target system name

The name by which the target system is known in the network must be provided so that NetView DM can communicate with the system.

2. Target system address

   In some cases (if LAN security is to be used), a LAN address for the target system must be provided.

3. Change management method

   The method of change management that will be allowed must be specified. This can be either server and target initiated (push) or target only (pull).

4. Short target name

   This is used by NetView DM to create a network-wide unique name for the target that is used in the cataloging of change files for the target system.

5. Target operating system

   The operating system of the target needs to be supplied so that NetView DM knows what type of distribution mechanism to employ.

6. Target system users

   Those users on the target system that will be authorized to perform change management operations need to be defined to the server.

The rest of the target definition can be allowed to default, but includes:

1. Target system availability windows

   The windows of availability at the target system for distributions to take place can be defined.

2. Tokens

   Tokens, or variables, can be defined to be used in scripts that are executed at the target at various times during a change cycle. These variables can be used to test certain elements at the target if decisions need to be made during the change cycle. Testing for the existence of certain files, for example.

3. Details

   Information regarding contact personnel at the target can be included in the definition.

4. Logging level

   The level of logging of information during the change cycle can be defined.

5. Target hardware

   Information about the hardware at the target can be supplied although NetView DM is capable of discovering most of this information if required. This is useful if hardware prerequisistes need to be tested prior to any changes being made at the target.

This completes the necessary definitions for target systems at the server. At the targets themselves, the definition is even simpler. If users have been specified as authorized for NetView DM change management activities, then they must be defined to the target operating system. In addition, the server name needs to be defined to the target in the NetView DM configuration file.

All of the rest of the configuration is performed automatically and only needs to be modified in special circumstances.

For remote nodes, the configuration is slightly more complex at the server, as SNA/DS routing information needs to be defined. This is beyond the scope of this book though because the focus here is on the actual creation and distribution of ADSM change files.

- Change file creation

  Now that the targets have been defined, the change files that comprise the distribution elements need to be created. There are two kinds of distribution package:

  – Data file

    A data file is simply a single file that needs to be distributed to targets within the network. It can be text data, microcode, application binaries or indeed any type of file that may need to be centrally managed and distributed.

    This kind of distribution is not appropriate to integration with ADSM as it does not involve installation, only distribution.

  – Change file

    A change file is comprised of installation images and, optionally, other data files that may be needed during the installation process. There are three types of change file:

    - Refresh

      A refresh typically contains a new release of software that will replace existing back-level software or provide a new installation.

    - Update

      An update typically contains new elements of software that provide an update to part of an existing software product.

    - Fix

      A fix typically contains replacements to elements of a software product that are needed to fix problems with the existing code.

    These change file types correspond to the types of software installation processes that can occur on a system. For distribution of ADSM, only the refresh is required as all ADSM distributions, including fixes, are packaged as a complete copy of the product.

    There are two types of refresh operation supported:

    - AIX installp

      This will utilize the standard AIX `installp` process. This method will be used to distribute the ADSM software and requires that images are supplied in installp format.

    - Generic

      This will utilize standard copy functions at the target to install the specified software files.

    In both cases, the files included in the change file can be packaged and shipped from the NetView DM server, or can be resident at a file server or the target itself. The latter cases allow smaller change files to be built

as only pointers to the information are required; the utilization of file servers can also be effected.

The change file must now be defined, requiring the following information:

1. Component Name

   The component name is used to distinguish between change files; it must be unique.

2. Level

   This field represents the new change level; it must be greater than or equal to any previous change levels at the targets.

3. Version

   This field represents the new version and must be greater than any previous versions for this level of software at the targets. This field is optional.

4. Description

   This field allows an optional description of the change file to be provided.

5. File name

   This field contains the file name that the change file will be stored as at the server.

6. Files included

   This dialog is used to specify the images and other files that will comprise the change file.

7. Tokens

   This dialog allows tokens to be included in the change file if required.

8. Change cycle options

   When the change file arrives at a target, the change cycle begins. In its simplest form, the cycle will consist of installing the images supplied with the change file. It is possible to finely control the change cycle by supplying various hardware and software prerequisite conditions, as well as scripts that are to be executed at various points in the change cycle. This dialog allows hardware and software prerequisites to be stipulated and scripts to be specified for execution at the following points in the change cycle:

   a. Request

      Commands can be specified to execute before and after every change management command that operates during this cycle. This option will not be required in these scenarios.

   b. Install

      Scripts can be specified to execute before and after the installation of the change file images. This option will be used to perform pre- and post-installation functions in some of the scenarios.

   c. Accept

      It is possible to specify that the change file images are installed in a removable fashion. In order to commit the change, an accept

operation must be initiated. Scripts can be specified here to operate before and after the accept operation. This function could be used if testing of a new release of software was required and backups of certain information were required, for example. However, this option will not be used in these scenarios and is only supported with installp for the first installation of a product.

d. Remove

If it the change images are to be removed instead of accepted, then this option allows scripts to be executed before and after the removal. This can be useful for restoring backups of information, for example. This option will also not be used in these scenarios and is only supported with installp for the first installation of a product.

e. Uninstall

This option allows scripts to be executed before and after the uninstallation of a change. This option is not supported for installp.

f. Activate

It is possible to specify that change file information be loaded on the target, but no further processing occur until an activation is received. This option is not supported for installp.

In the following scenarios, hardware and software prerequisites will be stipulated and scripts defined for pre- and post-installation processing.

9. Profile

It is possible to use the change file being created as a template for subsequent change files. If this is required, a profile can be created for this purpose.

10. Compression

This dialog allows compression options to be specified for the distribution of the change file.

Two change files will be created in these scenarios, one containing the ADSM server code and one the ADSM client code.

Now that the change files have been created, it only remains to distribute them when required.

- Start NetView DM

  The NetView DM server and client code needs to be started to allow distribution operations to begin.

- Initiate distribution

  There are two ways to initiate distribution. The target system can make the request (pull) or the server system can start the distribution itself (push). In these scenarios, the push method of distribution will be used as the focus here is on central maintenance of ADSM software levels.

  In order to initiate the distribution, the change files required must be selected along with those targets that are to receive them. In this case, the server and client change files will be shipped to ASDM server systems and the client change files shipped only to ADSM client systems.

Once the change files and targets have been selected, the following steps will be performed:

1. Define install options

   At this point, it is possible to stipulate the following distribution options:

   – Install requiring activation

     This option would cause the change file to be transferred to the targets but further processing suspended pending receipt of an activation request from the server. This option is not supported for installp.

   – Install as removable

     As discussed previously, it is possible to install change files so that they can be removed at a later date in case of problems. This option is not be used in these scenarios.

   – Automatically accept install

     The change files will be committed when they are installed in these scenarios, therefore this option is not required.

   – Install as corequisites

     For the installation of an ADSM server, both client and server software is required. For this reason, the two change files created need to be installed as corequisites on a server system with the client package being installed first.

   – Ignore current status of change file

     NetView DM performs various checks to ensure that only valid change files are installed at the target. It is possible to override these checks. This option is not required in these scenarios.

   – Force installation of change file

     Installation of ADSM software refreshes requires overwriting of the previous version of the software. This option must therefore be taken for targets being refreshed.

   – Extend file system

     This option allows NetView DM (or installp) to extend file systems at the target during installation if required.

   Different combinations of these options will be selected in these scenarios depending on the type of installation to be performed.

2. Schedule install

   The actual installation can either be initiated immediately or scheduled to occur at an opportune moment at the target systems. To illustrate the scheduling process, installations are scheduled in these scenarios.

Distribution of the change files to the target systems has now been initiated and the final processing at the targets themselves is all that remains to be done.

- Target installation

  Actual installation at the target systems will be automatic, based upon the way in which the environment will be set up in the scenarios. As such, there

is no further activity to be performed, other than testing to ensure the success of the operations at the targets.

This completes the description of the methodology that is employed to implement ADSM software distribution with NetView DM. To illustrate the process, three scenarios are used:

1. ADSM client and server refresh

   This scenario illustrates an automatic refresh of ADSM client and server software to ADSM server and ADSM client targets.

2. ADSM client installation

   This scenario illustrates installation of ADSM client software at a new client system.

3. ADSM server installation

   This scenario illustrates installation of ADSM server software at a new server system.

The next sections detail these scenarios.

12.3 Integration

This section looks at several scenarios to illustrate the integration of ADSM with NetView DM from the point of view of packaging ADSM client and server software for distribution to systems in a network.

12.3.1 Client and Server Refresh

This scenario looks at packaging the client and server software and distributing it to systems in the network that already have the ADSM client and server software loaded.

- ADSM client systems

  For ADSM client systems, this process involves installing the new release of the client software over the previous release. Prior to the installation, it will be necessary to shut down any ADSM applications that are executing at the client, such as space management or archive and backup clients waiting for scheduled events. After the installation, these functions will need to be restarted.

