ServiceCenter ®

SCAuto for

NetView for AIX

October, 14, 1997

Product Author: Toby Torrento
Technical Documentation: Cyril R. Vidergar
Developer(s): bebe Brown, Darin Johnson

Table of Contents

Appendix B IPAS Conversion

Index

Chapter 1 Introduction

SCAuto for NetView for AIX™, from Peregrine Systems, Inc., is a problem reporting and automated network inventory system. SCAuto for NetView is designed to enhance ServiceCenter™ Problem and Inventory Management applications. SNMP (Simple Network Management Protocol) traps from NetView are opened as ServiceCenter problem tickets, and NetView objects added to ServiceCenter as inventory records. Inventory for IP, SNA, and Novell networks can be maintained.

SCAuto for NetView is composed of component programs that handle inventory and problem reporting automation, as well as facilities to integrate your ServiceCenter applications into a network manager's operational environment. SCAuto for NetView utilizes supplied NetView APIs and standard facilities to operate.

SCAuto for NetView is an SCAutomate client application, and comunicates with ServiceCenter via the SCAutomate server. The SCAutomate protocol uses TCP/IP. This allows connections to ServiceCenter from the remote platforms, including those on which no component of ServiceCenter has been installed.

SCAuto for NetView major components are:

- A discovery program that creates ServiceCenter inventory.
- A trap monitor that gathers and filters NetView traps.
- An event monitor that creates problem tickets and inventory updates from the information from the trap monitor.
- Integration with NetView.
- Utility programs.

Topology Management Overview

The primary function of the topology components is to create and maintain inventory records in ServiceCenter. The inventory records created correspond to the objects discovered by NetView and additional programs such as OpenSNA, or StationView and ServerView. Each record is maintained as a ServiceCenter device record, with special fields for connection and control information. The connection data is important to graphical, path determination, outage analysis, and dependency propagation applications. The connection relationships maintained are:

- Container (contained in relationship)
- Hierarchical (parent, child relationship)

SCAuto for NetView topology management transforms any of the NetView objects into generic device types in ServiceCenter.

ServiceCenter inventory records also contain various inventory fields: device name, type, mode, serial, vendor, location address, contact information, hardware adaptor address, protocol, and description information. See the tables later in this chapter for particular inventory types generated and their associated ServiceCenter fields. The amount of information supplied is dependent on the platform that you are running and the information available from the installed network manager programs.

IP Network Inventory

IP (Internet Protocol) inventory is discovered via the NetView IP map. The major ServiceCenter inventory device types created for IP networks are:

- Network (specific network)
- NetworkSegment (section of network)
- Node (IP host)
- Interface (for each IP adaptor in a node).

Figure 1-3 provides a hierarchial view of these device types.

The IP environment normally provides specific configuration information from the MIB (Management Information Base). If MIB information is maintained through network administration, ServiceCenter inventory records can be updated automatically. Inventory can be dynamically added, updated, and deleted by responding to NetView traps. The dynamic updates ensure the IP network is up to date and the current status is reflected in your ServiceCenter files.

IP inventory information is obtained through the NetView *ovtopodump* command.

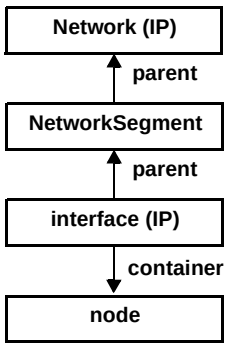

Figure 1-1 IP Topology Types and Relationships

SNA Network Inventory

The SNA inventory is obtained from the OpenSNA object database. These additional types are created from SNA inventory:

- Network (SNA network)
- Domain (section of Network)
- Subarea (sub addressable section of Network)
- Host (MVS host)
- Group (group designator)
- CDRMseg (cross domain manager group)
- CDmgr (cross domain manager)
- CDRSC (cross domain resource group)
- CDresource (cross domain resource)
- Switched (switched LU)
- NCP (network controller)
- Line (individual line)
- Link (NCP to NCP connection)
- PU (SNA Physical Unit)
- LU (SNA Logical Unit)
- APPLS (application group)
- Appl (application designator)
- NonSNA (local nonSNA terminals)
- Direct (director for nonSNA terminal)
- LocalSNA (local SNA terminals).

Figure 1-2 provides a hierarchial view of these device types.

OpenSNA will issue SNMP traps whenever new objects are discovered, deleted, or change. The SCAuto for NetView event monitor processes these events and updates ServiceCenter inventory accordingly.

SNA inventory is obtained through the OpenSNA *sna_print* command.

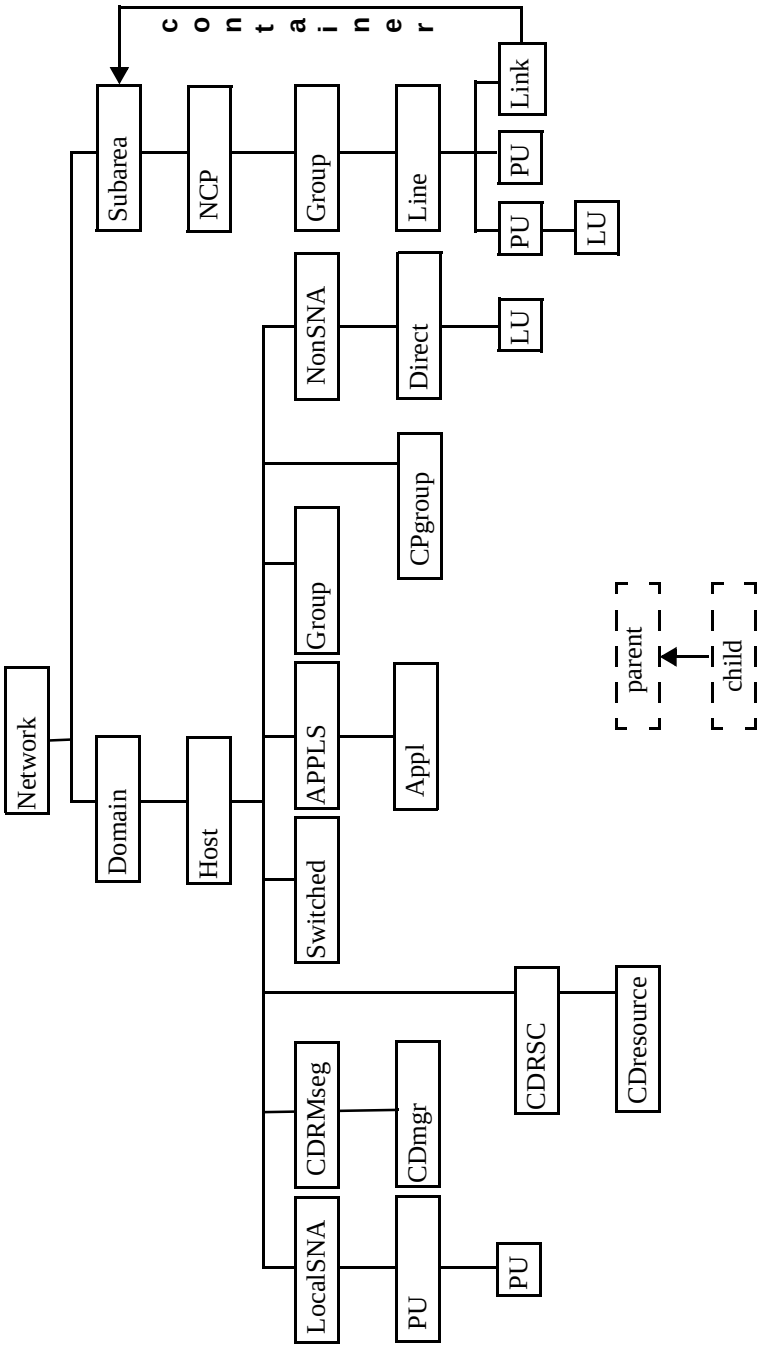

Figure 1-2 SNA Topology Types and Relationships

Novell Network Inventory

The Novell inventory is obtained from the StationView and ServerView products. Additional types created when Novell inventory is discovered are:

- Network (Novell IPX network)
- Workstation (PC workstation)
- Server (Novell server)
- Interface (Novell IPX interface)

Figure 1-3 provides a hierarchial view of these device types.

The Novell inventory is obtained from the StationView and ServerView products. The inventory will not be updated automatically, but can be refreshed periodically to maintain the current Novell network configurations.

Novell inventory is obtained through the ServerView *siget* command.

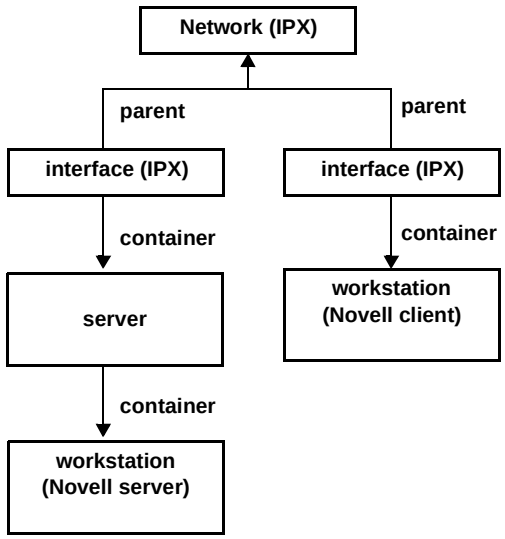

Figure 1-3 Novell Topology Types and Relationships

Novell Software Inventory

SCAuto for NetView also has the capability of gathering station software inventory fromServerView. These use the **icmswa** and **icmswd** ServiceCenter events.

ServiceCenter Event Integration

SCAuto for NetView utilizes standard inventory add, update, and delete events as described in the ***Event Services Guide***. The Event Manager is the ServiceCenter component which maps the input events and gives control to the RAD applications that process the events. In the case of inventory, an *icma* (inventory control

management add) standard event is created and placed in the ServiceCenter *eventin* file. The event scheduler reads the *eventin* file and maps the ServiceCenter event data into the device and attribute files. The scheduler then performs a background inventory add operation. Refer to the ***Event Services Guide*** for more information on the Event Manager and its standard facilities.

A firm understanding of the ServiceCenter Event Manager is helpful in reviewing any of the subsequent tables which map various network components to ServiceCenter inventory records. The tables provide a quick reference of SCAuto for NetView generated fields and their corresponding Event Manager eventmap fields in ServiceCenter.

Problem Management Overview

The Problem Management component of SCAuto for NetView is comprised of a pair of programs that interface with NetView to dynamically open, close and update ServiceCenter problems. Problem actions are based on SNMP traps which are issued to notify NetView of a specific event for a device or software. An event often indicates a problem. SCAuto monitors these traps and turns them into ServiceCenter problem tickets. Since there are a large number of traps generated from a typical network, a user-specified *keywords* file is used as a filter so that only traps deemed important are processed by SCAuto.

Specialized or global filters may also be defined from within ServiceCenter to block problem reporting. Filtering can be based on time of day, event type, event data, frequency of occurrence, and thresholds. If a filtering requirement is not solved by keywords or standard filter specifications, user-written RAD functions or expressions can be used.

SCAuto for NetView forwards problem reporting information to the ServiceCenter Event Manager as ***pmo*** (problem management open) and ***pmc*** (problem management close) events. The Event Manager passes the event data through the standard Problem Management application to open, update or close a problem.

The Problem Management application opens only one problem ticket for each device for traps reported by SCAuto. All subsequent traps received for the same device are considered an update or close to the existing ticket.

For example, an SNMP trap is received by NetView that indicates a possible problem with a device. SCAuto receives this trap from NetView. If the trap is not filtered out by the keywords file, SCAuto tries to create a problem open event (pmo) for the device in ServiceCenter. ServiceCenter Event Manager filters will be applied to the problem open event, and the event will be created or discarded. If the problem is created, the Problem Management application will create a new problem ticket.

If a subsequent trap arrives for the same device, the same sequence of events occur, except the Problem Management application will note that there is already an open problem ticket for that device, and will turn the problem open event into a problemupdate event (pmu). If a trap arrives that contains a keyword indicating the problem has been resolved, SCAuto for NetView creates a problem close event (pmc) and sends it to the Event Manager. The Event Manager closes the problem for the specified device.

Problems opened by SCAuto for NetView contain the following information:

- Date and time the trap was reported.
- Name of the device for which the trap was sent.
- The SNMP codes for the trap.
- Description of the trap.

NetView User Interface Integration Overview

SCAuto for NetView provides facilities for integrating into the NetView graphical user interface. This integration provides quick access to ServiceCenter from within the NetView user interface, and visual notification of ServiceCenter probem status.

A **ServiceCenter** menu is added to the main NetView menu bar during installation of SCAuto for NetView. This menu provides easy access to ServiceCenter functions, including inventory and problem management functions. When selecting a **ServiceCenter** menu entry, a ServiceCenter GUI client is started which automatically logs in and brings up the requested application screen, bypassing intervening menus and application screen. This process of starting ServiceCenter and bringing up the desired screen automatically is called a *cut-through*.

Note: This capability is available only if the ServiceCenter client is installed, and the ServiceCenter cut-throughs are enabled when SCAuto for NetView is installed.

For example, aNetView operator selects a host icon in the NetView window and would like to request a list of open ServiceCenter problems for this host. The operator selects the ServiceCenter menu and chooses the **Problem List** menu option. The selected host name is passed to a streamlined problem lookup. Any problems for that host are displayed in a new ServiceCenter client window.

Refer to *Chapter 2,Installation* for inormation on enabling these facilities.

SCAuto For NetView System Flow

SCAuto for NetView runs as a well-behaved agent under the NetViewProcess Monitor. SCAuto is started using the standard NetView process management commands. As a user, you can start SCAuto for NetView with the **ovstart** command, and SCAuto will also start automatically whenever NetView is started. You can inquire into SCAuto status using the **ovstatus** command. SCAuto for NetView can also be run as a standard foreground or as a background process under UNIX, which insures that all error messages are displayed and allows you to run as a standard user during testing.

