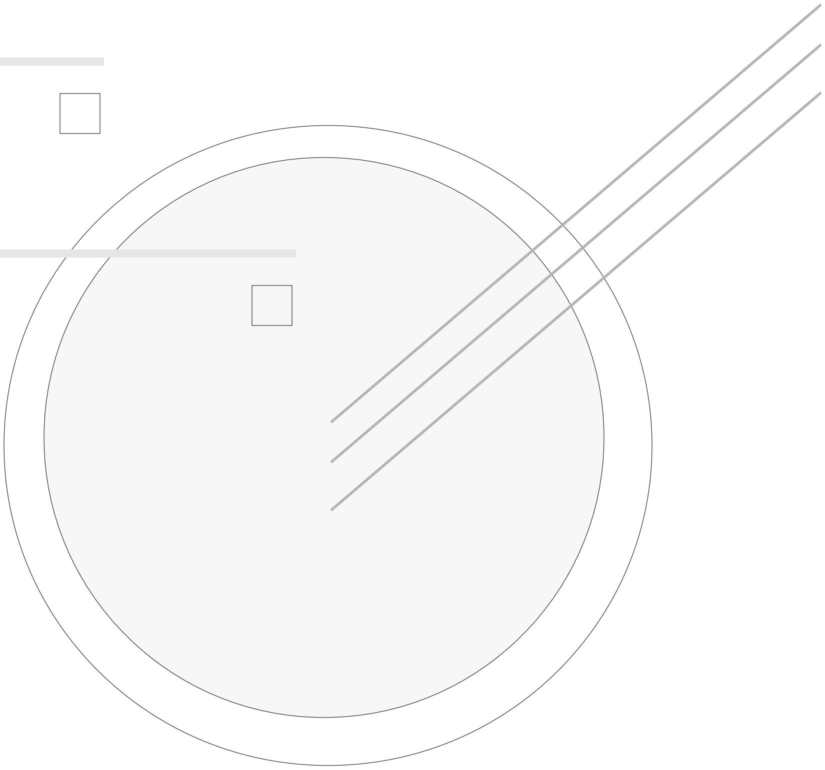
Cooperative Consoles 1.1
Administrator's Guide



2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

Part No: 802-4501-10
Revision A of September 1995

SunSoft
A Sun Microsystems, Inc. Business

Contents

Figures

Tables

Preface

The *Cooperative Consoles 1.1 Administrator's Guide* provides information on the functions and features of Cooperative Consoles 1.1 for SunNet Manager.

Who Should Use This Book

This document is intended for network administrators who need to set up and configure Cooperative Consoles for information sharing between SunNet Manager Console instances running on multiple hosts.

Installation Information

For information on installing Cooperative Consoles software, refer to the *Cooperative Consoles 1.1 Installation Guide.*

How This Book Is Organized

This document is organized as follows:

Chapter 1, "Introduction," provides an overview of the components of Cooperative Consoles.

Chapter 2, "Cooperative Consoles Configurations," describes a number of possible configurations in the information forwarding relationships among SunNet Manager Consoles with Cooperative Consoles installed.

Chapter 3, “Cooperative Consoles Operation,” describes the functioning of Cooperative Consoles and its underlying architecture.

Chapter 4, “Using the Configuration Tool,” describes the use of the Configuration Tool for customizing the sharing of information between management stations.

Compatibility

See the *Cooperative Consoles 1.1 Important Product Information* (IPI) for compatibility information.

Conventions Used in This Book

Command Line Examples

All command line examples in this guide use the C-shell environment. If you use either the Bourne or Korn shells, refer to `sh`(1) and `ksh`(1) man pages for command equivalents to the C-shell.

What Typographic Changes and Symbols Mean

The following table describes the type changes and symbols used in this book.

Table P-1 Typographic Conventions

Typeface or Symbol	**Meaning**	**Example**
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `system% You have mail.`

`AaBbCc123`	What you type, contrasted with on-screen computer output	`system%` **`su`**
		`Password:`

<AaBbCc123>	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *<filename>*`.`

Table P-1 Typographic Conventions

Typeface or Symbol	**Meaning**	**Example**
AaBbCc123	Book titles, new words or terms, or words to be emphasized	These are called *class* options. You *must* be root to do this.
Code samples are included in boxes and may display the following:		
`%`	UNIX C shell prompt	`system%`
`$`	UNIX Bourne and Korn shell prompt	`system$`
`#`	Superuser prompt, all shells	`system#`

Mouse Conventions

This book assumes that you are using a standard Sun workstation three-button mouse. The mouse buttons are called SELECT (left), ADJUST (middle), and MENU (right).

Click means to press and quickly release a mouse button.

Press indicates you should hold the button down until an action is completed, such as a menu appearing.

Introduction 1

Cooperative Consoles 1.0 implements sharing of information between multiple instances of the SunNet Manager Console.