- ADSM server systems

  For ADSM server systems, this process involves installing the new release of the server software over the previous release. Prior to installation, it will be necessary to halt the server and take backups of critical server components. After the installation, the server will need to be started.

In this scenario, there is one server system and one client system. The backup/archive client is running on the client system, waiting for centrally scheduled operations.

The objective is to install the new releases of the software and leave both client and server operating normally.

12.3.1.1 Packaging the ADSM Software

The first step is to package up the ADSM client and server software along with the pre- and post-installation scripts. This is accomplished as follows:

Packaging the Server: From the NetView DM main window shown in Figure 38, select the following from the menu bar:

- **Catalog**
- **New**
- **Change File**
- **Refresh**

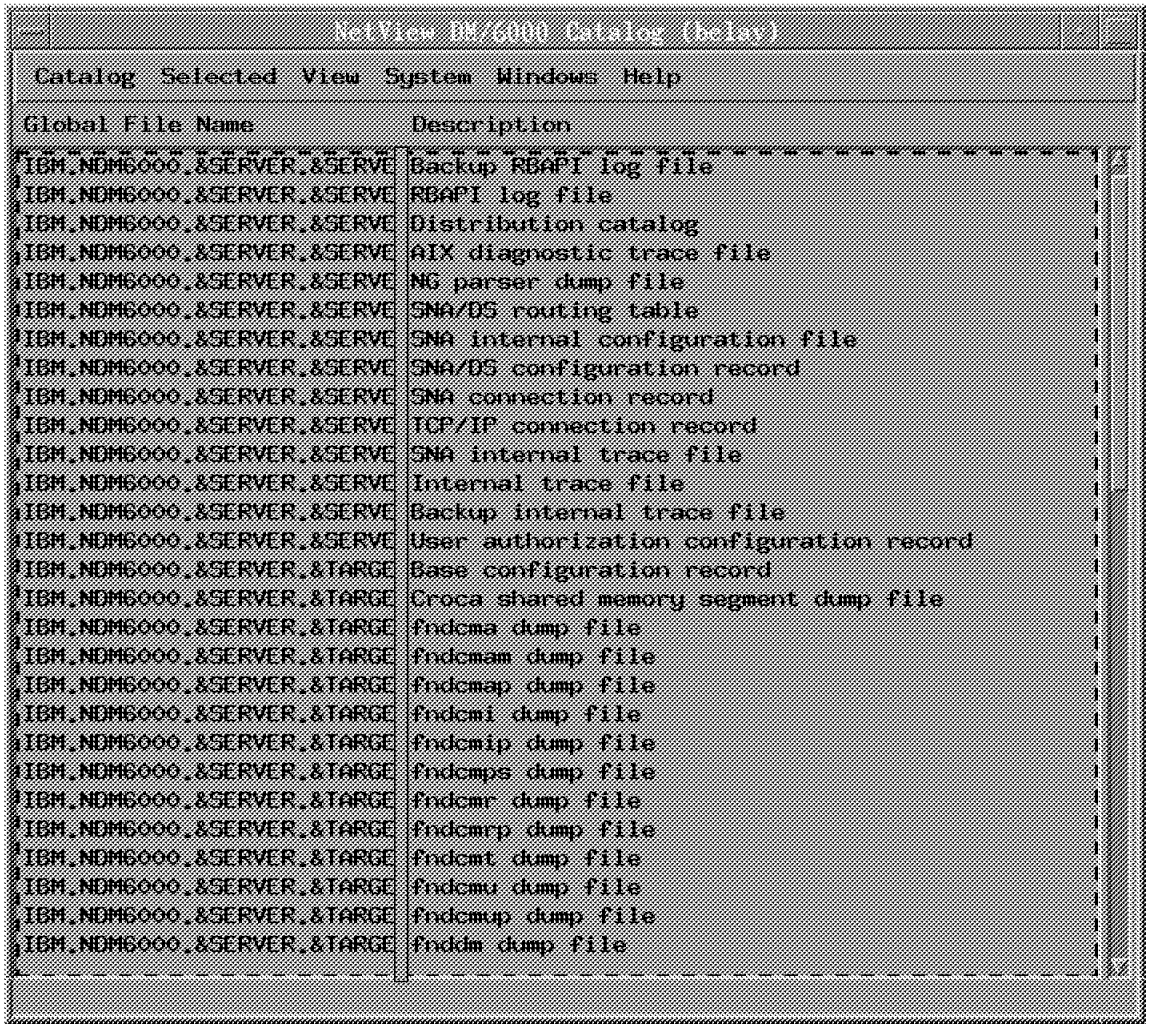


Figure 38. NetView DM Main Window

This produces the Change File Type window shown in Figure 39 on page 157. Select the **AIX installp** option as shown, and press the **OK** button.

Figure 39. Change File Type Window

This will produce the Change File window, as shown in Figure 40. Fill in the fields as shown in this picture. This will define a new component called ADSM_SERVER_REF at Level 2, Version 2 and will save the change file as adsm_v2r1_server_refresh.

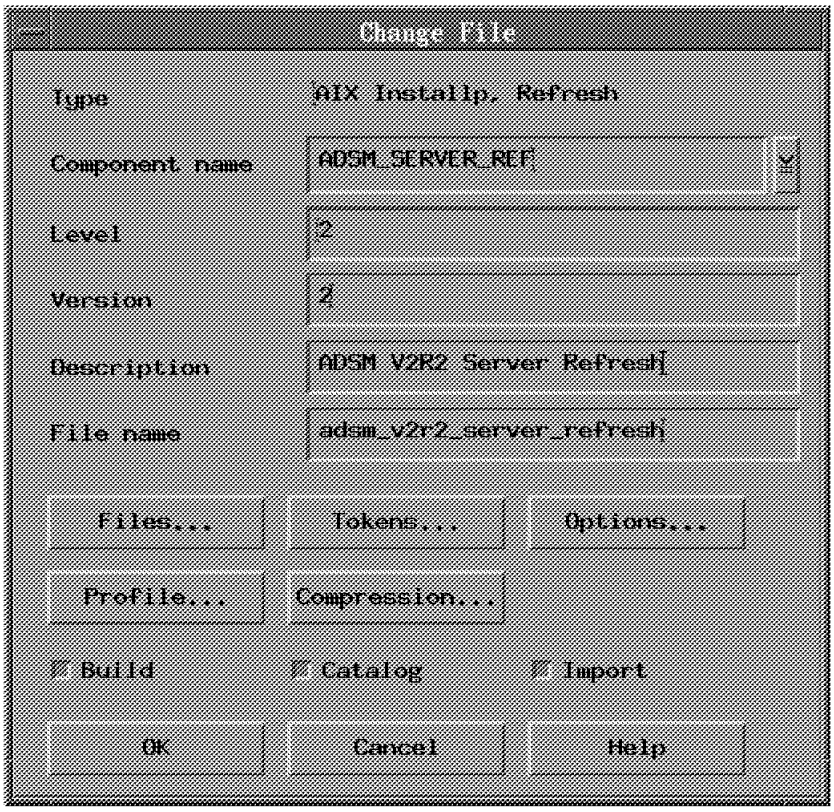


Figure 40. Change File Window

Now, press the **File** button to add the images and scripts that will comprise this refresh. This produces the Files in Installp Change File window as shown in Figure 41 on page 158. Select the image files to be included; in this example, the Version 2 Release 1 install image file has been selected along with a script

file which will be used for post-installation processing. The contents and operation of this script will be described later in this section. The Find button can be used to locate image files and script files for inclusion in the change file.