When the **ovstart** command is issued, NetView starts two SCAuto processes, *scevmon* and *sclogger*. Refer to Figure 1-4.

The *sclogger* daemon monitors the NetView *trapd.log* file for traps which contain SCAuto keywords. When a trap is selected, it is analyzed, reformatted and time stamped. A standard record is written to the *scauto.traps* file for processing, and the daemon waits for the next event.

The *scevmon* daemon connects to ServiceCenter via the SCAuto server. Upon connection, *scevmon* daemon ensures that you are licensed for use. The *scauto.log* file is created or updated with each action and is especially useful in problem determination. *Scevmon*

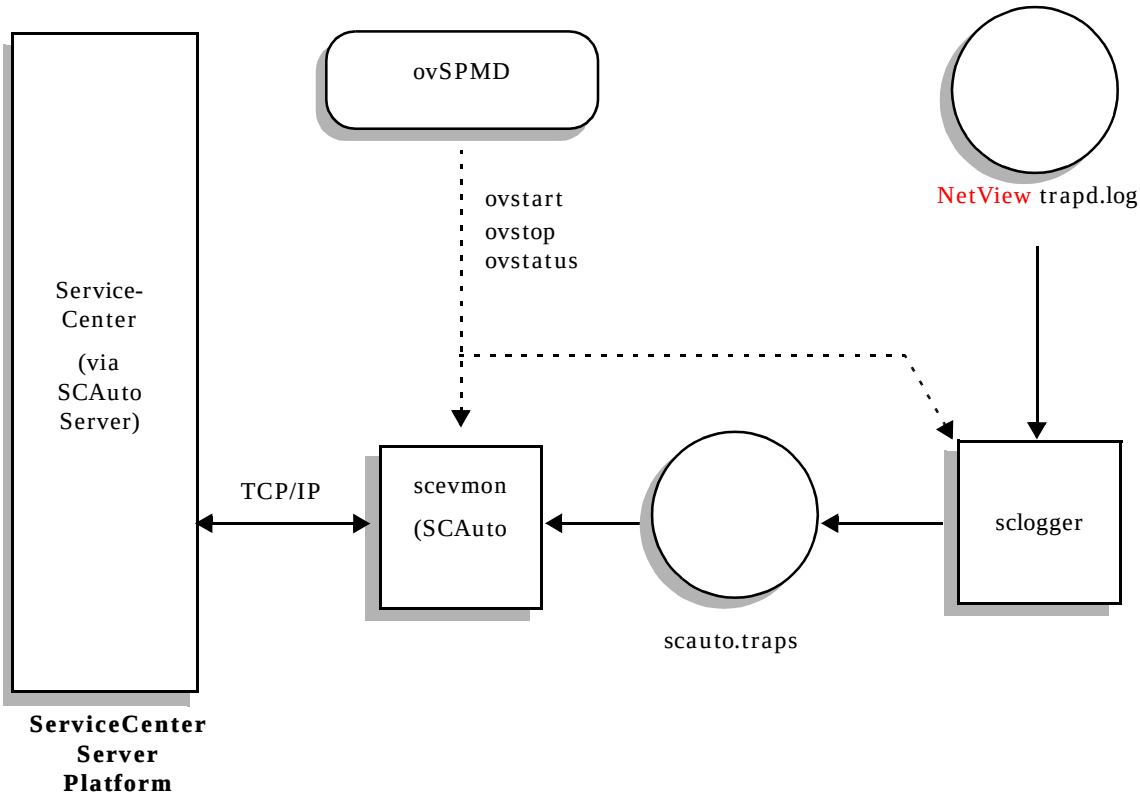

Figure 1-4 SCAuto for NetView System Flow

then reads the *scauto.traps* file for any unprocessed traps. It maintains its position in the *scauto.traps* file with the *scauto.chk* checkpoint file, and positions itself during startup if necessary. The checkpoint file allows *scevmon* to be stopped and then restarted from where it last left off trap processing. Trap records are processed from the checkpoint position forward, creating ServiceCenter corresponding to each trap.

Scevmon and *sclogger* may be stopped with the **sctop** command.

The *scdiscover* program is used to discover inventory and create inventory records for them within ServiceCenter. It is not run automatically from NetView, but can be run manually from the command line. Normally, *scdiscover* only needs to be run once, to gather initial inventory. Afterwords, *scevmon* will maintain the inventory records by processing traps that indicate inventory changes.

Note: Novell inventory is currently not maintained automatically. If you have Novell inventory, the *scdiscover* program should be run periodically to keep the inventory up to date.

Chapter 2 Installation

11/13/97

Overview

This chapter provides information, instructions, and verification procedures for installing SCAuto for Netview.

Pre-Installation Notes

- SCAuto for Netview requires approximately 10 MB of hard disk space.
- SCAuto for Netview must run on the same platform as your NetView.
- Obtain an SCAuto for NetView authorization code from your Peregrine Systems Account Executive or Customer Support. After adding the new authorization code to the server *sc.ini* file, restart ServiceCenter.
- The installation asks for the name of a user that will own the SCAuto for NetView files. If you wish to create a new user for this, do so before starting the installation. It is suggested that this be the same user that owns the ServiceCenter files. The *root* user cannot be used for this purpose. You may also create a new user group for this purpose as well.
- If you wish to enable ServiceCenter cut-throughs, you should ensure that the ServiceCenter client and GUI files (*scclient*, *scguimtf*, *scgui.uid*, and the *bitmaps* directory) exist on the NetView platform. They may need to be copied from a remote server machine if NetView and the ServiceCenter server do not run on the same machine.
- Decide where you want to install SCAuto for NetView. The recommended directory is */usr/scauto/nvaix*. Other good choices are */opt/OV/scauto* and */usr/OV/scauto*. Any other choice should be used only for testing or development, otherwise SCAuto may be unable to find important files when it starts. (That is, one may have a production version of SCAuto running, and still install a new version for testing in a different directory.)

Installation Procedures

This section provides the SCAuto for NetView installation instructions.

1. SCAuto for NetView is contained as a **tar** file on the distribution CD-ROM in the *nvaix* (NetView for AIX) directory. The tar file will be named according to the UNIX platform; *hpux.tar* for HP-UX, *aix.tar* for AIX, etc.

NOTE: For HP-UX, all file and directory names on the CD-ROM will probably be in all uppercase. File names will have a "*;1*" suffix as well. Adjust these file names accordingly in the instructions below.

2. Set your user to root, or login as root. The following steps will need to have privilege to update NetView.
3. Mount the CD-ROM on your system. The following steps will assume the CD-ROM is mounted on the /cdrom directory. Adjust the steps accordingly if you mount the CD-ROM elsewhere.

   Ex: **mount /cdrom**
4. Create a temporary directory. This directory will hold files extracted from the distribution tar file, and can be deleted after installation is finished. This directory will not be the final location for SCAuto for NetView.

   Change directory into the temporary directory.

   Ex: **mkdir /tmp/scauto; cd /tmp/scauto**
5. Un-tar the files from the CD-ROM into the temporary directory.

   Ex: **tar xvf /cdrom/nvaix/aix.tar**

   ("*/cdrom/OVNNM/HPUX.TAR\;1*" for most HP-UX systems)
6. You can now unmount the CD-ROM if you wish.
7. Run the SCAuto for NetView installation script. It is named *INSTALL*.

   Ex: **./INSTALL**
8. The installation script will now explain and ask a series of questions, copy the necessary files, and update NetView to know about SCAuto. The questions that will be asked are:

   a. Install into which directory?

   By default, the product will install into the recommended location of */usr/scauto/nvaix*. Other alternatives that may be preferred are */usr/OV/scauto* or */opt/OV/scauto*.

   This directory will be referred to as the *installation directory* elsewhere in this manual.

b. *User id to own SCAuto files?*

   This is the name of a user id that will own the SCAuto files. The installation script will verify that this is a correct user. If you wish to specify a group id as well as user id, you may answer this question as: "*userid.groupid*".

c. *Do you wish to enable ServiceCenter cut-throughs from within NetView?*

   If you have a ServiceCenter client on your local machine, and wish to add the *ServiceCenter* menu to NetView, answer yes here.

d. *In which directory is the ServiceCenter client binary located?*

   If you answered yes to the previous question, you will be asked where the installation script can find the ServiceCenter client binaries. The install script will make symbolic links to those binaries. If you don't have these binaries on your platform, you can accept the default directory, then patch up the links later.

e. *Press Enter to continue.*

   You will be presented with the answers you gave to the questions. If you are unsatisfied, you can press control-c to interrupt now. Up until this point, the installation script has made no changes to your system.

f. *Would you like me to build an initial checkpoint file from the existing IPAS checkpoint files? Would you like me to unregister IPAS?*

   These questions are asked near the end of the installation if the script detects that you have IPAS installed on your system. SCAuto for NetView supersedes IPAS.

9. Review the output produced from the installation script for errors and warnings. You may safely rerun *INSTALL* to finish an incomplete installation.

10. Update the *scauto.ini* file in the installation directory with your SCAuto server name, and update the *sc.ini* file with any ServiceCenter client specifications.

    The *scauto.ini* configuration options are described in the next section.

If the installation was successful, and *scauto.ini* has been set up:

11. You may remove the temporary directory.

12. Run the *scdiscover* command to discover the initial NetView inventory. See the next chapter for details.

13. Start the *scevmon* and *sclogger* background monitors. See the next chapter for details.

SCAuto Configuration File

The *scauto.ini* file contains configuration information used by SCAuto for NetView. It is a normal text file, and may be updated with any text editor. All lines in *scauto.ini* beginning with a "**#**" are treated as comments. The following options may be set in the configuration file:

- **server: *hostname.service***

  SCAuto server specifier. *Hostname* should be the machine name or IP address of the machine the SCAuto server is running on. *Service* should be the service name (from */etc/services*) or port number that the SCAuto server is listening on (this should be the same value that is specified under the *scauto:* keyword in the server's *sc.ini* file).

- **inventory: *yes | no***

  Should SCAuto automatically maintain inventory? The default is yes.

- **problem: *yes | no***

  Should SCAuto open and close problems? The default is yes.

- **location: *default location***

  This value is used as the default problem and inventory location (for when there is no location defined in a MIB). This may be left blank.

- **category: *default problem category***

  This category is used when creating problem records. The default is "equipment".

- **network: *ip | sna | novell | all***

  Specifies the type of inventory to discover and maintain. This line may be repeated multiple times, if more than one network type needs to be discovered.

- **notify:**

  This option is currently unused and is kept only for compatibility with SCAuto for Openview.

- **eventsuffix:**

  An alphanumeric option to be appended to all events created by SCAuto. The default is no suffix. This option only be necessary in when using customized RAD code.

- **sleep: *seconds***

  The number of seconds to sleep between checking for new events or traps.

- **dateformat: *1 | 2 | 3***

  This specifies the type of date format to use when creating events. This should match the equivalent option within ServiceCenter. This option should not be necessary unless the ServiceCenter defaults have been changed. The values correspond to:

1: US date format (default). Dates are in the form *mm/dd/yy*.

2: European date format. Dates are in the form *dd/mm/yy*.

3: Sortable date format. Dates are in the form *yy/mm/dd*.

Installation Verification

This section includes installation checks and possible solutions for installation problems.

1. Start your ServiceCenter server and SCAuto server. Ensure that the SCAuto server has started correctly (using the *status* display from within ServiceCenter).
2. Give the command: **scdiscover -test**

   This will perform all the normal initialization and connection to SCAuto server, but will not perform any discovery. This is useful to test that the *scauto.ini* can be parsed correctly and that a connection to the SCAuto server can be made.
3. Give the command: **scdiscover**

   This will perform initial inventory discovery. Verify that events are being created and inventory records added to ServiceCenter. View the *scauto.log* for any possible error messages.
4. To start the SCAuto for NetView monitors, enter the command:
   ovstart scevmon sclogger

   You will need to be logged in as *root* to perform this command. These monitors will automatically be started whenever NetView is started, so this command does not need to be added to your system start-up scripts. You can review the status of these monitors at anytime by issuing:
   ovstatus scevmon sclogger
5. Verify that the *sclogger* monitor is running by viewing the *scauto.traps* file. This file contains traps that *sclogger* has processed. This file may be empty if there are all the NetView traps are filtered by the *scauto.keywords* file.
6. Verify that the *scevmon* monitor is running by viewing the scauto.log file. This file contains log messages, both for normal informative messages as well as errors and warnings.
7. When traps show up in the *scauto.traps* file, verify that events are being created within ServiceCenter (give the **eventin** command at a ServiceCenter command prompt). Look for events of the form *OV-hostname*, where *hostname* identifies that machine SCAuto for NetView is running on.
8. If the ServiceCenter cut-throughs were enabled during installation, verify the ServiceCenter menu was added by starting NetView (with the **ovw** command). The NetView menu bar should contain a **ServiceCenter** option. Try out some of the menu choices to verify that the ServiceCenter client can be started. If the client does not start, change directory to the SCAuto installation directory and give the command "**scclient -G**". If this does not cause a ServiceCenter client to open, review any *sc.log* files for error messages. Ensure that you have the DISPLAY environment set to the appropriate X Window server.

Customizing

When processing NetView traps, SCAuto can dynamically creates four types of events in ServiceCenter:

- pmo (problem open)
- pmc (problem close)
- icma (inventory add)
- icmd (inventory delete).

The event type and subsequent processing is determined by the trap information in *scauto.traps*. The parse of the trap record determines if the record is ignored (filtered), or what type of operation should be performed.