Geographically dispersed organizations with large networks often have the need for a division of network management responsibilities among multiple management Consoles. For such network environments, Cooperative Consoles (CC) provides the ability to implement forwarding of information about selected changes in the state of critical network devices or changes in selected aspects of network topology between multiple management stations running SunNet Manager. CC provides the flexibility to implement a variety of possible information-sharing configurations between multiple SNM Consoles.

Note – Cooperative Consoles presupposes the installation of the 2.2.1 or later version of SunNet Manager for management stations on the Solaris 1.x environment and the installation of the 2.2.2 or later version of SunNet Manager on management machines running under the Solaris 2.x environment.

CC currently enables the forwarding of three types of information between SNM Consoles:

- **SNM Events** — These are generated by agents in response to SNM event requests when conditions specified by the event request (such as a device being unreachable) are satisfied.

- **Traps** — These are unsolicited events, not generated in response to SNM event requests; for example, a Simple Network Management Protocol (SNMP) `linkDown` trap.
- **SNM database traps** — The SNM Console generates traps when changes are made to the SNM database, such as addition of a new element or loading of a background image for a view.

To implement information forwarding between SNM Consoles, Cooperative Consoles uses three executable software modules:

- **Receiver Application** — A Receiver is to be installed on each SunNet Manager Console machine that is to receive forwarded event and topology information from other SNM Consoles. An SNM Console machine with a Receiver process running is a *receiving station.*
- **Sender Daemon** — A Sender daemon is to be installed on each SunNet Manager host that is to forward event and topology information to remote SNM Consoles. An SNM Console machine with a Sender process running is a *sending station*.
- **Configuration Tool** — This is the user interface for configuring operation of the Sender and Receiver processes on the local host.

The operation of the Receiver and Sender processes is described in Chapter 3, “Cooperative Consoles Operation.” The use of the Configuration Tool is explained in Chapter 4, “Using the Configuration Tool.”

A SunNet Manager Console machine may function as both a sending station and a receiving station, or it may serve as only a sending station, or as only a receiving station. The particular distribution of the Sender and Receiver on the SunNet Manager Console machines in your network depends upon the desired network management configuration. Several sample configurations are discussed in Chapter 2, “Cooperative Consoles Configurations.”

In addition to the Receiver and Sender processes, the SunNet Manager Event Dispatcher (`na.event`) also plays an important role in the functioning of Cooperative Consoles on both receiving and sending stations.

The information flow between SNM Consoles with Cooperative Consoles installed is illustrated in Figure 1-1.