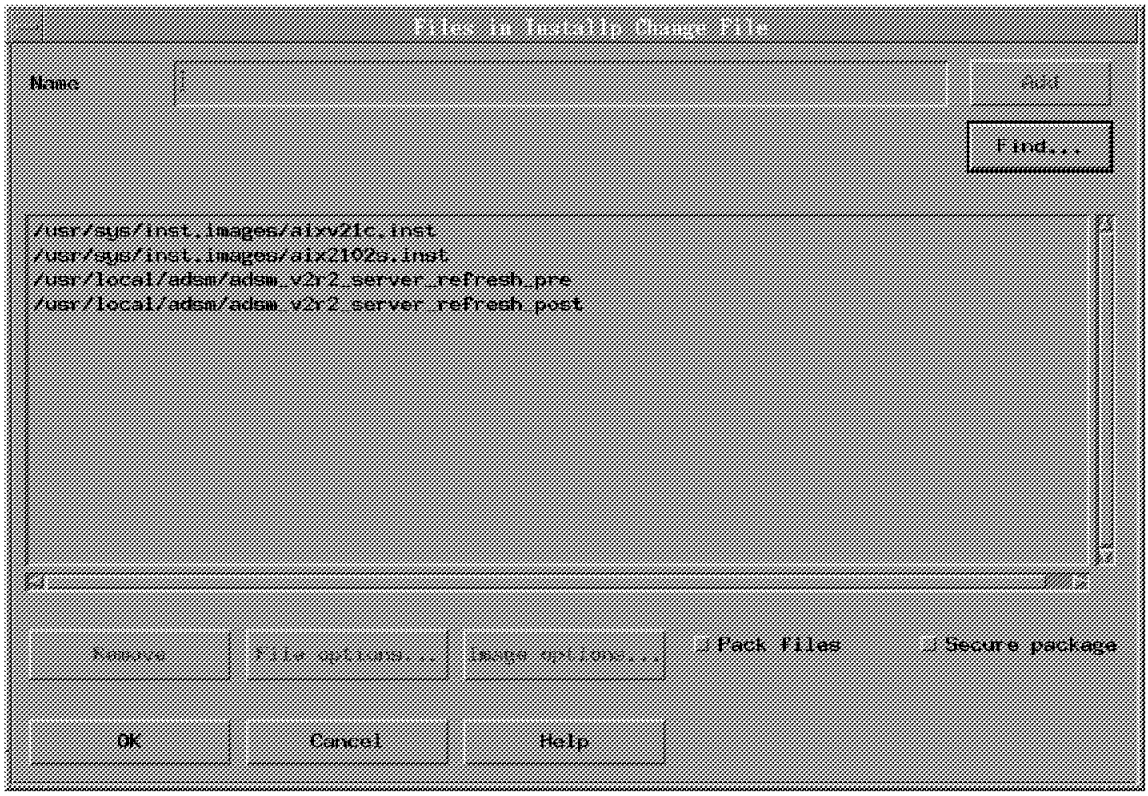


Figure 41. Files in Installp Change File Window

The File options button can be used to define characteristics of script files, such as whether they are physically resident at the server or another system and what name they should be installed under.

The Image options button can be used to select components from within the image to be excluded from the install as well as specifying whether the image file is resident at the server or at another system.

The Pack files and Secure package options can be used to pack the files together and encrypt them respectively.

In this scenario, none of these options are used.

Now, press the **OK** button to complete the file specification. In this scenario, no tokens will be required, and compression will not be used. The use of these options is not directly relevant to this scenario. A profile will not be used to create this Change File either.

The next step is to press the **Options** button. This will produce the Change Management Options window as shown in Figure 42 on page 159. This window allows the hardware and software prerequisites to be stipulated as well as pre- and post-processing script files and commands.

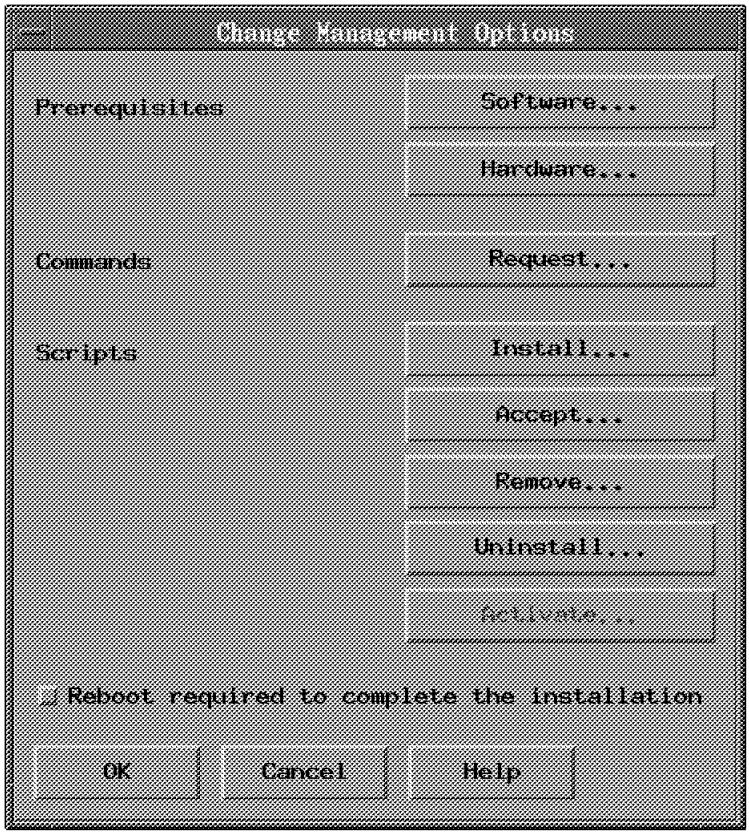


Figure 42. Change Management Options Window

Only the Install option will be required in this scenario since it will be assumed that the hardware and software prerequisites have already been met because there are existing ADSM software levels at the target sites. Pressing the **Install** button produces the Install Scripts window shown in Figure 43. This window allow the scripts that will be run before and after installation to be specified.

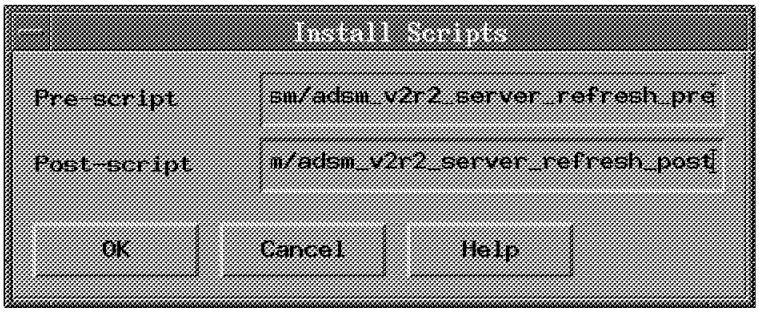


Figure 43. Install Scripts Window

The scripts that were included in the change file are the ones specified here. The scripts are as follows:

- Pre-installation script

  This is the script that is executed prior to the install:

```
#!/bin/ksh
#
# Stop the server and then
# copy any important files
#

/usr/lpp/adsm/bin/dsmadmc -id=admin -pa=admin halt

cp /usr/lpp/adsmserv/bin/dsmserv.opt /tmp
cp /usr/lpp/adsmserv/bin/dsmserv.dsk /tmp
cp /usr/lpp/adsm/bin/dsm.opt /tmp
cp /usr/lpp/adsm/bin/dsm.sys /tmp

exit 0
```

- Post-installation script

  This is the script that is executed after the install:

```
#!/bin/ksh
#
# Copy any important files and then
# restart the server in the background
#

cp /tmp/dsmserv.opt /usr/lpp/adsmserv/bin
cp /tmp/dsmserv.dsk /usr/lpp/adsmserv/bin
cp /tmp/dsm.opt /usr/lpp/adsm/bin
cp /tmp/dsm.sys /usr/lpp/adsm/bin

/usr/bin/nohup /usr/lpp/adsmserv/bin/rc.adsmserv&

exit 0
```

It is possible to instruct NetView DM to reboot the target system after the change cycle is complete. This will not be required in this scenario. The **OK** button can now be pressed. The final options in the Change File window are as follows:

1. Build

   Selecting this option will cause the Change File to be built. This is the default and is required in this scenario.

2. Catalog

   Selecting this option will add the new Change File to the NetView DM catalog. This is required before a Change File can be used.

3. Import

   Selecting this option will cause this Change File definition to be imported into the existing change file specified by the File name field. Select this option.

Press the **OK** button in the Change File window to cause the Change File to be built. The main window will be updated with the new Change File, as shown in Figure 47 on page 164.

Packaging the Client: From the NetView DM main window shown in Figure 38 on page 156, select the following from the menu bar:

- **Catalog**
- **New**

- **Change File**
- **Refresh**

This produces the Change File Type window shown in Figure 39 on page 157. Select the **AIX installp** option as shown, and press the **OK** button.

This will produce the Change File window, as shown in Figure 44. Fill in the fields as shown in this picture. This will define a new component called ADSM_CLIENT_REF at Level 2, Version 2 and will save the change file as adsm_v2r2_client_refresh.