The file that determines the parse is *scauto.keywords*. For each trap, the keywords are searched in the order they appear in *scauto.keywords*, and the category of the first matching keyword determines what SCAuto will do with the trap. There are five categories (which may be repeated if necessary):

- *PROBOPEN*: Traps matching these keywords are turned into problem open events.
- *PROBCLOSE*: Matching traps cause problem close events.
- *TOPOADD*: Matching traps cause inventory add events.
- *TOPODE*L: Matching traps cause inventory delete events.
- *NONE*: Matching traps will always be ignored, even if they match keywords in later sections.

Traps that match no keywords will be ignored. The format of a category is:

***CATEGORY*: '*keyword*' '*keyword*' ... ;**

New lines may be placed between keywords. The category is ended with a ";" character. Lines that being with "#" are treated as comments. The supplied *scauto.keywords* file should be used as an example of allowed formatting.

A keyword may contain one or more wildcards characters. A wildcard character is an asterisk (*). A wildcard matches any number of characters. For instance, the keyword '**IF * down**' will match a trap containing the stri '**IF 127.0.0.1 down**'.

The entire trap record as contained in *scauto.traps* is checked when looking for a keyword match. Thus, the keyword can be used to match on hostnames, SNMP trap codes, and trap descriptions.

The supplied *scauto.keywords* should be appropriate for most initial tests and operation. However, if *scauto.keywords* does not contain the codes or descriptions of traps you want processed by SCAuto, you can update the specific category with a new keyword. If you want fewer traps processed, you can delete keywords, or comment them out with the # character. The supplied *scauto.keywords* comes with several entries already commented out, to reduce on the volume of events created while still providing real examples of keywords.

If the available traps and keywords are not suitable, you can also modify the NetView trap configuration file in order to produce unique trap records to be parsed by SCAuto.

The NetView *trapd* monitor receives the SNMP trap and converts the SNMP information into a *trapd.log* record. The log record is created from the variables received in the trap and the format specified in NetView's *trapd.conf* configuration file. You could modify the configuration file to reflect a meaningful description, such as *SCAutoOpen*, in the description field of the trap. This would then provide a unique keyword for *scauto.keywords*. Refer to the NetView documentation on *trapd.conf* for information on how to change this file.

Whenever you update the *scauto.keywords* file, you must stop and restart the SCAuto monitors to pick up the changes.

Chapter 3. Operations

11/13/97

Overview

This chapter covers operation procedures for starting, stopping, and checking the status of SCAuto for NetView. The use and syntax of all commands and utilities are explained.

Inventory Discovery and Refresh

The **scdiscover** command is used to gather IP, SNA, and Novell inventory. This should be done before starting the scevmon and sclogger background monitors, otherwise they may open problems for devices that ServiceCenter does not yet know about.

For IP and SNA inventory, *scdiscover* only needs to be run once, just after installation. IP and SNA inventory will be maintained automatically through NetView traps. For Novell inventory, *scdiscover* should be run periodically. You may also wish to refresh inventory for several reasons; e.g., OpenSNA has just been installed and you need to initialize SNA inventory; a data corruption problem; etc.

The discovery utility may be run concurrently with the SCAuto background monitors.

The Inventory Discovery Utility can be started by:

scdiscover

This will cause *scdiscover* to connect to the SCAuto server, and discover inventory for all enabled networks in the *scauto.ini* configuration file.

Options for *scdiscover* may be given on the command line to override the default behavior (i.e., you wish to refresh only Novell inventory):

-server *hostname*.*service*

Connects to the named SCAuto server.

-ip | **-sna** | **-novell** | **-all**

Discover inventory for the specified networks. You may specify more than one of these options, e.g., **scdiscover -ip -sna**.

-software	When Novell inventory is enabled, this will gather station software inventory from ServerView. Normal inventory gathering for Novell will not be done in this instance.
-type *invtype*	Only gather inventory for the named inventory type (*node*, *Network*, *Subarea*, etc.). Normally all types are discovered.
-name *host*	Start inventory discover from the named host. For IP and SNA networks, only the named object and its subobjects are discovered. For Novell inventory, this is the name of a Novell server, and it and its clients will be discovered.
-test	Used for testing SCAuto for NetView installation. It parses *scauto.ini*, connects to the SCAuto server, but does not do any discovery at all.
-debug	Enable additional log messages.

SCAuto for NetView Background Monitors

The SCAuto background monitors process NetView traps, creating problem and inventory events in response. These monitors are normally started automatically whenever NetView starts. They may be run from the command line for testing or development.

Trap Logger

The *sclogger* monitor, also known as the trap logger, reads the NetView *trapd.log* file to find SNMP traps. After filtering the traps through the *scauto.keywords* file, it writes a processed version to *scauto.traps*. *Sclogger* does not talk to the SCAuto server, and thus can be run at all times, even if ServiceCenter is down.

The *sclogger* monitor has a few debugging options. These do not normally need to be used except for debugging and development.

-debug This causes additional messages to be logged.

-trapfile *file* Use the named file for monitoring traps, instead of the default NetView *trapd.log*.

-onetime Causes *sclogger* to quit after processing all traps. Normally it will sleep while waiting for more traps to arrive.

Event Monitor

The *scevmon* monitor, or event monitor, reads the processed traps from *scauto.traps* and sends them on to the SCAuto server to be created as ServiceCenter events. If the trap is for an inventory addition or change, *scevmon* will discover inventory for just the affected device.

The *scevmon* monitor has only one parameter, *-debug*. This turns on additional debugging messages for the *scauto.log* log file. To use this option, you will need to run *scevmon* from the command line.

Starting the Background Monitors

The SCAuto for NetView background monitors are controlled by the NetView Process Manager and are started as **WELL_BEHAVED_AGENT**s.

1. This is accomplished by issuing the NetView command:

   ovstart

   You normally need special permissions to execute this command. The background monitors will automatically start whenever NetView is started.

Note: This and all other NetView commands are contained in the NetView binaries directory. This directory should normally be added to your *PATH* environment variable, or you may prefix the commands with this directory. For instance, the above command might be **/usr/OV/bin/ovstart** or **/opt/OV/bin/ovstart**. See your platform system administrator or NetView administrator if you are having trouble with these commands.

2. The SCAuto monitors may also be started individually under NetView process management by issuing the following command:

   ovstart scevmon sclogger

3. You can also start the monitors outside of the NetView process management by issuing the following commands:

   scevmon &

   sclogger &

   This is most useful when running the commands as a test, verifying installation, or determining problems.

   - You should first ensure that these processes are not already running under the NetView Process Manager. See the later *Stopping SCAuto for NetView* section for instructions.
   - If you elect to use this method as the standard method of starting the background monitors, you should delete the Process Manager objects created during the installation. This prevents these monitors from starting whenever NetView starts. To delete these objects, change to the SCAuto installation directory and issue these commands:

     ovdelobj scevmon.lrf

     ovdelobj sclogger.lrf

Stopping the Background Monitors

This section provides the steps for stopping SCAuto or NetView background monitors.

1. You can stop either SCAuto or all NetView agents by issuing the following NetView commands.

   a. To stop all agents, issue this command:

      ovstop

   b. To stop just SCAuto for NetView, issue this command:

      ovstop scevmon sclogger

2. A **kill** should be issued if a background monitor is taking an excessive time to respond to the **ovstop** command, of if the monitors were not started with **ovstart**. To execute the **kill** command:

   a. First issue the following commands to get the necessary process ids to complete the **kill** command:

      ps -ef | grep scevmon

      ps -ef | grep sclogger

   b. Extract the *scevmon* and/or the *sclogger* process ids. Substitute those values in the following command:

      kill *pid1* [*pid2*]

      Where *pid1* is the *scevmon* process id, and *pid2* is the *sclogger* process id.

SCAuto for NetView Status

This section provides commands for checking on the status of SCAuto for NetView.

1. During execution, you may monitor the *scauto.log* for any messages produced by the background monitors.
2. The NetView Process Manager also provides commands to query agent status:
   a. To get the status for all NetView agents, issue this command:

      ovstatus

   b. To get the status for only the SCAuto monitors, issue this command:

      ovstatus scevmon sclogger

Additional Utilities and Commands

This section describes additional SCAuto for NetView utilities and their operation.

Trap Archive Utility

An **Archive** utility is provided to archive the *scauto.traps* file, which can be deleted or saved to external media. The *scauto.traps* could grow infinitely based on the disk available. Periodically, you should remove processed records and save the disk. The archive utility produces two files: an archive file containing all processed records, and a new *scauto.traps* with all unprocessed records.

In order to run the archive, the background monitors (scevmon and sclogger) must be stopped. The archive utility can stop the background monitors for you if they were started with the **ovstart** command. To execute the archive utility, issue the following command:

scarchive

You must be in the SCAuto for NetView installation directory when running the command. There are no parameters.

After running, the *scauto.traps* will contain only unprocessed traps. There will also be a *scauto.traps.*<*date*> file containing all the old processed traps, where *date* is a date and time stamp to identify the file.

Uninstall utility

During installation, an *uninstall* script is created. This program may be run to uninstall SCAuto for NetView, cleaning up any files and registrations added to NetView during installation.

Auxiliary Commands

The following commands are not meant to be run directly, but are executed from other SCAuto components.

scelogin This command is used when selecting a menu item from the NetView ServiceCenter menu. It executes the ServiceCenter *scclient* program. For debugging purposes, if you define and export the SCELOGIN_DEBUG environment variable *before* starting *ovw*, then this cut-through will not execute *scclient*, but will instead print the command it would normally execute.

svvprint This prints a list of all Novell servers that NetView knows about.

svvget This is a simple script to interface to ServerView's *siget* command.

scevauth	This script is called when *scevmon* starts. Its primary purpose is to perform any actions that need to be done during start-up. For instance, any necessary NetView **nvauth** commands can be placed here if your NetView setup requires authorization.

Miscellaneous Files

scauto.reg	This is the registration file for the NetView **ServiceCenter** menu. There is a symbolic link from the NetView registration directory to this file, so that it does not need to be copied to NetView if it is changed.
scevmon.lrf	This is the registration file that allows *scevmon* to be started from the **ovstart** command.
sclogger.lrf	This is the registration file that allows *sclogger* to be started from the **ovstart** command.
scauto.msg	Contains the messages that SCAuto display or write to *scauto.log*. It may be modified for use with languages other than English.

Chapter 4. The ServiceCenter Menu

11/13/97

Overview

SCAuto for NetView provides an enhanced operator interface to ServiceCenter that can run under NetView. From aNetView window, you can access a number of ServiceCenter screens to gather information related to the current window or selected object.

This capability is only available if the ServiceCenter cut-throughs were enabled during the SCAuto for NetView installation. If the cut-throughs were not enabled, you can start a ServiceCenter client from the UNIX command line instead.

Note: Refer to the appropriate ServiceCenter documentation for more information on using ServiceCenter.

ServiceCenter clients are started through the **ServiceCenter** menu in an NetView window. Depending upon the menu item selected, the client window will be opened to different ServiceCenter applications and screens.

The login name of the UNIX user that started the NetView window session will be passed as the name of the ServiceCenter operator. Thus it is recommended that ServiceCenter operator names and UNIX user names coincide. The ServiceCenter login screen will be presented if a password is needed or the operator name does not exist, after which the requested application will be displayed.

Note: Inorder to run a ServiceCenter client, the NetView window must not have been started from the *root* user account.

All menu options are available if an icon for an object is selected in an NetView window (Figure 4-1). Specific requests requiring an object selection are grayed out if an icon is not selected.

The following screens are a tutorial representation of SCAuto general operations. Screens and functions may change from release to release, so reference the help files on your specific platform for the latest operational details.

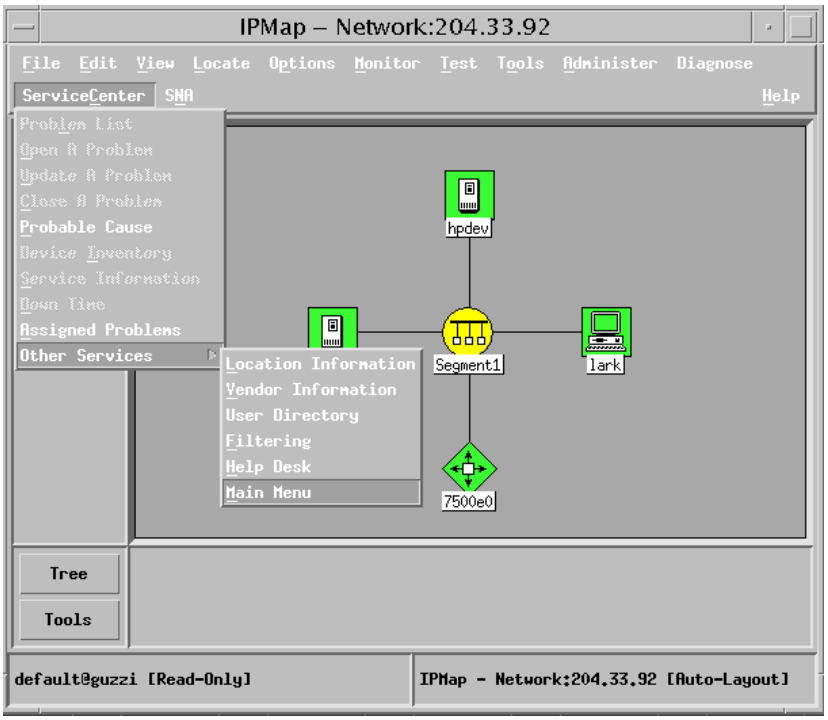

Figure 4-1 ServiceCenter Menu

Requirements

Before using SCAuto for NetView, you should have a good working knowledge of:

- ServiceCenter applications.
- ServiceCenter Client/Server.
- NetView graphical user interface.

While some procedures for these applications are explained, others are referenced. You should refer to the appropriate ServiceCenter documentation for a more detailed explanation.

ServiceCenter Menu Options

The ServiceCenter menu options take you directly to the ServiceCenter applications fromwithin NetView. These services save the time of logging in and navigating through ServiceCenter to get to these applications. The following sections provide a brief description of the screens.

Note: While some of the ServiceCenter application options are mentioned in this manual, you should refer to the ServiceCenter documentation for complete instructions on using the ServiceCenter applications.

To use a ServiceCenter application under SCAuto for NetView:

1. Select the **ServiceCenter** menu in the NetView window and select the appropriate menu option. Some ServiceCenter menu options are not available unless an object is selected in the NetView window.
2. Use the mouse or keyboard to navigate through a screen.
3. To leave the application, select the **Back** button or press the **F3** key. This takes you to the previous screen or to a logout screen.
4. When the logout screen is displayed (Figure 4-2), select the **EXIT** button from the popup menu or press the **F1** key to exit the ServiceCenter session. .

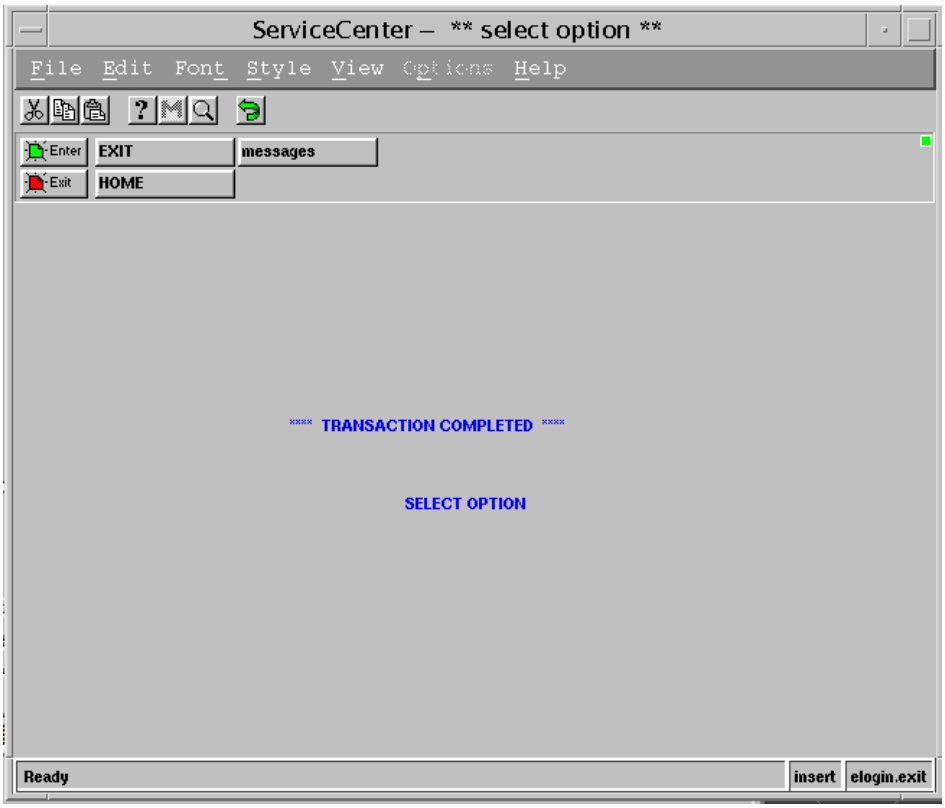

Figure 4-2. Logout Screen

Note: The descriptions on the following pages are for ServiceCenter *Release 2*. If you are using an earlier version of ServiceCenter, the procedures will differ and the screens will not be similar to what are shown. The menu options will be the same however.

Problem List

The ***Problem List*** menu option provides a list of problems currently open in ServiceCenter for the selected object. When this option is selected, a problem list is displayed (Figure 4-5). Use the **Options** menu to get a list of operations to perform on the problems.

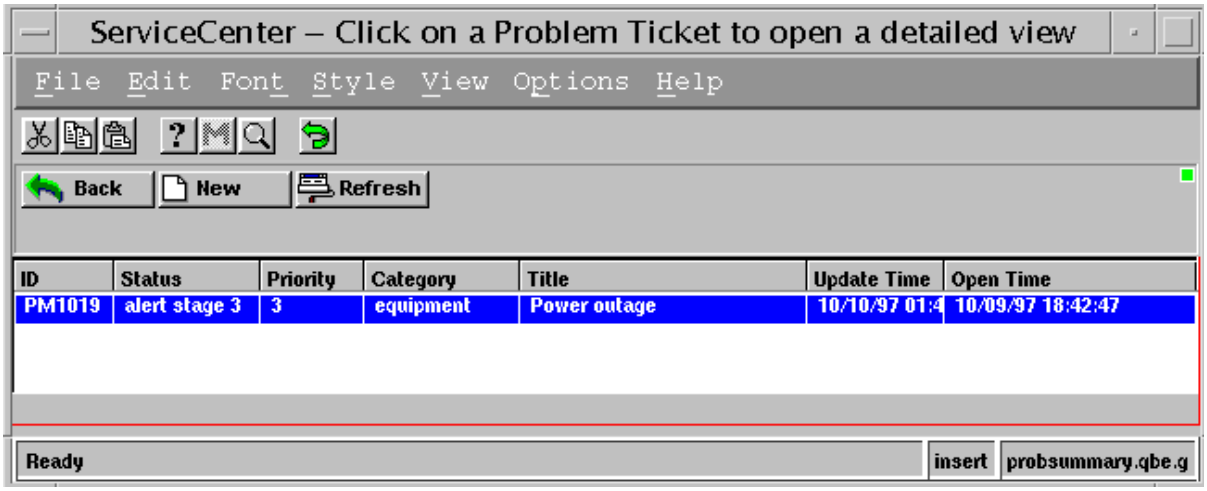

Figure 4-3. Problem List Screen

Open A Problem

The ***Open A Problem*** menu option allows you to open a problem in ServiceCenter for the selected object. When the option is selected, the **Create a New Problem Record** ServiceCenter screen is displayed (Figure 4-5).

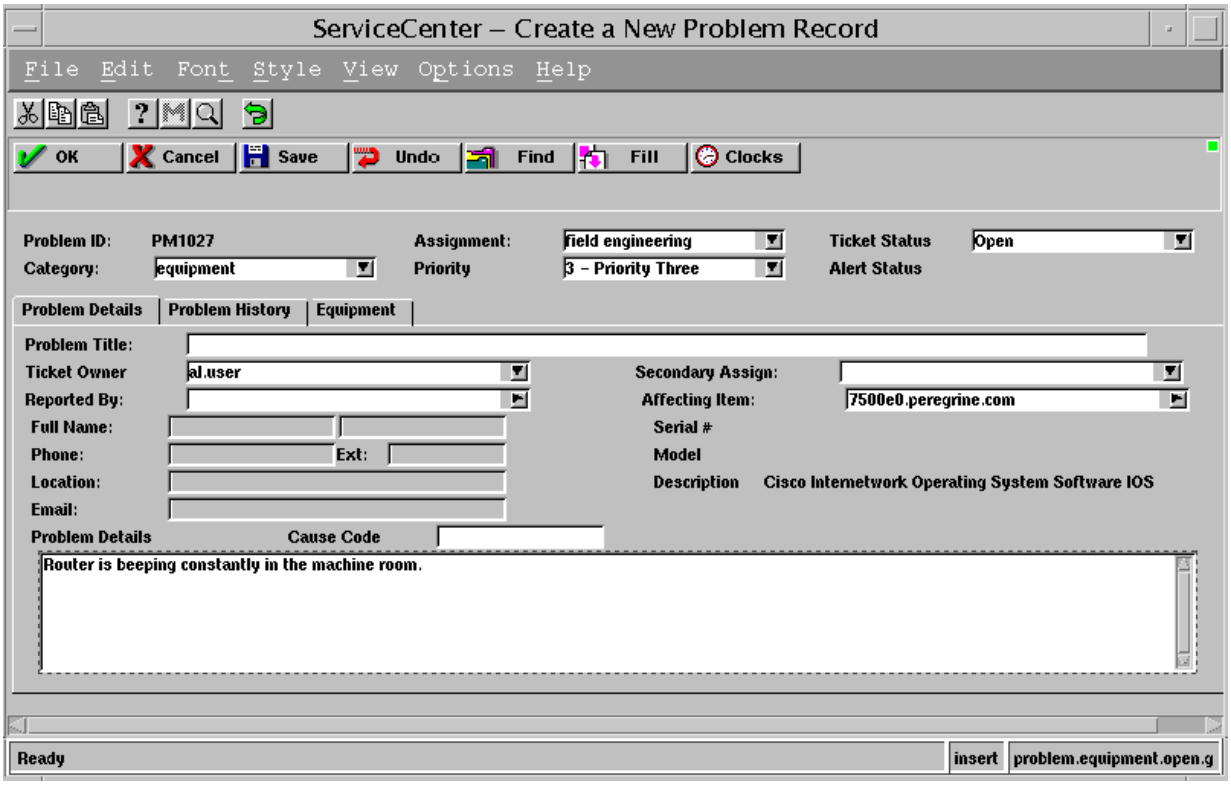

Figure 4-4. Open a Problem Screen

Update A Problem

The ***Update A Problem*** menu option allows you to update an open problem in ServiceCenter for the selected object. When the menu option is selected, a list of ServiceCenter problems for the device is displayed. ServiceCenter will display an appropriate message if there are no open problems for the selected device.

- Double click on the desired problem in the list. The **Examining Problem** ServiceCenter screen is displayed (Figure 4-5).
- Click on the **Action Descriptions** tab. Enter your update description.
- Click on the **Save** or **OK** button to save your changes and update the problem ticket.

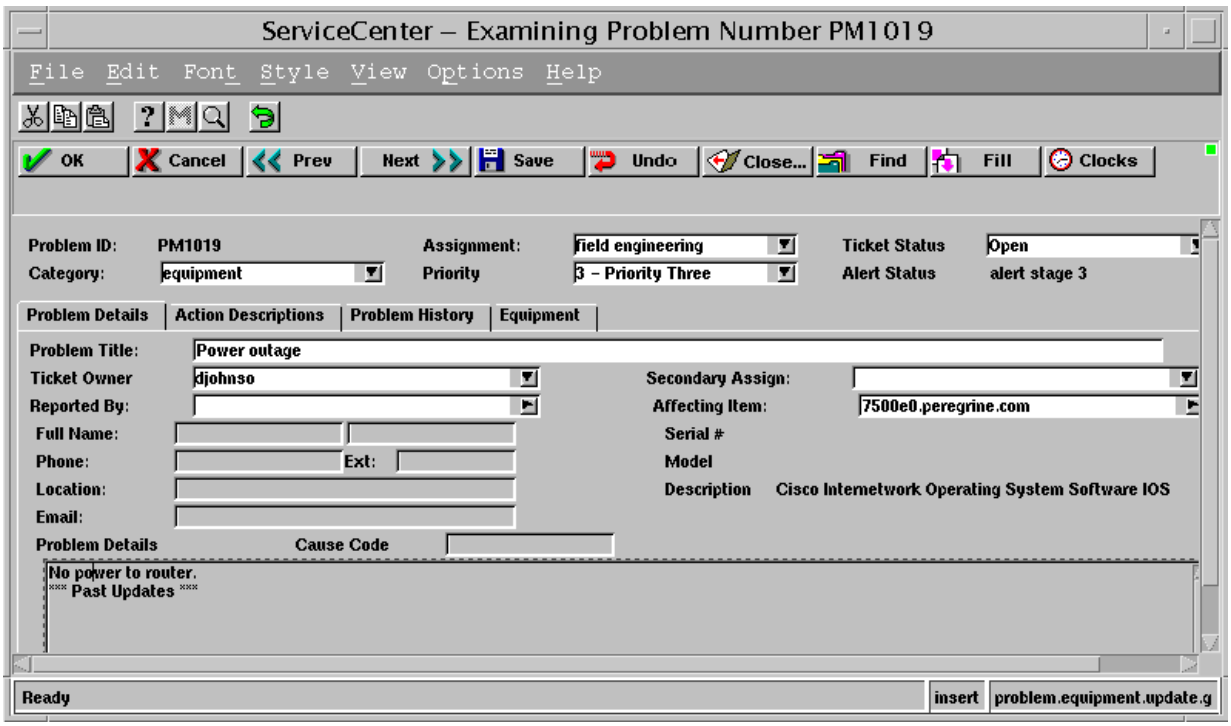

Figure 4-5. Update a Problem Screen

Close A Problem

The ***Close A Problem*** menu option allows you to close an open problem in ServiceCenter for the selected object. When the menu option is selected, a list of ServiceCenter problems for the device is displayed. ServiceCenter will display an appropriate message if there are no open problems for the selected device.

Double click on the desired problem in the list. The **Examining Problem** ServiceCenter screen is displayed (Figure 4-5).

- Click on the **Close...** button.
- Click on the **Action Descriptions** tab. Enter your resolution description in the **Solution** window.
- Click on the **Save** or **OK** button to save your changes and close the problem ticket.