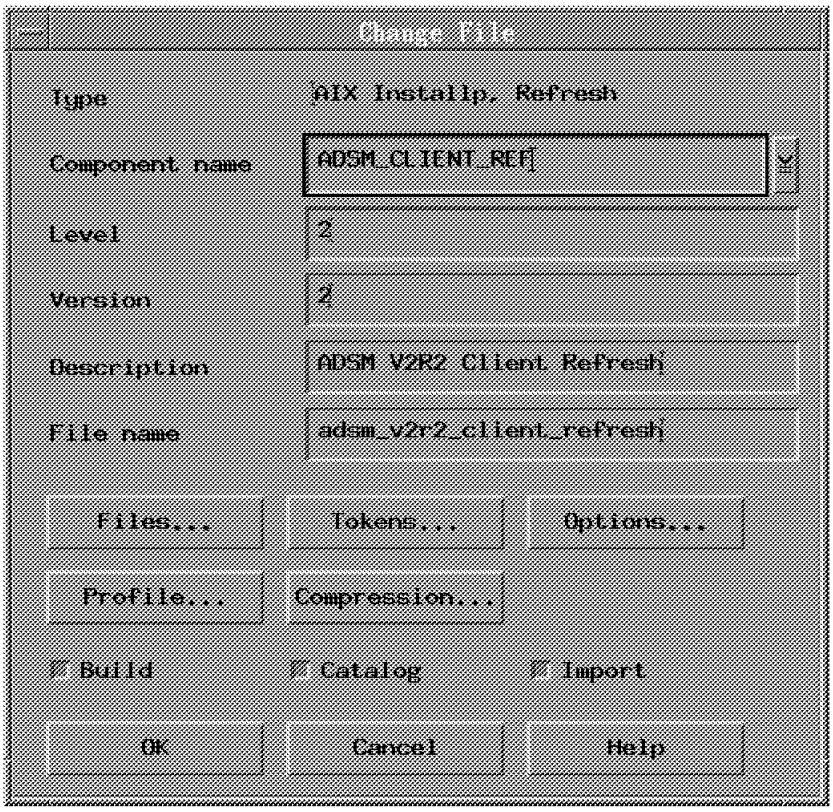


Figure 44. Change File Window

Now, press the **File** button to add the images and scripts that will comprise this refresh. This produces the Files in Installp Change File window, as shown in Figure 45 on page 162. Select the image files to be included; in this example, the Version 2 Release 1 install image file has been selected along with script files which will be used for pre- and post-installation processing. The contents and operation of these script will be described later in this section. Note that the V2R1 install image is used though this has been described as a V2R2 refresh; this is just an example of the process. The Find button can be used to locate image files and script files for inclusion in the change file.

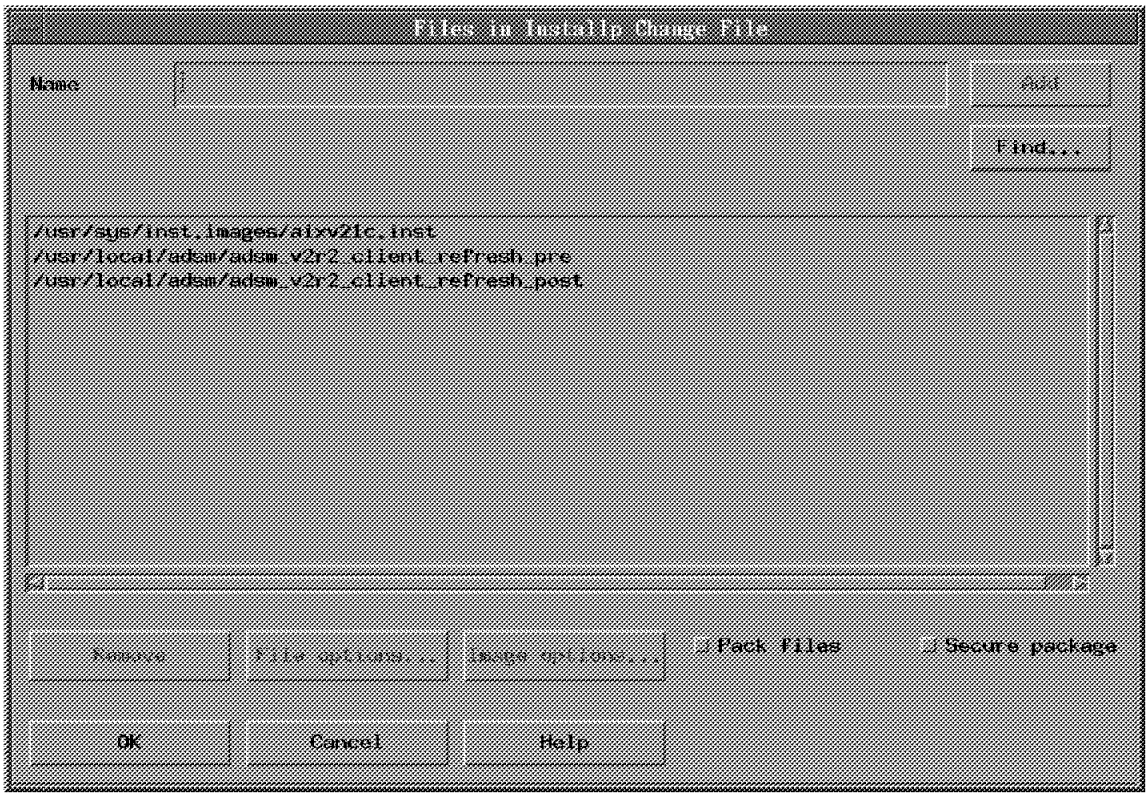


Figure 45. Files in Installp Change File Window

The File options button can be used to define characteristics of script files, such as whether they are physically resident at the server or another system and what name they should be installed under.

The Image options button can be used to select components from within the image to be excluded from the install as well as specifying whether the image file is resident at the server or at another system.

The Pack files and Secure package options can be used to pack the files together and encrypt them respectively.

In this scenario, none of these options are used.

Now, press the **OK** button to complete the file specification. In this scenario, no tokens will be required, and compression will not be used. The use of these options is not directly relevant to this scenario. A profile will not be used to create this Change File either.

The next step is to press the **Options** button. This will produce the Change Management Options window as shown in Figure 42 on page 159. This window allows the hardware and software prerequisites to be stipulated as well as pre- and post-processing script files and commands.

Only the Install option will be required in this scenario since it will be assumed that the hardware and software prerequisites have already been met because there are existing ADSM software levels at the target sites. Pressing the **Install** button produces the Install Scripts window shown in Figure 46 on page 163. This window allow the scripts that will be run before and after installation to be specified.

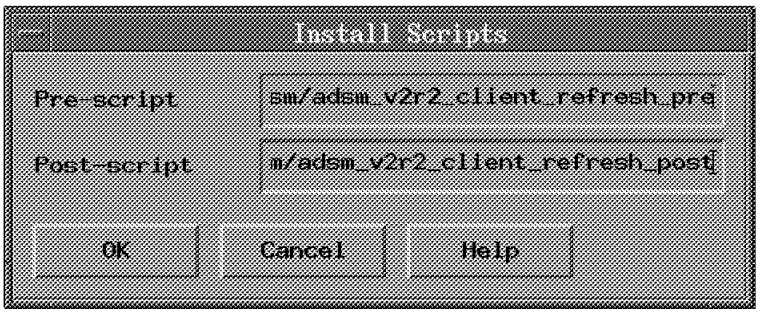


Figure 46. Install Scripts Window

The scripts that were included in the change file are the ones specified here. The scripts are as follows:

- Pre-installation script

  This is the script that is executed prior to the install:

```
#!/bin/ksh
#
#  Kill the old backup/archive client and then
#  copy the configuration files to tmp
#

PID=ps -ef|grep dsmc|tail -1|tr -s ' '|cut -d' ' -f3
kill -9 $PID

cp /usr/lpp/adsm/bin/dsm.opt /tmp
cp /usr/lpp/adsm/bin/dsm.sys /tmp

exit 0
```

- Post-installation script

  This is the script that is executed after the install:

```
#!/bin/ksh
#
#  Restore the copied configuration files and then
#  restart backup/archive client in schedule mode in the background
#

mv /tmp/dsm.opt /usr/lpp/adsm/bin
mv /tmp/dsm.sys /usr/lpp/adsm/bin

/usr/bin/nohup /usr/lpp/adsm/bin/dsmc schedule &

exit 0
```

Again, the reboot will not be required after installation.