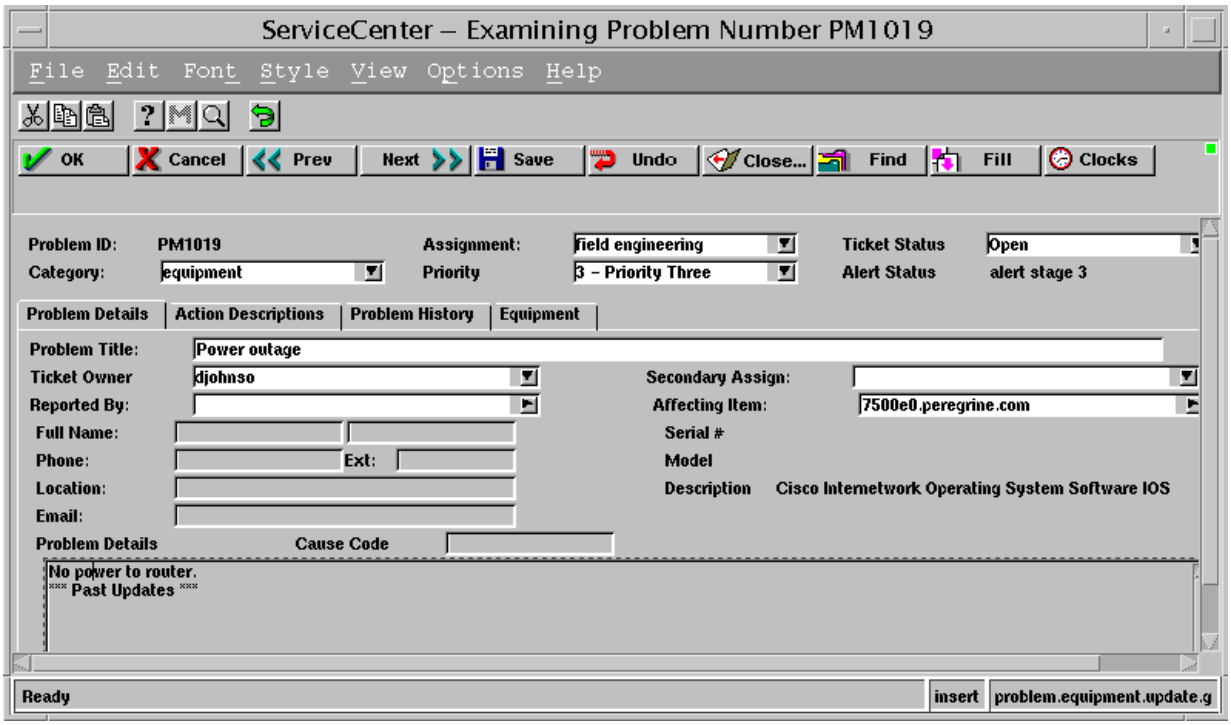

Figure 4-6. Close a Problem Screen

Probable Cause

The ***Probable Cause*** menu option allows you to query ServiceCenter for the probable cause of a problem. When first accessed, a blank *probable cause* screen appears. If you press **return**, a list of cause codes appears. You can select one of the listed probable causes by double-clicking on it, which will display the *probable cause* screen (Figure 4-7). The **Resolution** field lists any solution that has been determined for the problem.

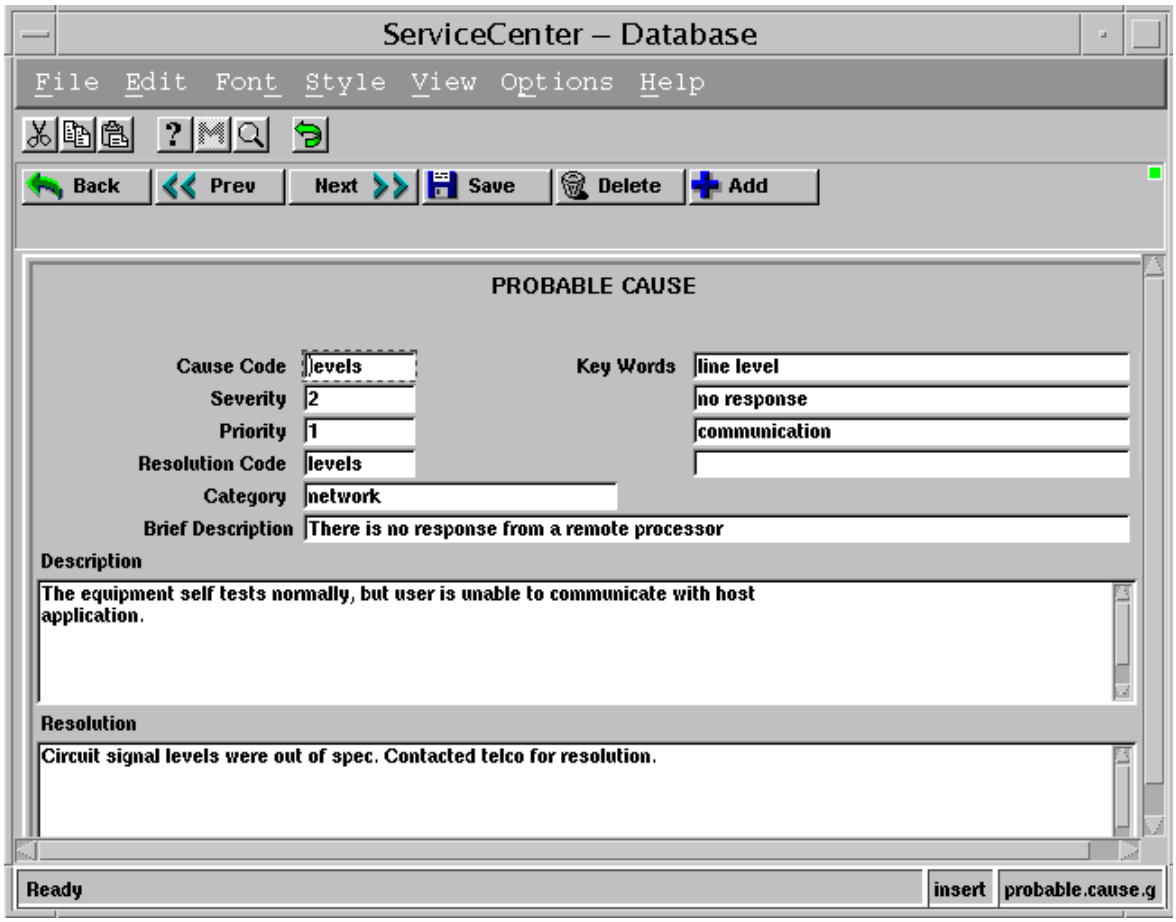

Figure 4-7. Probable Cause

Device Inventory

The ***Device Inventory*** menu option takes you to the ServiceCenter **Asset Mangement** screen (Figure 4-8)

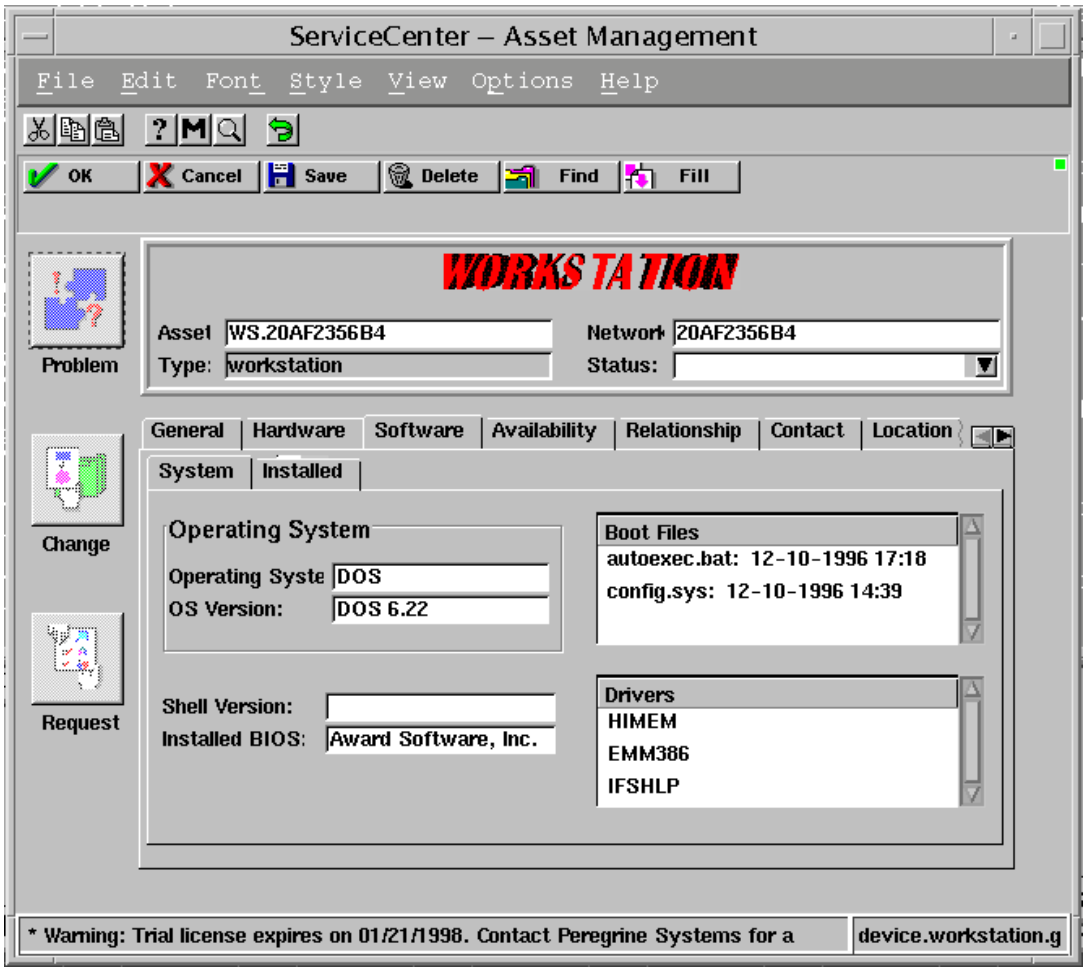

Figure 4-8. Device Inventory Screen

Service Information

The ***Service Information*** menu option takes you to the ServiceCenter **Asset Management** screen (Figure 4-9). Under ServiceCenter 1.4, the options available will be different from the ***Device Inventory*** menu option.

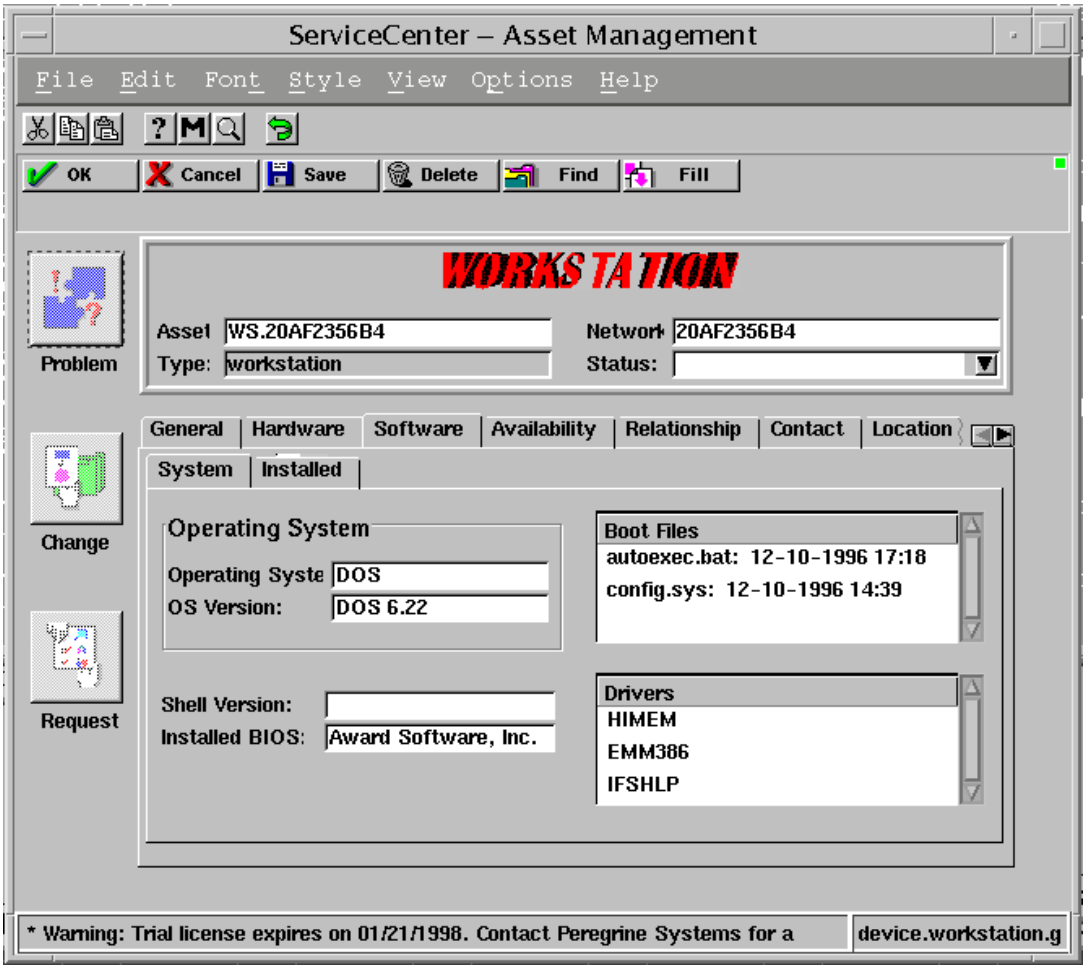

Figure 4-9. Service Information

Down Time

The ***Down Time*** menu option displays the Downtime screen for the selected object (Figure 4-10).

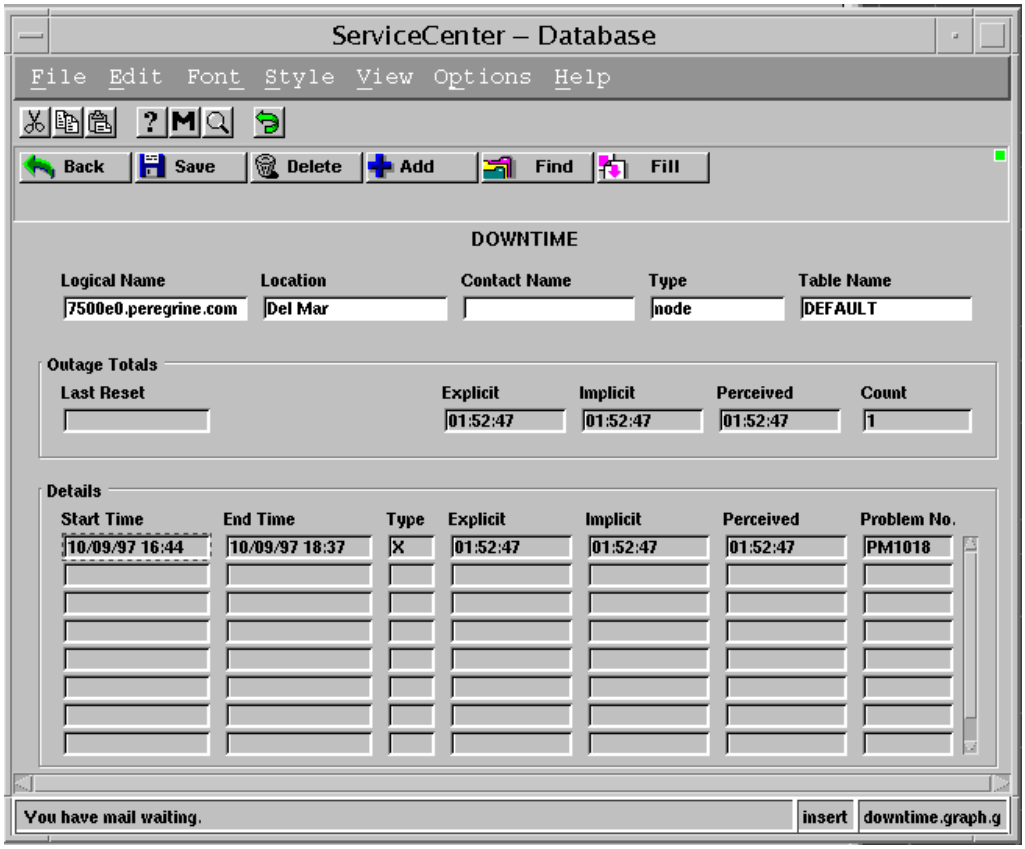

Figure 4-10. Down Time Screen

Assigned Problems

The ***Assigned Problems*** menu option provides a list of open problems assigned to the operator using the current NetView session. This summary is displayed in the **Open Problems** screen (Figure 4-11).

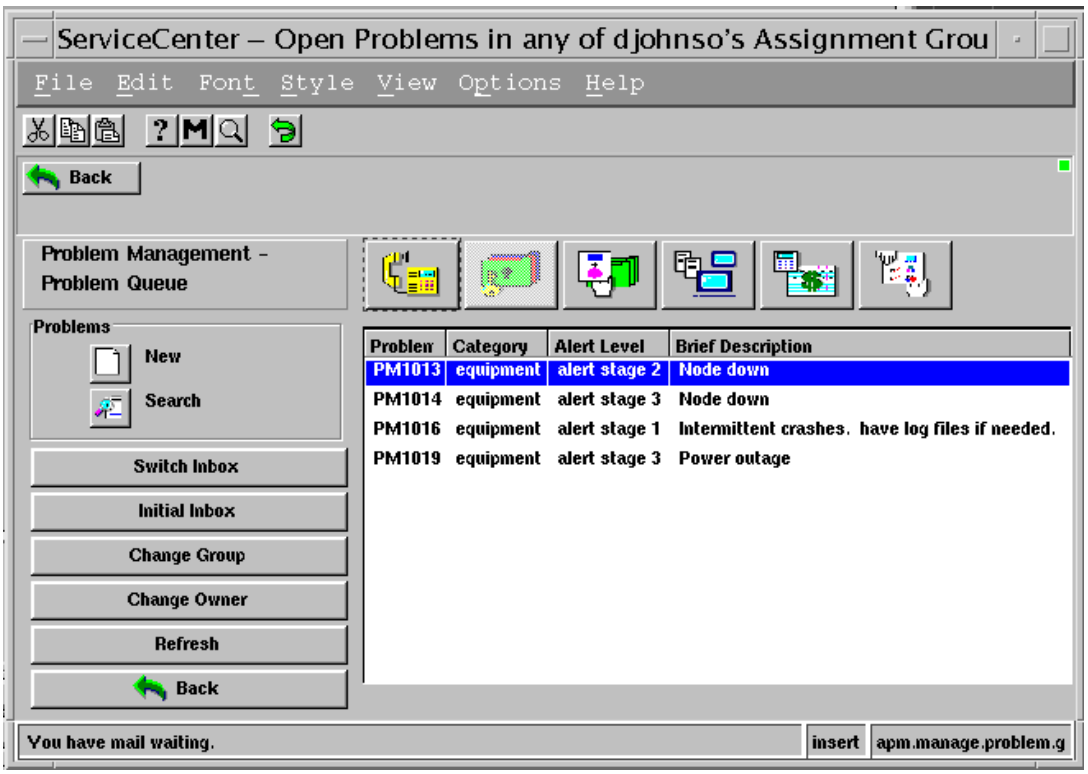

Figure 4-11. Assigned Problems Screen

Other Services

The ***Other Services*** menu option provides a sub-menu with additional options.

Location Information

The ***Location Information*** menu option provides location records, much like an address book. When the **location** screen (Figure 4-12) is first accessed, the screen is blank.

- To find location data, enter the **location name** and click o the **Find** button.
- To view a list of locations, press **Return** while in the blank screen. A summary list is displayed.
- **Double-click** on the desired location to see the data for that location.

The location screen will also allow to search, edit, add and update location information.

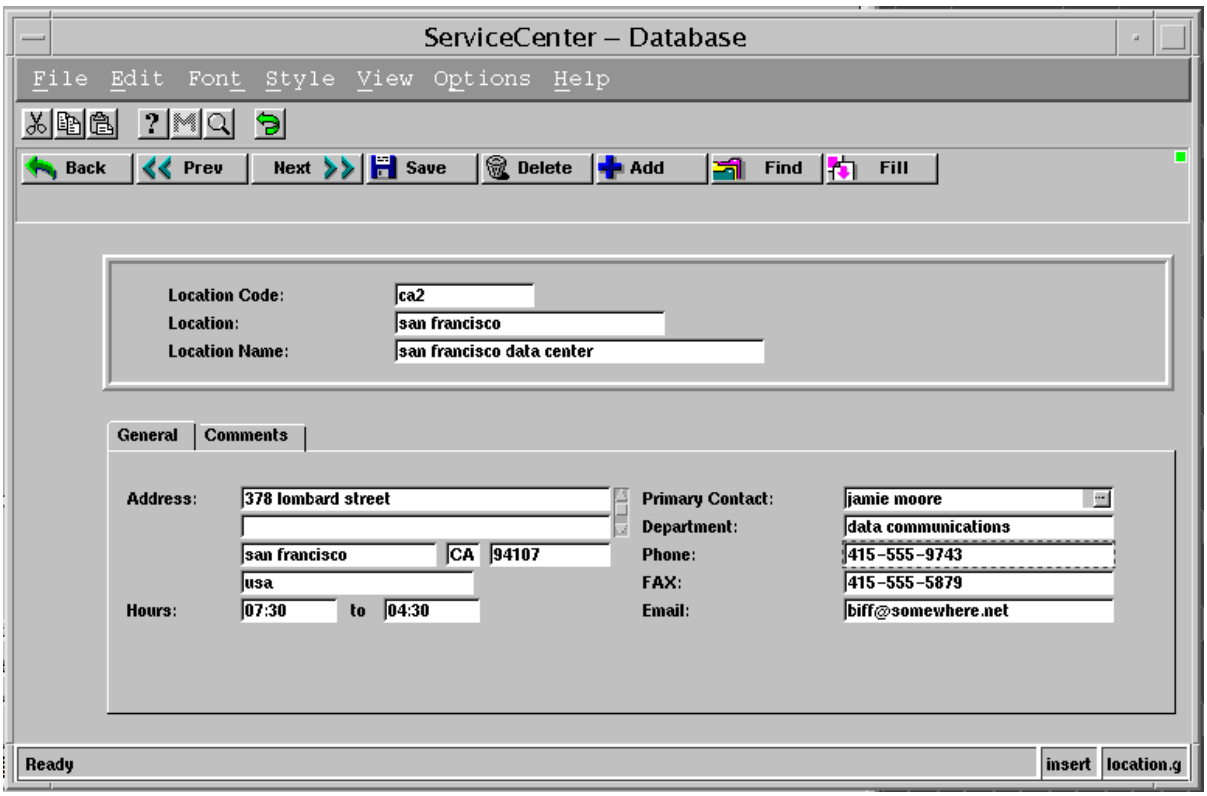

Figure 4-12. Locations Screen

Vendor Information

The ***Vendor Information*** menu option takes you to the ServiceCenter *vendors* table (Figure 4-13). When Vendor Information is first accessed, a blank **vendor** screen is displayed.

- Press **Enter** to display a vendor summary list.
- **Double-click** on the desired vendor to access the data for that vendor. The **vendor** screen is displayed with the ServiceCenter information for that vendor. These fields can be edited and updated.

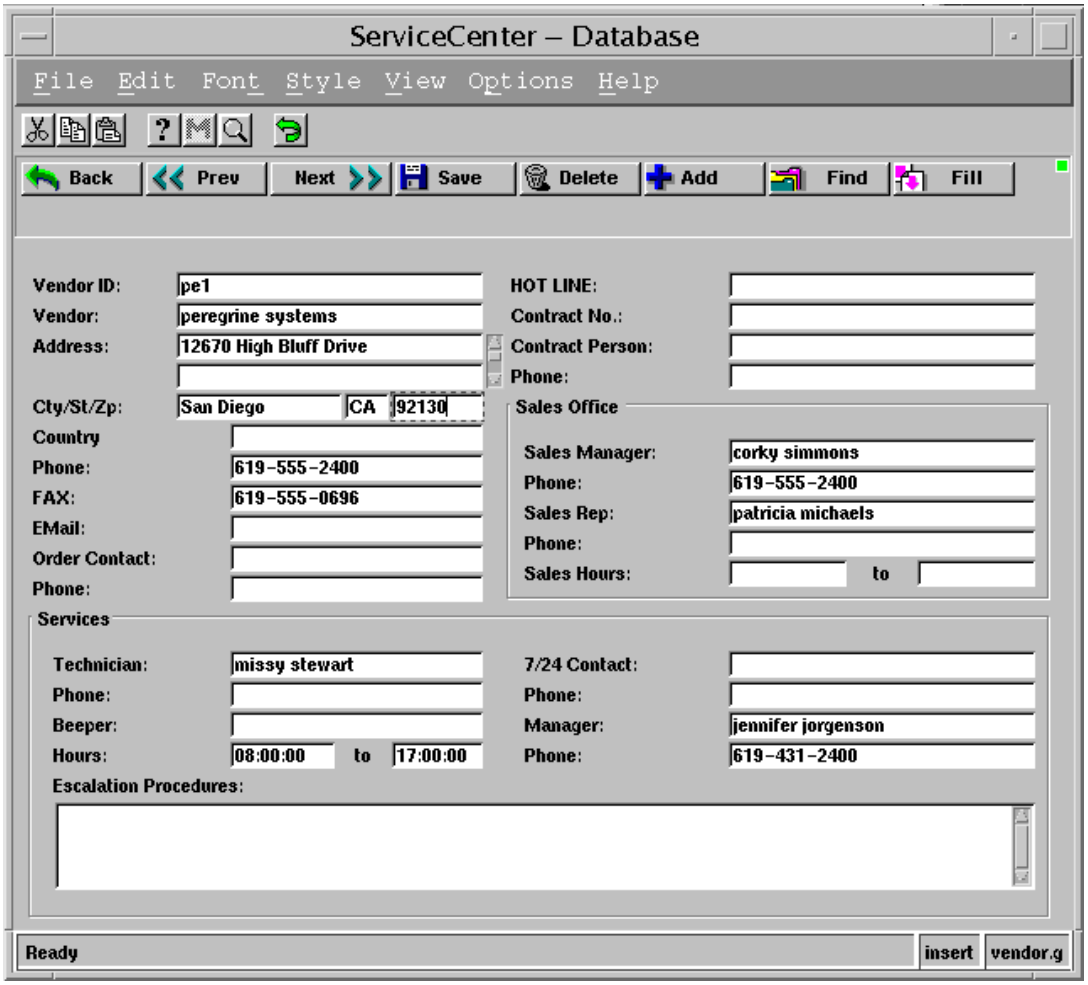

Figure 4-13. Vendors Screen

User Directory

The ***User Directory*** menu option allows you to add or query for a particular user. When ***User Directory*** is first accessed, a blank *user directory* ServiceCenter screen appears (Figure 4-14). To query for user information, enter known data in the appropriate field and click on the **Find** button.

- To get a user list, press **Enter** after the blank screen appears. A user list is displayed.
- **Double-click** on the desired user to get the User Directory information for that user.

You can also use the ***User Directory*** option to add or update user information in the ServiceCenter User Directory.