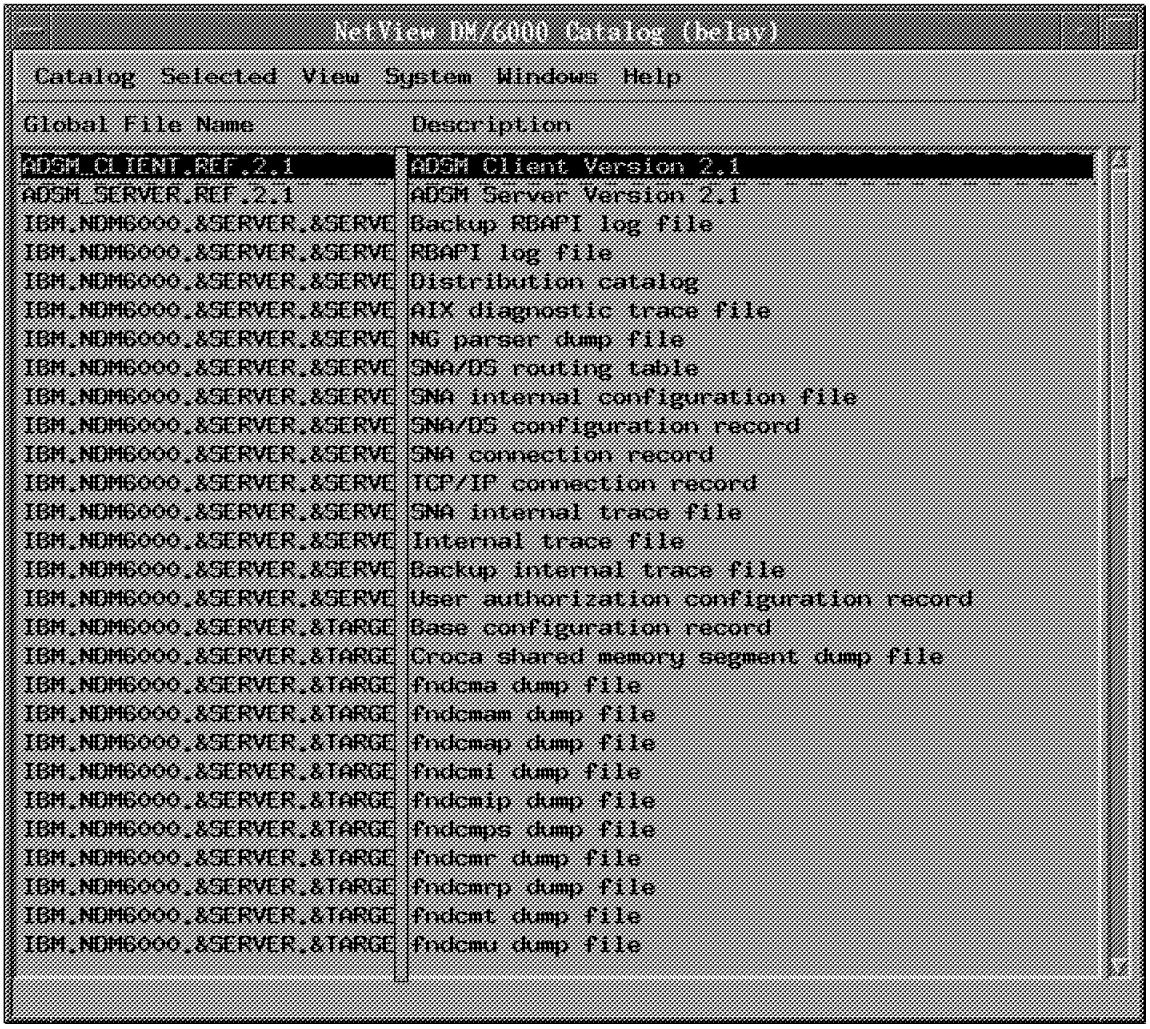


Figure 47. NetView DM Main Window after Change File Creation

The **OK** button can now be pressed. The final options in the Change File window should be left as defaults, as described in the server section (“Packaging the Server” on page 156).

Press the **OK** button in the Change File window to cause the Change File to be built. The main window will be updated with the new Change File, as shown in Figure 47.

12.3.1.2 Distributing the ADSM Software

The next step is to initiate the distribution of the ADSM software to the target systems. This is accomplished as follows:

From the NetView DM main window (shown in Figure 47), highlight the two new Change Files by clicking on them with the mouse, and then select the following from the menu bar:

1. **Selected**
2. **Install**

This will produce the Install Change Files window, as shown in Figure 48 on page 165. From this window, change files can be selected for distribution to the target systems.

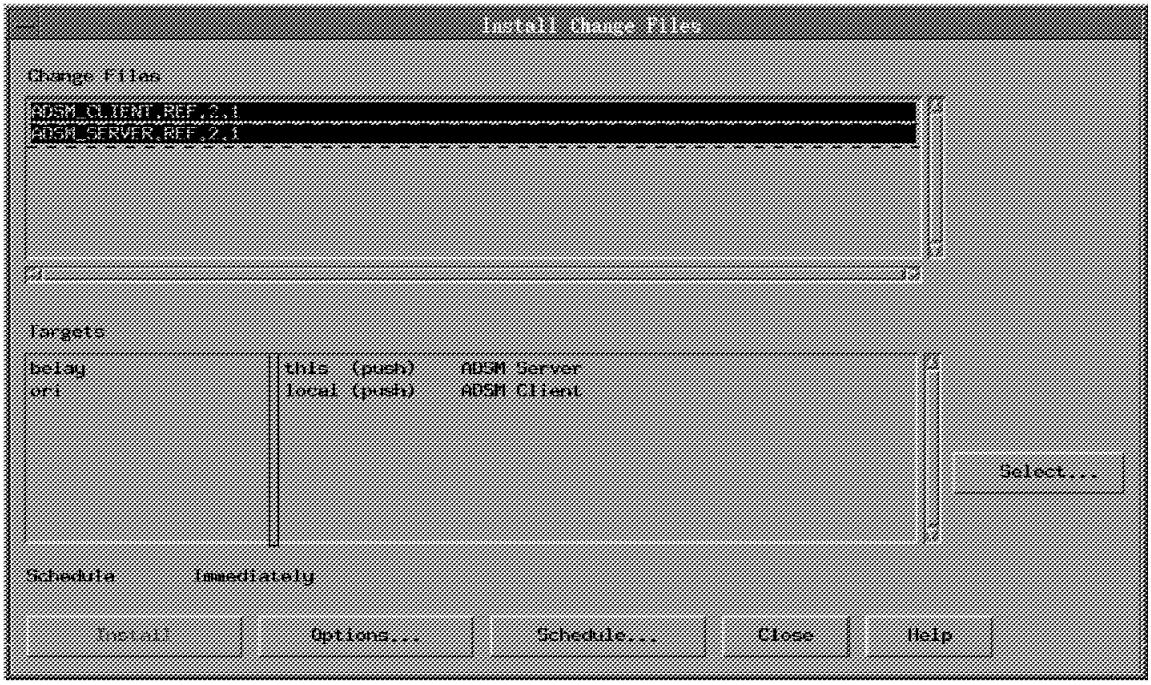


Figure 48. Install Change Files Window

Distributing the Server: Highlight the server and client change files listed in the Change Files section of the Install Changes window by clicking on them with the mouse. Next, highlight the server machines that the distribution is destined for. In this case, the server machine belay.

Since it is important that the change files be installed in the correct order, press the **Options** button. This will produce the Install Options window shown in Figure 49 on page 166. The options listed here have already been described in 12.2, “Method” on page 149; for this scenario, the following options should be selected:

- Install as corequisites

  Both image files are required.

- Extend File System

  Extend the target file systems if more space is required.

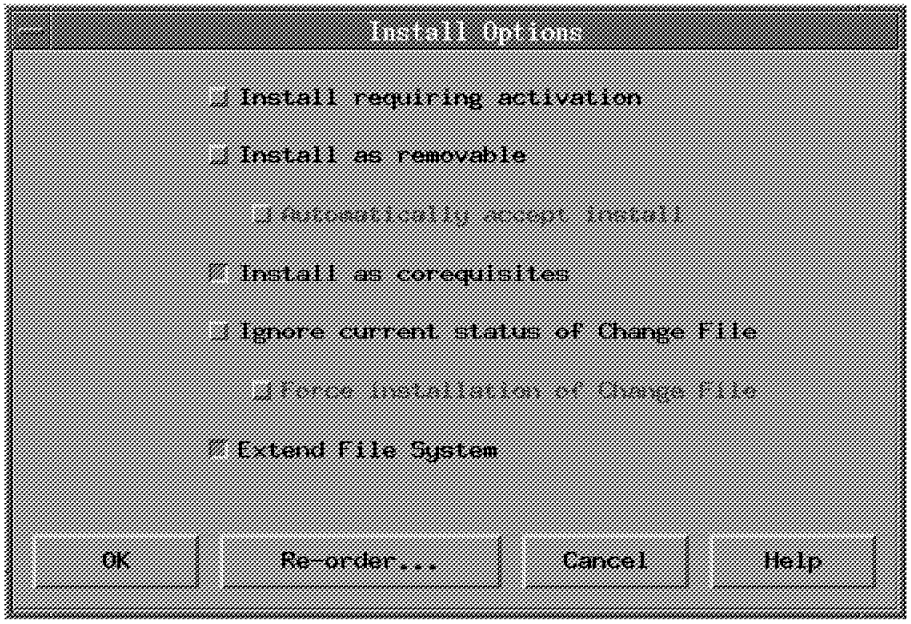


Figure 49. Install Options Window

The rest of the options should be unselected. Next, press the **Re-order** button to check on the order in which the image files will be installed. It is necessary to install the client software before the server software. Once the order is correct, as shown in Figure 50, press the **OK** button in the Re-order Corequisites window and then the **OK** button in the Install Change Files window.

Figure 50. Re-order Corequisites Window

The final step is to schedule the distribution for the required time. Press the **Schedule** button in the Install Change Files window. This will produce the Schedule Time window shown in Figure 51 on page 167, which allows the time of the distribution to be set. Select **Deferred**, and set the time and date fields to the required values, as shown in the figure.

Figure 51. Schedule Time Window

All that remains to be done now is to press the **Install** button to schedule the distribution. This will pop up the Correlator window to confirm the scheduling, as shown in Figure 52. The Install Change Files window can now be closed by pressing the **Close** button.