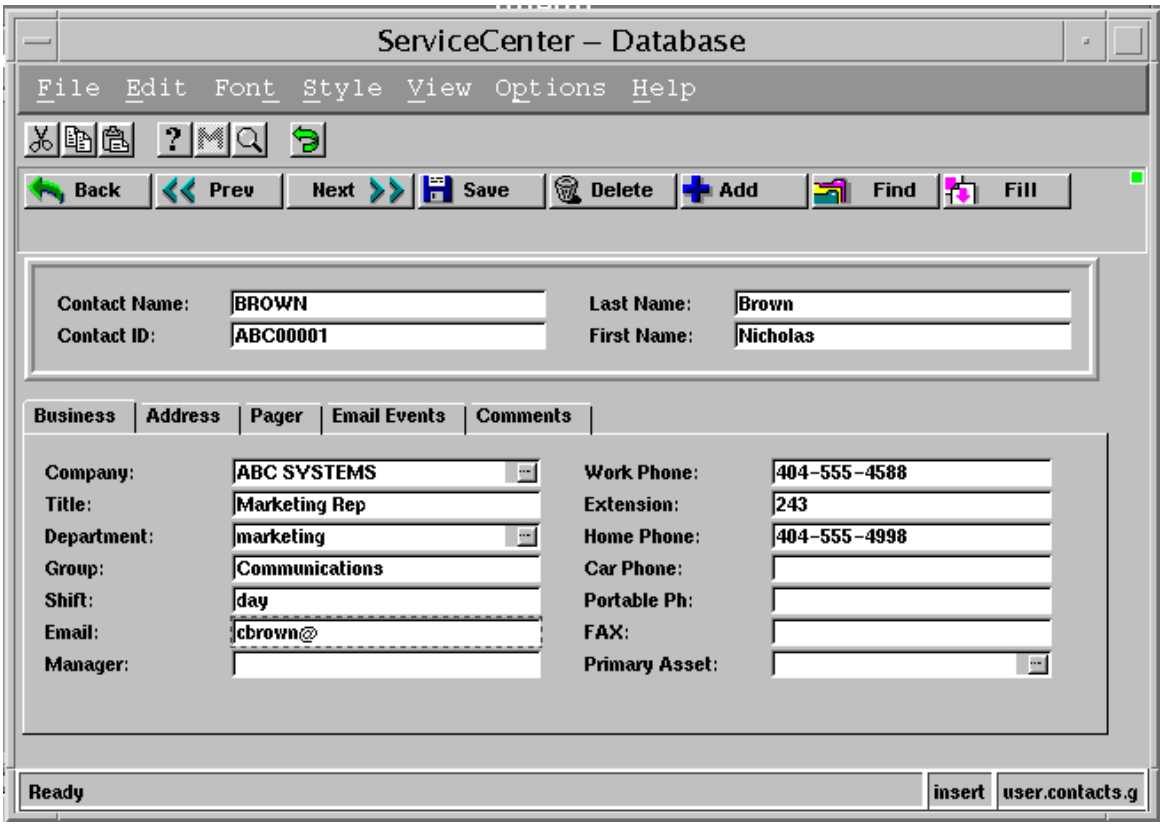

Figure 4-14. User Directory Screen

Filtering

The ***Filtering*** menu option takes you to the Event Services Filters screen (Figure 4-15). In this screen, you can set ServiceCenter and SCAuto event filters, or query for an existing filter. All fields in the setup screen are optional, therefore you can either set one field, all fields, or a combination of fields. This provides flexibility in creating filters.

Multiple filters can be set to seek problems under different conditions. Refer to the Event Services documentation for more information on the Event Manager application.

The filter setup screen contains the following fields:

Event Type Allows you to specify an existing or custom event code to define the filter.

User Name Allows you to specify the user name as defined in the **evuser** field in the event record. A blank user name will match any user name.

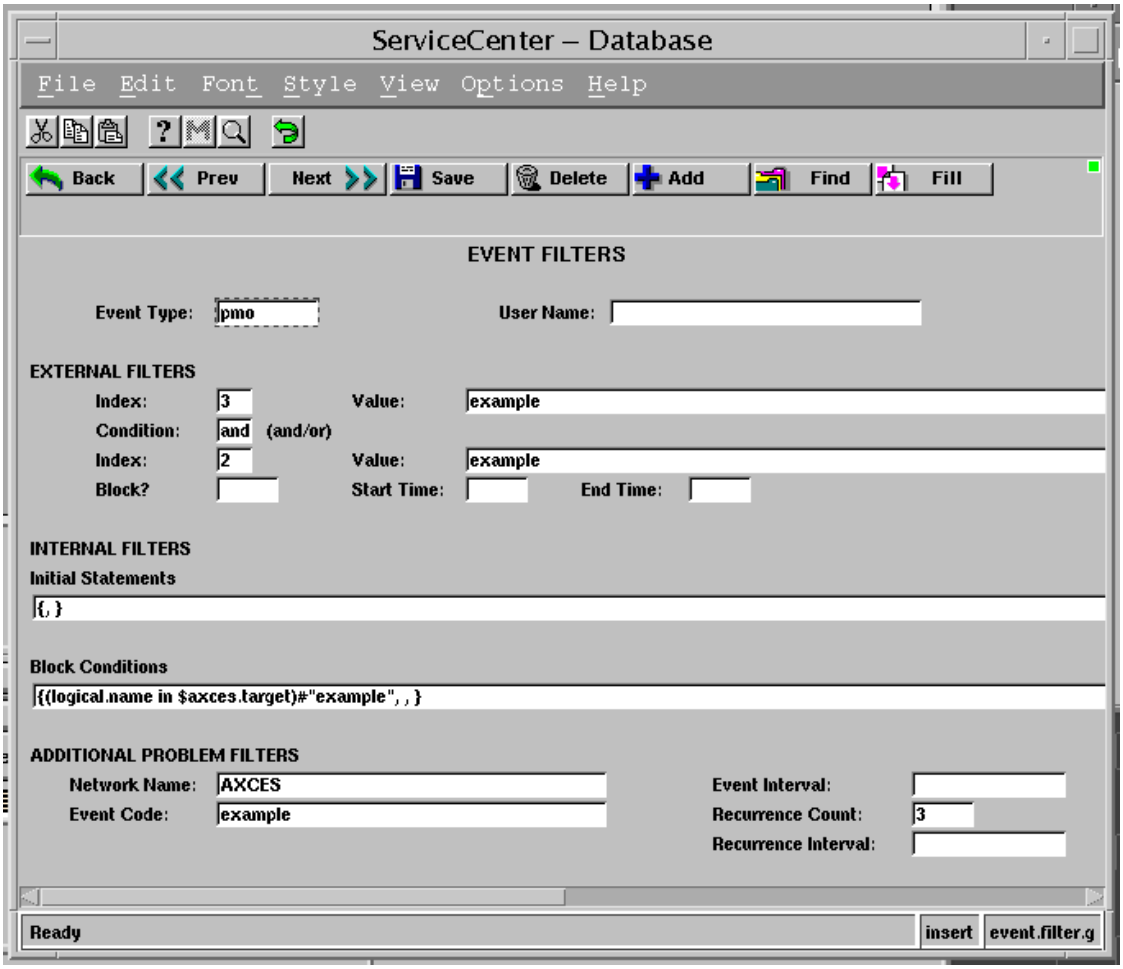

Figure 4-15. Filtering Screen

EXTERNAL FILTERS

Allows you to specify the filtering based upon the **evfields** in the Event Record structure. These fields provide specific separator characters that divide them into subfields.

Index

Allows you to specify the subfield in the **evfields** that is to be read by the filter. In Figure 4-15, **4** represents the fourth field in the record. One or two indexes can be defined for a filter.

Value

Allows you to specify the value the filter is to compare to to thesub field specified by the index. In Figure 4-15, the value **6** represents the generic SNMP trap code the filter is to look for. The value **58916865** is the specific SNMP trap code the filter is to look for.

Condition

Allows you to specify a relational operator, **and** or **or**, if a second index and value are to be used in the filter.

Block

Enter **true** to completely block the event. Enter false to allow the **Recurrence Count** to take effect.

Start Time

Enter a time for the filter to begin monitoring or block the alert specified by the filter. The format for the **Start Time** field is **hh:mm**.

End Time

Allows you to enter a time for the filter to stop monitoring for the alert specified by the filter. The format for the **End Time** field is **hh:mm**.

Note: If the times are not specified, then the filter continuously remains in effect.

Note: For best performance, only the above fields should be used. The fields below will cause additional event processing overhead when used.

Network Name

Specify the part of the system you want the filter to be applied. For a system-wide filter, enter **AXCES** in this field. For a specific host enter the host name.

Event Code

Enter an SNMP trap code for the filter to search for. If you are not familiar with SNMP trap codes, refer to SNMP documentation for information. The values in this field will be matched against the SNMP trap codes for incoming problem events.

Event Interval

Allows you to specify a time period an event is active before a problem is opened if the filter condition occurs. The format for the **Event Interval** field is **hh:mm:ss**.

Recurrence Interval

Allows you to specify a time period to open a problem if the filter condition occurs. The format for the **Recurrence Interval** field is **hh:mm:ss**. This parameter is used in conjunction with the **Recurrence Count**.

Recurrence Count

Allows you to set the number of alerts the must occur for the filter before a problem is opened. If the **Recurrence Count** is used in conjunction with the **Recurrence Interval**, a problem is opened if the count value is reached in the set interval.

If **Recurrence Count** is used without a time period set in the **Recurrence Interval**, then the count continues over an indefinite period, while the filter is active.

The count is reset to zero (0) if a problem is opened by the filter.

When the filter is configured, click on the **Add** button.

Help Desk

The ***Help Desk*** menu option takes you to a problem summary screen for the current NetView operator (Figure 4-16). For ServiceCenter 1.4, this takes you to the main **Help Desk** screen.

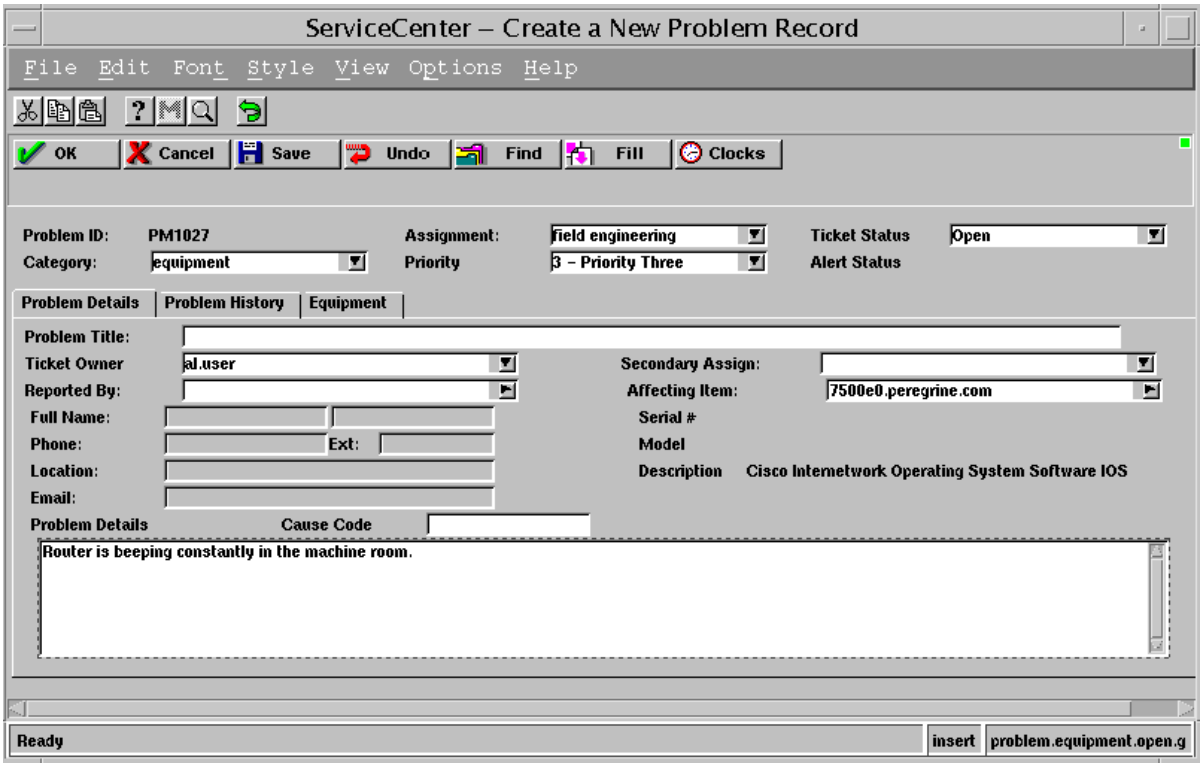

Figure 4-16. Help Desk Screen

Main Menu

The ***Main Menu*** option takes you to the main ServiceCenter screen (Figure 4-17). From here, you can access all ServiceCenter functions

Figure 4-17. ServiceCenter Main Menu

Chapter 5 Troubleshooting

11/13/97

Support Information

This chapter provides the information necessary to obtain Peregrine support for the SCAuto for NetView product.

General Problem Isolation

Some common problems can occur when executing SCAuto for NetView for the first time, when changing platforms, etc. Use the following suggestions to isolate, or fix your problem prior to notification:

When any problem occurs:

1. Check the *scauto.log* for any errors, warnings, and other informative error messages.
2. Try running "**scdiscover -test**". This verifies basic connectivity and *scauto.ini* setup.
3. In some cases, system messages are lost when SCAuto monitors are executed under the Netview Process Manager. Run the monitors from a command line.
4. Run the background monitors or utilities with the **-debug** option. This produces additional log information.
5. Ensure permissions in your installation directory are correct, and that the executable commands have execute permissions. *Scevmon* should be set-uid. All files should be owned by the same user id.

If the verification test fails during connection between the SCAutomate Base (scautod) and the SCAuto for NetView (scdiscover, scevmon) please check the following common failures:

6. Is the SCAuto server (*scautod*) running on ServiceCenter Server Platform? Use ServiceCenter *status* command to check. If it's ***not running*** start it from the status option *start schedulers* selecting *scauto.startup*. If the start fails the scautod server logs messages to the *sc.log* file, so be sure to check the log for any error messages.

   Check the *scauto.ini* file for the **scauto:** keyword. This should use a service name *other* than that used for the ServiceCenter server. Ensure that the service name or port number matches that used by the SCAuto server exactly. Ensure that a hostname was given as well if the SCAuto server is running on a different machine.

Check that the service name is defined in your */etc/services* file, or is available from your NIS server. Check with your network administrator if you are unsure of this.