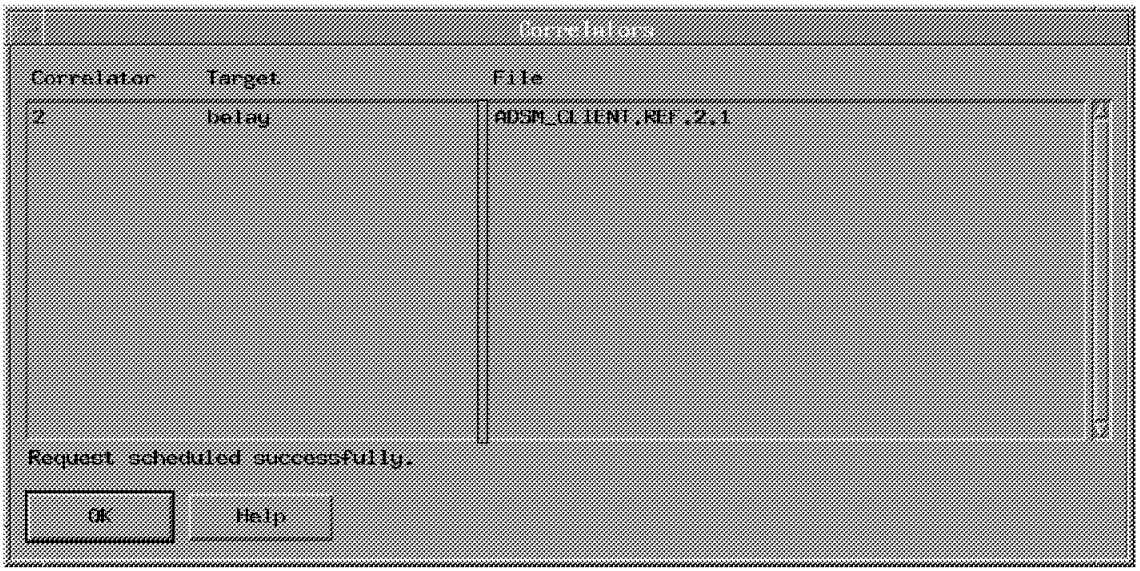


Figure 52. Correlator Window

Pending requests can be displayed as follows. From the NetView DM main window (shown in Figure 38 on page 156), select the following from the menu bar:

- **System**
- **View pending requests**

This produces the Pending Requests window, as shown in Figure 53 on page 168. From here, it is possible to view details of any pending requests and to delete them from the schedule, if required.

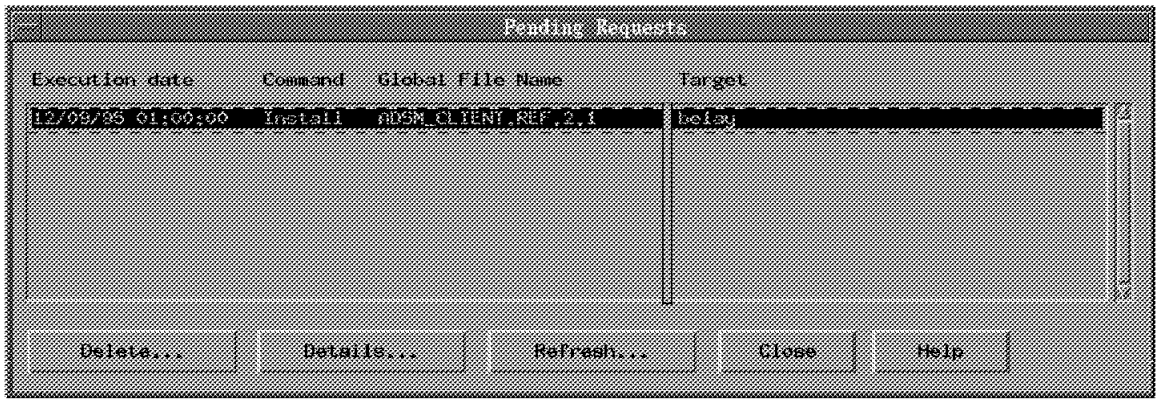


Figure 53. Pending Requests Window

This completes the necessary work to distribute the server software.

Distributing the Client: Highlight the client change file listed in the Change Files section of the Install Changes window by clicking on it with the mouse. Next, highlight the server machines that the distribution is destined for. In this case, the client machine ori.

In order to ensure correct installation at the target systems, press the **Options** button. This will produce the Install Options window shown in Figure 49 on page 166. The options listed here have already been described in 12.2, “Method” on page 149; for this scenario, only the Extend File System option needs to be selected. The rest of the options should be unselected. Now, press the **OK** button in the Install Options window to complete options processing.

The final step is to schedule the distribution for the required time. Press the **Schedule** button in the Install Change Files window. This will produce the Schedule Time window shown in Figure 51 on page 167, which allows the time of the distribution to be set. Select **Deferred**, and set the time and date fields to the required values, as shown in the figure.

All that remains to be done now is to press the **Install** button to schedule the distribution. This will pop up the Correlator window to confirm the scheduling, which will be similar to the one shown in Figure 52 on page 167. The Install Change Files window can now be closed by pressing on the **Close** button.

Pending requests can be displayed as described in 12.3.1.2, “Distributing the ADSM Software” on page 164. This completes the necessary work to distribute the client software.

12.3.1.3 Installation/Update of ADSM Software

The final step is to confirm the successful refresh at the ADSM client and server systems. This can be accomplished by confirming the transfer from the NetView DM message log. From the NetView DM main window (shown in Figure 38 on page 156), select the following from the menu bar:

- **Windows**
- **Message log**

This causes the Message Log window to be displayed, as shown in Figure 54 on page 169. Scroll through the messages to confirm that the requested

distributions were effected successfully. An example message is shown at the bottom of the figure.

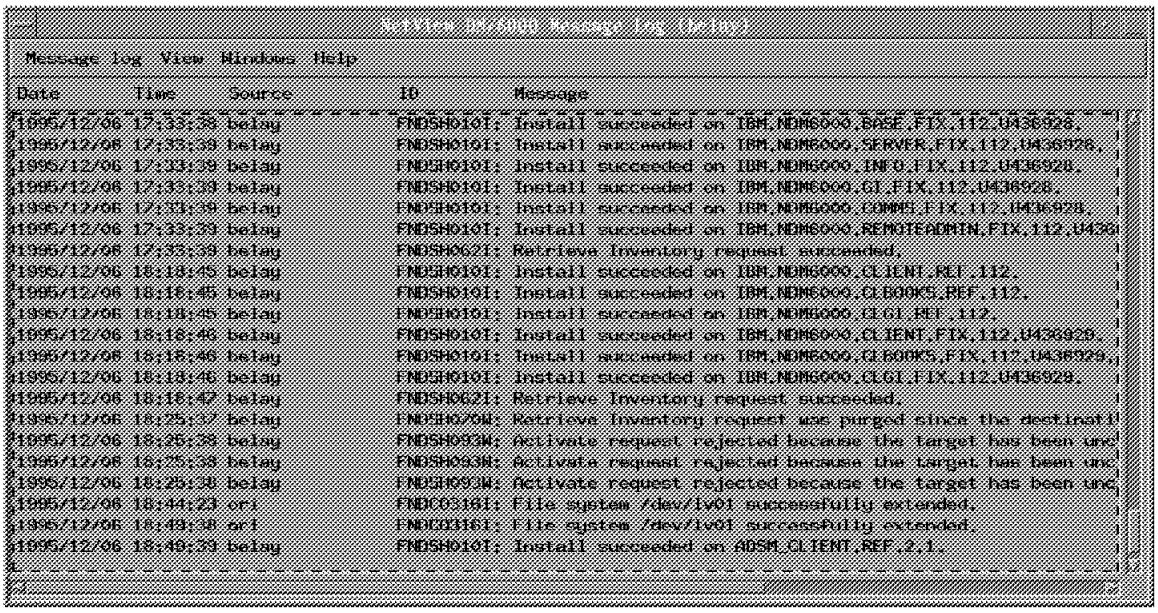


Figure 54. Message Log Window

This completes the first scenario.

12.3.2 New Client Installation

This scenario looks at installing ADSM client software on a new system in the network. This process involves installing the ADSM client along with the necessary configuration of the client system to allow the client to begin functioning. Configuration files need to be distributed with the client software. Subsequent to the installation, the configuration files will need to be copied to their correct locations and any necessary system configuration performed prior to starting the ADSM applications. In this example, the backup/archive client will be installed, configured and executed, operating in the background and waiting for centrally scheduled operations.

The objective is to end up with an operational ADSM client system.

12.3.2.1 Packaging the ADSM Software

The first step is to package the ADSM client software along with the necessary configuration files and post-installation scripts. This is accomplished in exactly the same way as described in “Packaging the Client” on page 160. The main difference will be that only a post-installation script will be required. The Installp Change File window is shown in Figure 55 on page 171. The post-installation script that was used in this scenario is as follows:

```
#!/bin/ksh
#
#  Restore the new configuration files and then
#  restart backup/archive client in schedule mode in the background
#

cat << XXX > /usr/lpp/adsm/bin/dsm.opt
************************************************************************
```

```
* ADSTAR Distributed Storage Manager                                  *
*                                                                     *
* Sample Client User Options file for AIX and SunOS (dsm.opt.smp)     *
***********************************************************************

*  This file contains an option you can use to specify the ADSM
*  server to contact if more than one is defined in your client
*  system options file (dsm.sys).  Copy dsm.opt.smp to dsm.opt.
*  If you enter a server name for the option below, remove the
*  leading asterisk (*).

*  For information about additional options you can set in this file,
*  see the options.doc file in the directory where ADSM was installed.

***********************************************************************

SErvername       BELAY
XXX
cat << XXX > /usr/lpp/adsm/bin/dsm.sys
***********************************************************************
* ADSTAR Distributed Storage Manager                                  *
*                                                                     *
* Sample Client System Options file for AIX and SunOS (dsm.sys.smp)   *
***********************************************************************

*  This file contains the minimum options required to get started
*  using ADSM.  Copy dsm.sys.smp to dsm.sys.  In the dsm.sys file,
*  enter the appropriate values for each option listed below and
*  remove the leading asterisk (*) for each one.

*  If your client node communicates with multiple ADSM servers, be
*  sure to add a stanza, beginning with the SERVERNAME option, for
*  each additional server.

*  For information about additional options you can set in this file,
*  see the options.doc file in the directory where ADSM was installed.

***********************************************************************

SErvername  BELAY
   COMMmethod         TCPip
   TCPPort            1500
   TCPServeraddress   belay.itsc.austin.ibm.COM

Passwordaccess        Generate
SCHEDMODe             PRompted
XXX

/usr/bin/nohup /usr/lpp/adsm/bin/dsmc schedule &

exit 0
```

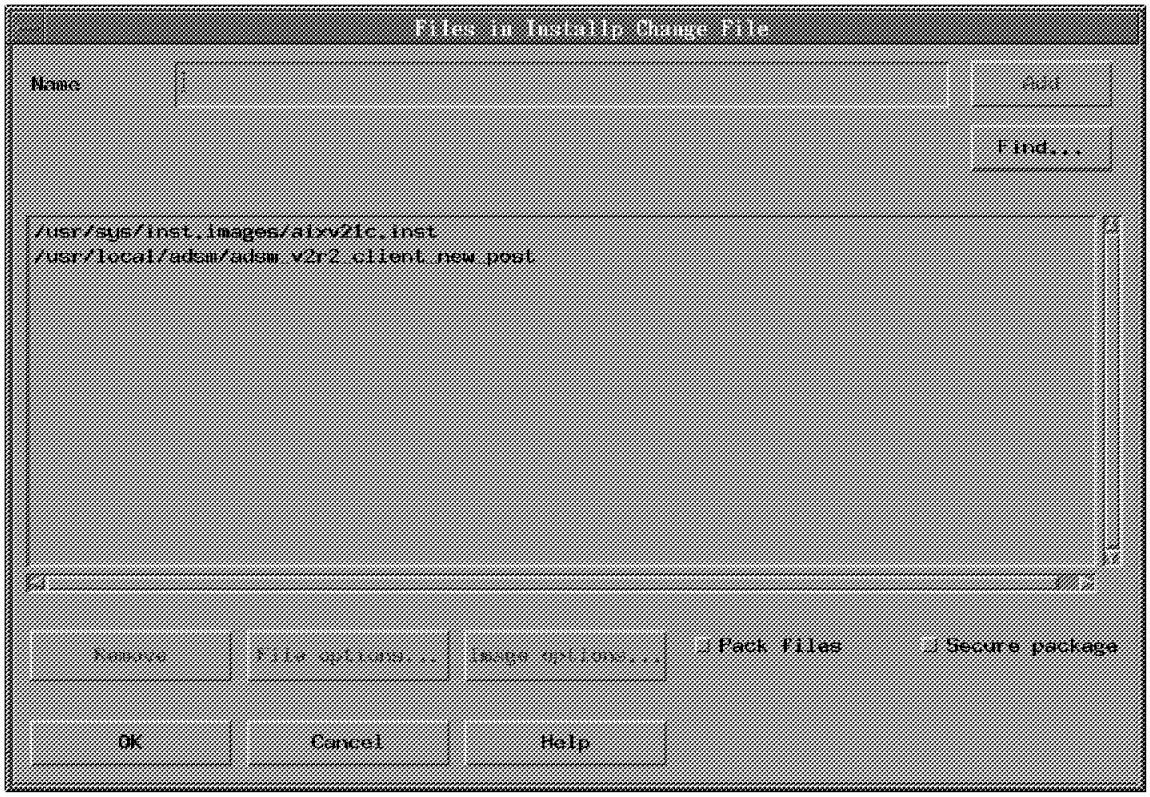


Figure 55. Files in Installp Change File Window

Note that the customized dsm.opt and dsm.sys files are installed from the post-installation script itself.

12.3.2.2 Distributing the ADSM Software

The next step is to initiate the distribution of the software to the target system. This is accomplished in exactly the same fashion as was described in “Distributing the Client” on page 168.

12.3.2.3 Installation/Update of ADSM Software

The final step is to confirm the successful operation of the new ADSM client system. This can be effected as described in 12.3.1.3, “Installation/Update of ADSM Software” on page 168.

This completes the second scenario.

12.3.3 New Server Installation

This scenario looks at installing ADSM server software on a new on a new system in the network. This process involves installing the ADSM client and server software, along with the necessary configuration of the server system and control files to allow the server to begin functioning. Configuration files will need to be distributed with the server software and copied to the correct directories after installation. In addition, the necessary server database, recovery log and storage pool formatting and definitions will need to be made, along with the client definitions. The devices that the server will use also need to be configured on the server system, and then finally, the server needs to be started.

12.3.3.1 Packaging the ADSM Software

The first step is to package the ADSM server software along with the necessary configuration files and post-installation scripts. This is accomplished in exactly the same way as described in “Packaging the Server” on page 156. The main difference will be that only a post-installation script will be required. The Installp Change File window is shown in Figure 56 on page 174. The post-installation script that was used in this scenario is as follows:

```
#!/bin/ksh
#
#  Create client opt file
#  Create client sys file
#  Create server opt file
#  Define the tape devices
#  Format any volumes
#  Create macro for server to perform the following:
#    Define library and tape device
#    Storage pool creation
#    Define nodes
#  Start server in install mode
#

cat << XXX > /usr/lpp/adsm/bin/dsm.opt
*****************************************************************************
* ADSTAR Distributed Storage Manager                                        *
*                                                                           *
* Sample Client User Options file for AIX and SunOS (dsm.opt.smp)           *
*****************************************************************************

*  This file contains an option you can use to specify the ADSM
*  server to contact if more than one is defined in your client
*  system options file (dsm.sys).  Copy dsm.opt.smp to dsm.opt.
*  If you enter a server name for the option below, remove the
*  leading asterisk (*).

*  For information about additional options you can set in this file,
*  see the options.doc file in the directory where ADSM was installed.

*****************************************************************************

SErvername       ORI
XXX
cat << XXX > /usr/lpp/adsm/bin/dsm.sys
*****************************************************************************
* ADSTAR Distributed Storage Manager                                        *
*                                                                           *
* Sample Client System Options file for AIX and SunOS (dsm.sys.smp)         *
*****************************************************************************

*  This file contains the minimum options required to get started
*  using ADSM.  Copy dsm.sys.smp to dsm.sys.  In the dsm.sys file,
*  enter the appropriate values for each option listed below and
*  remove the leading asterisk (*) for each one.

*  If your client node communicates with multiple ADSM servers, be
*  sure to add a stanza, beginning with the SERVERNAME option, for
*  each additional server.
```

```
*  For information about additional options you can set in this file,
*  see the options.doc file in the directory where ADSM was installed.

************************************************************************

SErvername  ORI
   COMMmethod         TCPip
   TCPPort            1500
   TCPServeraddress   ori.itsc.austin.ibm.COM

Passwordaccess        Generate
SCHEDMODe             PRompted
XXX

cat << XXX > /usr/lpp/adsmserv/bin/dsmserv.opt
*  ====================================================================
*  ADSTAR Distributed Storage Manager
*  Server Sample Options - Version 2, Release 1
*  5765-564 (C) Copyright IBM Corporation, 1990, 1995, All Rights Reserved
*  ====================================================================
*   ADSTAR Distributed Storage Manager (ADSM):
*   Sample Server Options File (dsmserv.smp)
*   Platform: AIX
*
*   COMMmethod NONE
*     TCPPort 1500
*     TCPWindowsize 16
*     MSGINTerval 1
*     MAXSessions 25
*     BUFPoolsize 512
*     LOGPoolsize 128
*     COMMTimeout 60
*     IDLETimeout 15
*     LANGuage AMENG
*     DATEformat 1
*     TIMEformat 1
*     VOLUMEHistory <filename>
*     DEVCONFig <filename>
XXX

D=mkdev -c adsmtape  -t 'ADSM-SCSI-MT' -s 'scsi' -p 'scsi1' -w '40'
DEV="/dev/echo $D|cut -d' ' -f1"

/usr/lpp/adsmserv/bin/dsmfmt -m -data /usr/lpp/adsmserv/bin/arch00.dsm 5
/usr/lpp/adsmserv/bin/dsmfmt -m -data /usr/lpp/adsmserv/bin/back00.dsm 5

echo "define library manlib libtype=manual" > /usr/lpp/adsmserv/bin/INSTALL.NEW
echo "define drive manlib drive01 device=$DEV" >> /usr/lpp/adsmserv/bin/INSTALL.NEW
cat << XXX >> /usr/lpp/adsmserv/bin/INSTALL.NEW
define devclass 8mmtape devtype=8mm library=manlib
define stgpool tapepool 8mmtape
define stgpool arch disk next=tapepool
define stgpool back disk next=tapepool
define volume archive /usr/lpp/adsmserv/bin/arch00.dsm
define volume backup /usr/lpp/adsmserv/bin/back00.dsm
register node belay belay
XXX

/usr/lpp/adsmserv/bin/dsmserv install 1 /usr/lpp/adsmserv/log.dsm \
```

```
                                        1 /usr/lpp/adsmserv/db.dsm

exit 0
```