7. On the ServiceCenter server platform, check that the **scauto:** keyword is specified correctly in the *sc.ini* file. If none is specified, or given on the command line, the default service name of *scauto* will be used.
8. Is there connectivity between SCAuto server (*scautod*) and the SCAuto for NetView platform? *Ping* the SCAuto server platform from the NetView platform. If this fails and all TCP/IP specifications are correct contact your network administrator for assistance.
9. Are the SCAuto for NetView background monitors running? Use the NetView **ovstatus** command to check for the *scevmon* and *sclogger* processes. If they are not running, review the status message from **ovstatus**. These monitors are dependent upon NetView processes, so ensure that all other NetView processes started correctly.

If you have an event in the* event.in *file but no RAD application is invoked (e.g. pmo event and no problem opened) check the following:

1. Is the Event Scheduler running on the ServiceCenter server platform? Use ServiceCenter *status* command to check. If it's ***not running*** start it from the status option *start schedulers* selecting *event.startup*. If the start fails review error messages and retry. If errors persist call Peregrine Customer Support. If it **is running** and not processing, stop the Event Scheduler and build a new *schedule record* and restart.
2. Review **Basic Trouble Shooting** section in *Event Services Guide* in regard to schedule record specifications

No event created in* event.out *file but the RAD application has been invoked (e.g. a problem was opened and no pmo output event created) check the following:

1. Refer to *Using Format Control to Write Eventout Records* section in the Event Services documentation. Ensure the the Problem Management application has been set up correctly to create events.
2. Review the **Basic Trouble Shooting**, and relevant sections of *Event Services Guide* for output event generation (e.g.**Writing Eventout Records from Problem Management**).

If no traps are being discovered (scauto.traps is non-existent or empty):

1. Ensure that *sclogger* is running. Use the NetView **ovstatus** command to check on its status.
2. Compare the *scauto.traps* to the NetView *trapd.log*. The *scauto.traps* should contain all log records from the *trapd.log* that *scauto.keywords* specified. Review the keywords to ensure they are correct, as they may filter out more problems than expected.

Contacting Peregrine Systems

Peregrine Systems Inc. provides support for all SCAuto users. Before contacting Peregrine Customer Support, review the following section, *Obtaining Required Data/Information*, to see if additional data is required to help diagnose the problem.

You can contact Peregrine Systems support as follows:

- For SCAuto for NetView information or problems that is needed immediately, call Peregrine Customer Support at (800) 638-5231 or (619) 481-5000.
- For questions or information regarding SCAuto for NetView, use a written FAX or email.

  Send all FAXes to (619) 481-1751.
- For information that was requested of your installation that is on tape, cartridge, etc., send to:

  Peregrine Systems Inc.

  attn: SCAutomate Support

  12670 High Bluff Drive

  San Diego, CA 92130

Obtaining Required Data and Information

This section provides detailed instructions for gathering data and information needed for the Peregrine support staff to resolve your problem in the most efficient manner possible.

Environmental Information:

- ServiceCenter Release
- SCAuto for NetView Release
- Operating System Release (i.e., HP-UX, SunOS, Solaris, AIX)
- Type of hardware SCAuto is running on
- Any error messages or error logs.

The Peregrine Support staff can utilize the following error logs and files to resolve an IPAS problem:

- *scauto.chk*, *scauto.traps*, and *scauto.keywords*
- *scauto.log*, and any *sc.log* files.
- Any *sc.log*, scheduler logs, or messages from the ServiceCenter server platform.
- The output from an *ovtopodump -lr* command.
- If the problem resulted in a core dump, the resulting core file is helpful in determining the problem.
- Unloaded Event records if they are causing errors.
- Unloaded ServiceCenter event filters if relevant.

Sending Files to Peregrine

Files may be sent to Peregrine Systems either via magnetic tape, FTP, or electronic mail. If multiple files are being sent, store the files in the *tar* format. Compressed files are acceptable.

Please check with your customer support representative for the appropriate format and media to use.

Appendix A Tables

11/13/97

The following tables describe the mapping of discovered inventory onto ServiceCenter **icma** events. All eventmap fields belong to the **device** file except those marked with a **"*"**.

IP Inventory Mappings

IP inventory is discovered through the NetView command **"ovtopodump -lr**".

TABLE 1. IP Inventory mappings for **Network** devices

eventmap field	ovtopodump field	SCAuto supplied value
type		"Network"
logical.name	NETWORK NAME	
network.name	IP ADDR	
protocol		"Internet Protocol"
description		"Network specification"
location		Default location
protocol addr	NETWORK NUMBER	
last.update	MODIFIED TIME	
updated.by		"OV-*hostname*"
icount	NUM SEGMENTS	

TABLE 2. IP Inventory mappings for **NetworkSegment** devices

eventmap field	ovtopodump field	SCAuto supplied value
type		"NetworkSegment"
logical.name	SEGMENT NAME	
network.name	SEGMENT NAME	
subtype	FLAGS	
protocol		"Internet Protocol"
description		"Network segment specification"
location		Default location
last.update	MODIFIED TIME	
updated.by		"OV-*hostname*"
parent		Parent Network logical name
icount	NUM NODES	

TABLE 3. IP Inventory mappings for **node** devices

eventmap field	ovtopodump field	SCAuto supplied value
type		"node"
logical.name	HOSTNAME	
network.name	HOSTNAME	
subtype	FLAGS	
protocol		"Internet Protocol"
description	DESCRIPTION	
location	LOCATION	
contact.name	CONTACT	
vendor	NODE VENDOR	
last.update	MODIFIED TIME	
updated.by		"OV-*hostname*"
objid	SNMP OBJECT ID	
protocol.addr	SNMP ADDRESS	
icount	NUMBER OF INTERFACES	

TABLE 4. IP Inventory mappings for **interface** devices

eventmap field	ovtopodump field	SCAuto supplied value
type		"interface"
logical.name		"Interface:*ip-addr*"
network.name	IP ADDR	
protocol		"Internet Protocol"
description	INTERFACE	
location		Same as container node
model	IF TYPE	
last.update	MODIFIED TIME	
updated.by		"OV-*hostname*"
protocol.addr	IP ADDR	
network.address	PHYSICAL ADDRESS	
parent		Parent segment logical name
container		Container node logical name

SNA Inventory Mapping

SNA Inventory is discovered through the OpenSNA command "**snaprint -R**"

.

TABLE 5. SNA Inventory mappings for all devices

eventmap field	sna_print field	SCAuto supplied value
type	TYPE	
logical.name	NAME	
network.name	NAME	
protocol		"Systems Network Architecture(SNA)"
domain	DOMAIN	
location		Default location
last.update	UPDATED	
updated.by		"OV-*hostname*"
parent	PARENT	
container	PEER	
protocol.addr	SUBAREA	
icount	CHILDREN	

Novell Inventory Mapping

Novell inventory is discovered through the ServerView "**siget**" command (which is called from the SCAuto for NetView "**svvget**" command script). There are several options to **siget**. For Novell servers, the "**svvsum**" option is used for discovering servers, workstations, and interfaces. For Novell clients, the "**stsw**", "**sthw**", "**stnet**", and "**svsernum**" options are used for discovering workstations and interfaces.

Note: Eventmap fields marked with a "*" belong to the "*server*" or "*workstation*" files instead of the default "*device*" file.

TABLE 6. Novell Inventory mappings for **Network** devices

eventmap field	siget field	SCAuto supplied value
type		"Network"
logical.name		"Novell.*protoaddr*"
network.name		"Novell.*protoaddr*"
protocol		"IPX"
description		"Network specification"
location		Default location
protocol addr	ID String ; Local Network	
last.update		Current time
updated.by		"OV-*hostname*"

TABLE 7. Novell Inventory mappings for **Server** devices

eventmap field	siget field	SCAuto supplied value
type		"server"
logical.name		"SV.*servername*"
network.name		"SV.*servername*"
protocol		"IPX"
description	Netware Revision	
location		Default location
protocol addr	Internal Net Number	
last.update		Current time
updated.by		"OV-*hostname*"
container		Container workstation logical name
serial.no.	NetWare Serial Number	
*printers	Print Queues	
*servers	Print Servers	
*media	NetWare Volumes	

TABLE 8. Novell Inventory mappings for **Server Workstation** devices

eventmap field	siget field	SCAuto supplied value
type		"workstation"
logical.name		"WS.*servername*"
network.name		"WS.*servername*"
protocol		"IPX"
description		"Novell server workstation"
location		Default location
last.update		Current time
updated.by		"OV-*hostname*"
*processor	Processor Type	
*memory	Memory Installed	
*media	Floppy Drive, Hard Drive	
*local.software	Loaded NLMs	
*adapter	Physical Board	

TABLE 9. Novell Inventory mappings for **Server Interface** devices

eventmap field	siget field	SCAuto supplied value
type		"interface"
logical.name		"*servername.protoaddr*"
network.name		"*servername.protoaddr*"
protocol		"IPX"
description	Frame Type	
location		Default location
model	Board Name	
protocol addr	ID String	
network.address	Node (MAC) Address	
last.update		Current time
updated.by		"OV-*hostname*"
parent		Parent network logical name
container		Container workstation logical name

TABLE 10. Novell Inventory mappings for **Client Workstation** devices

eventmap field	siget field	SCAuto supplied value
type		"workstation"
logical.name		"WS.*stationid*"
network.name		"*stationid*"
protocol		"IPX"
description		"Novell client workstation"
location	via "svsernum"	
id *(asset number)*	via "svsernum"	
vendor	via "svsernum"	
model	via "svsernum"	
contact.name	via "svsernum"	
serial.no.	via "svsernum"	
last.update		Current time
updated.by		"OV-*hostname*"
*processor	Processor Type	
*math	Math Coprocessor	
*bios	BIOS	
*operating.system		"DOS"
*os.version	DOS Version	
*memory	Memory Installed	
*media	Floppy Drive, Hard Drive	
*drivers	Device Drivers	
*local.software	Programs and TSRs	
*boot.files	Boot Files	
*adapter	Network Adapter	

TABLE 11. Novell Inventory mappings for **Client Interface** devices

eventmap field	siget field	SCAuto supplied value
type		"interface"
logical.name		"Intf.*stationid.protoaddr*"
network.name		"Intf.*stationid.protoaddr*"
protocol		"IPX"
description	Description (via "stnet")	
location		Default location
model	Network Adapter	
protocol addr	Local Network	
network.address	Local Node MAC Address	
last.update		Current time
updated.by		"OV-*hostname*"
parent		Parent network logical name
container		Container workstation logical name

Appendix B IPAS Conversion

11/13/97

SCAuto for NetView is a replacement for IPAS. No IPAS functionality will be lost by replacing it with SCAuto.

Here is a summary of differences between IPAS and SCAuto for NetView. Most portions of IPAS have been examined and rewritten for SCAuto.

- Most file names have changed, as shown in the following table:

IPAS file name	SCAuto file name	Comments
ovIPAS.trapd.log	scauto.traps	
ovIPAS_Auth	scevauth	
ovIPAS_Siget	svvget	
ovIPAS_arf	scauto.reg	
ovIPAS_chkpt	scauto.chk	merged with ovIPAS_seq
ovIPAS_keywords	scauto.keywords	
ovIPAS_log	scauto.log	
ovIPAS_msgtable	scauto.msg	
ovIPAS_seq	scauto.chk	merged with ovIPAS_chkpt
ovIPASarc	scarchive	
ovIPASd	scevmon	
ovIPASd_lrf	scevmon.lrf	
ovIPASpnms	scelogin	
ovIPASr	scdiscover	
ovIPAStd	sclogger	
ovIPAStd_lrf	sclogger.lrf	
sc.ini	scauto.ini	IPAS configuration only. ServiceCenter configuration is left in sc.ini.

- No longer based on ServiceCenter clients, but uses SCAuto protocol. Results in smaller binaries, better error reporting, and less overhead.
- Keywords file allows comments and has more robust parsing. Wildcards are allowed in keywords.
- Can be installed into any directory.
- Has own *scauto.ini* configuration file.
- Can specify a starting node for inventory discovery.
- Event monitor does not do a “cold start” when first run. Instead use scdiscover to gather initial inventory.

Migration Notes

Here are some key points to be aware of when migrating from IPAS.

- During SCAuto for NetView installation, the *INSTALL* script can detect if there is an existing IPAS installation on the current platform. If so, the *INSTALL* script will ask the user if he or she wishes to build an SCAuto checkpoint file from the existing IPAS checkpoint files, and also if IPAS should be unregistered from NetView.

  This is the only automatic conversion that will be done. The original IPAS directory will not be removed.

- Ensure that a SCAuto base server is running.
- Place the correct SCAuto base server specification into your scauto.ini configuration file. *Do not use the ServiceCenter server specification that you had in sc.ini!* SCAuto and ServiceCenter do not use the same TCP/IP port.
- The IPAS *sc.ini* will need to be examined and any "#@" parameters should be converted into the appropriate *scauto.ini* parameter. See the installation chapter for more details.
- If you were using IPAS version 1.4, then you can copy *ovIPAS.trapd.log* over into the *scauto.traps* file in the SCAuto installation directory. IPAS 1.4 and SCAuto for NetView use the same format for the trap files.
- If you had a custom *ovIPAS_Auth* script, copy this over the *scevauth* script in the SCAuto installation directory.
- If you customized the *ovIPAS_Siget* script, copy this over the *svvget* script in the SCAuto installation directory.
- You may need to customer scauto.msg. Do **not** copy *ovIPAS_msgtable* on top of *scauto.msg*, these files do not use the same format!
- You may need to customize the scauto.keywords to match any IPAS customizations. The default *scauto.keywords* file is different from the default *ovIPAS_keywords* file, so you should examine the new entries.
- You may run IPAS and SCAuto for NetView concurrently while migrating. You may wish to use the **eventsuffix:** option in *scauto.ini* temporarily to ensure you don't get duplicate events.

Index

T

U

V