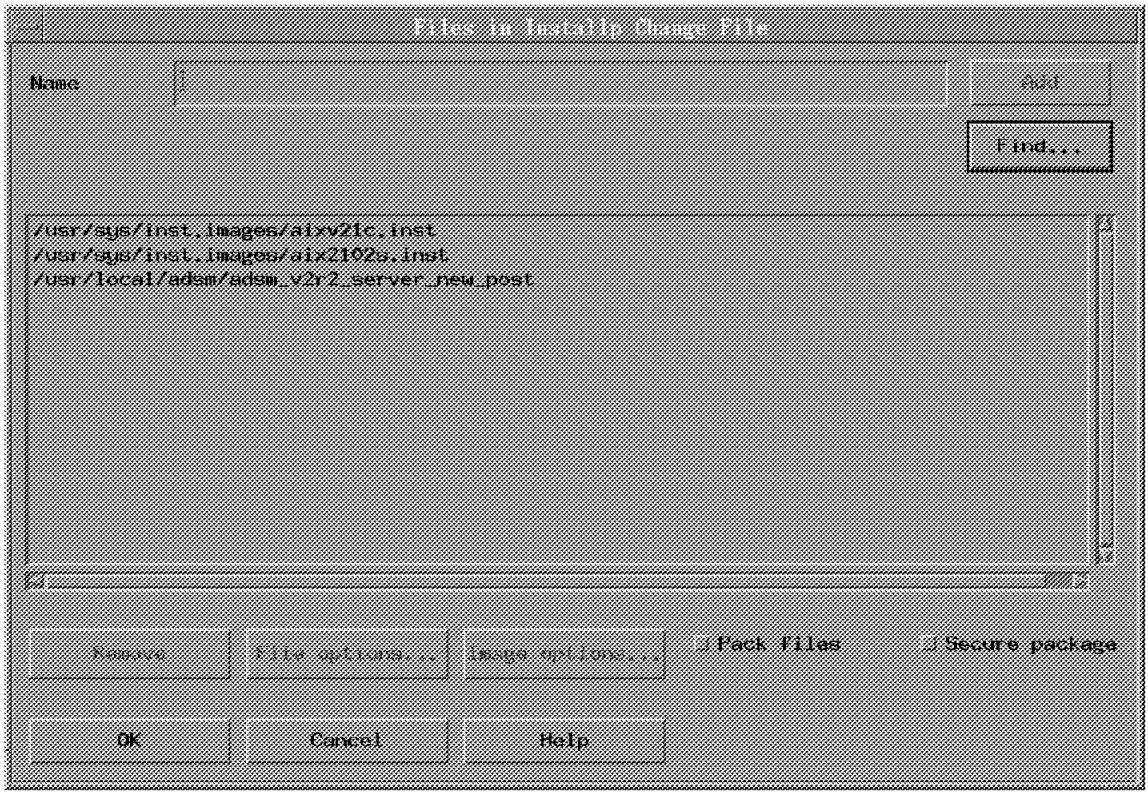


Figure 56. Files in Installp Change File Window

This script will perform the necessary installation and create customized versions of the dsm.opt, dsm.sys and dsmserv.opt files. It will also configure a tape device and format some volumes for ADSM storage pools. A macro is created that contains some basic definitions for storage devices and nodes.

In order to complete the configuration, it will be necessary to log in to the new system, start the ADSM server and execute the macro.

An alternative method would be to create an ADSM server with a standard configuration and backup the database. This database could then be used as the basis for new configurations and would avoid the need for logging in to each system. This is required in this scenario because it is not possible to define an administrator with the necessary authority to run the macro without first logging in to the server in console mode.

12.3.3.2 Distributing the ADSM Software

The next step is to initiate the distribution of the software to the target system. This is accomplished in exactly the same fashion as was described in “Distributing the Server” on page 165.

12.3.3.3 Installation/Update of ADSM Software

The final step is to confirm the successful operation of the new ADSM client system. This can be effected as described in 12.3.1.3, “Installation/Update of ADSM Software” on page 168.

This completes the second scenario.

12.4 Summary

The extent to which ADSM can be integrated with NetView DM is essentially limited to distribution of ADSM Software. NetView DM is designed to provide a comprehensive distribution service, and as is shown by these scenarios, it is possible to distribute and maintain products requiring far more complex installation and refresh conditions than required here.

The most complex part of the process is the design of the script files which perform the pre- and post-installation processing. It is here that the level of integration is really defined. It would be possible to perform many more functions in these scripts, for example performing more complex checks and configuration work. The level to which this is done depends on the requirements of the particular environment.

Lastly, once integration with NetView DM is implemented, change control is also implemented. This will ensure consistency of software levels across the enterprise and thereby simplify not only future change management but also assist in problem management.

List of Abbreviations

ADSM	ADSTAR Distributed Storage Manager
AIX	Advanced Interactive eXecutive
DRM	Disaster Mecovery Manager
DSMIT	Distributed System Management Interface Tool
ESE	Event Stream Enhancements
IBM	International Business Machines Corporation
ITSO	International Technical Support Organization
LAN	Local Area Network
LPP	Licensed Program Product
MB	MegaByte
MIB	Management Information Base
PTX	Performance Toolbox
RMON	Remote Network Monitoring
NetView DM	NetView Distribution Manager
SCSI	Small Computer System Interface
SMIT	System Management Interface Tool
SNA/DS	System Network Architecture/Distribution Services
SNMP	Simple Network management Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol

Index

A

B

C

D

E

F

G

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

ITSO Technical Bulletin Evaluation RED000

International Technical Support Organization
ADSM for AIX: Advanced Topics
December 1995

Publication No. SG24-4601-00

Your feedback is very important to help us maintain the quality of ITSO Bulletins. **Please fill out this questionnaire and return it using one of the following methods:**

- Mail it to the address on the back (postage paid in U.S. only)
- Give it to an IBM marketing representative for mailing
- Fax it to: Your International Access Code + 1 914 432 8246
- Send a note to REDBOOK@VNET.IBM.COM

Please rate on a scale of 1 to 5 the subjects below.
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction	____		
Organization of the book	____	Grammar/punctuation/spelling	____
Accuracy of the information	____	Ease of reading and understanding	____
Relevance of the information	____	Ease of finding information	____
Completeness of the information	____	Level of technical detail	____
Value of illustrations	____	Print quality	____

Please answer the following questions:

a) If you are an employee of IBM or its subsidiaries:
   Do you provide billable services for 20% or more of your time? Yes____ No____
   Are you in a Services Organization? Yes____ No____

b) Are you working in the USA? Yes____ No____

c) Was the Bulletin published in time for your needs? Yes____ No____

d) Did this Bulletin meet your needs? Yes____ No____
   If no, please explain:

_____

_____

What other topics would you like to see in this Bulletin?

_____

_____

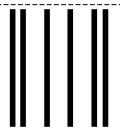
What other Technical Bulletins would you like to see published?

_____

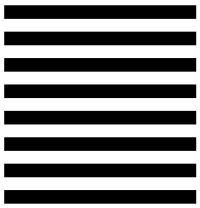
Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

Name	Address
Company or Organization	
Phone No.	

Fold and Tape **Please do not staple** Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM International Technical Support Organization
Department JN9B, Building 045
Internal Zip 2834
11400 BURNET ROAD
AUSTIN TX
USA 78758-3493

Fold and Tape **Please do not staple** Fold and Tape

This soft copy for use by IBM employees only.

Printed in U.S.A.