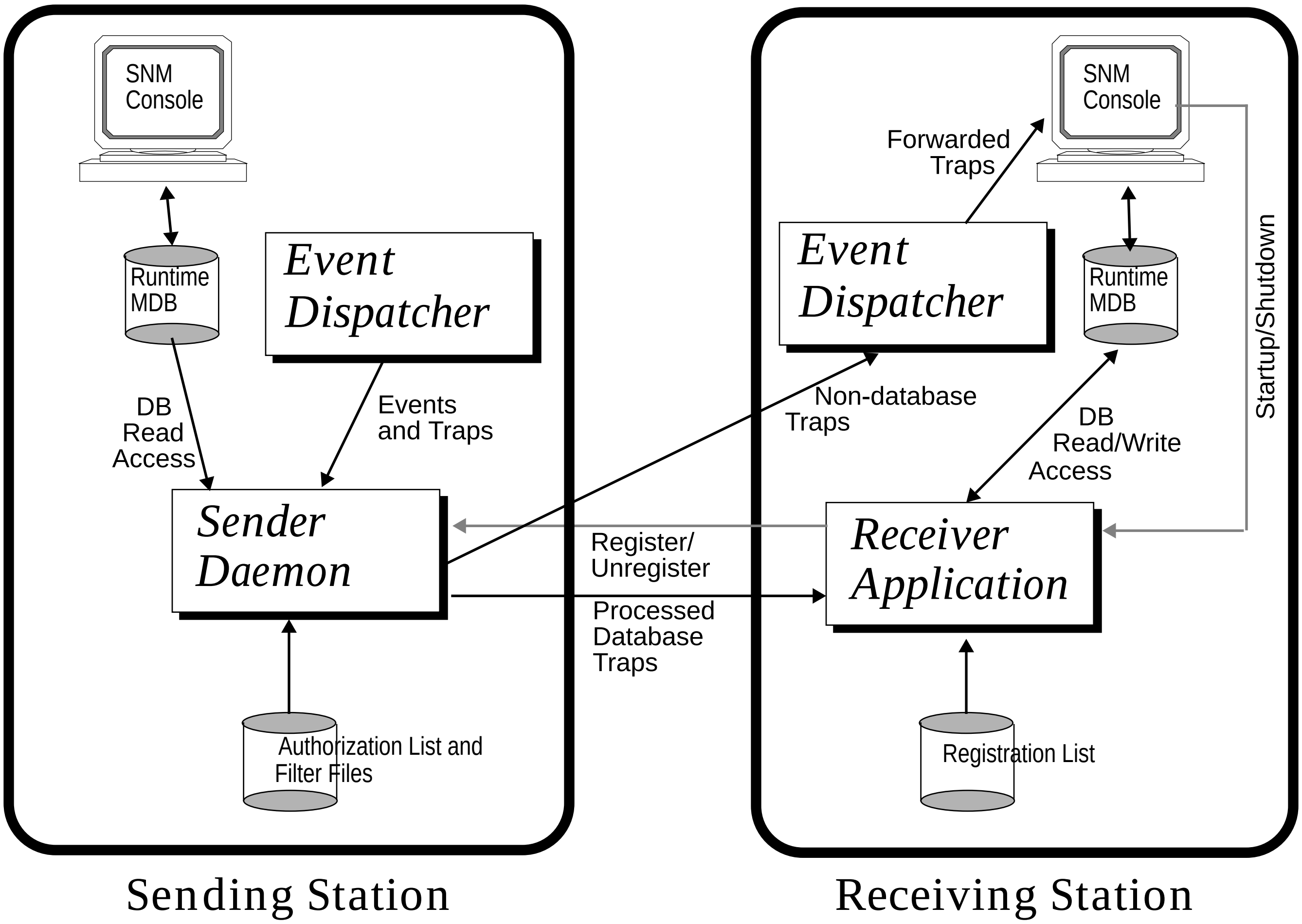


Figure 1-1 Cooperative Consoles Operation

1.1 Adding Cooperative Consoles to the SNM Console Tools Menu

After installation of the Cooperative Consoles software, you will need to load the `cooptools.schema` file into SunNet Manager in order to add the Cooperative Consoles Receiver and Configuration Tool to the SNM Console's Tools menu. There are two methods for loading the `cooptools.schema` file:

- **Restart SunNet Manager with the `-i` option.**

If you start SNM with the `-i` option after installing Cooperative Consoles, this forces SNM to reload all the schema files, including `cooptools.schema`.

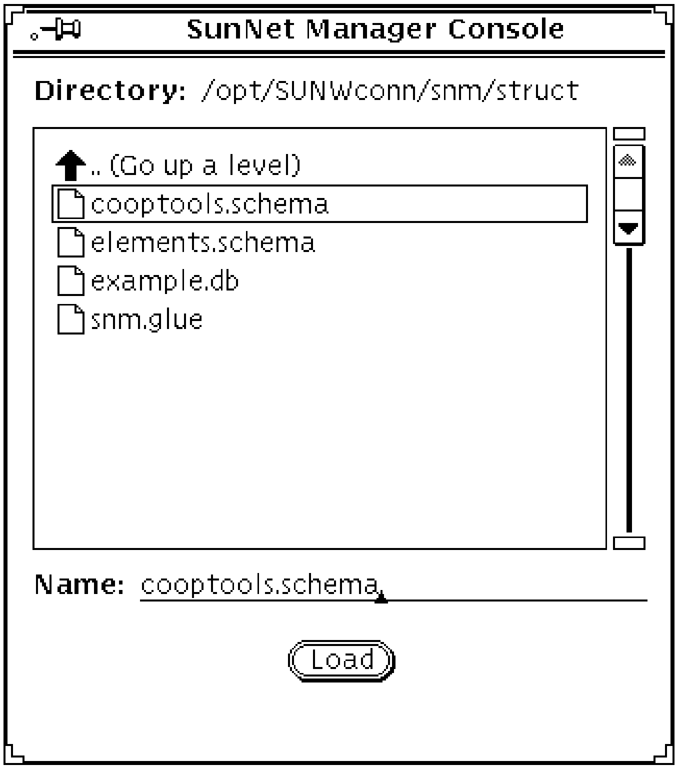
Note – Starting SNM with the `-i` option erases your runtime management database. If you want to save your current runtime database, you will need to use the Console File menu Save➤Management Database option to save your runtime database to an ASCII file. You can then reload this database using the Load➤Management Database option after restarting SNM with the `-i` option.

- **Load the `cooptools.schema` file from the SNM Console's File menu and then restart SNM.**

If you want to use this method, pull down the SNM Console's File menu and select the Load➤Management Database option. The `cooptools.schema` file is located in the following directory:

- `/opt/SUNWconn/snm/struct` for Solaris 2.x environments
- `/usr/snm/struct` for Solaris 1.x environments

Select the `cooptools.schema` file, as shown in Figure 1-2, and then click SELECT on the Load button.



Directory: /opt/SUNWconn/snm/struct

.. (Go up a level)
cooptools.schema
elements.schema
example.db
snm.glue

Name: cooptools.schema

Figure 1-2 Loading the cooptools.schema file

After loading the `cooptools.schema` file, you will need to quit SNM and then restart SNM again. When you restart SNM, you do not need to use the `-i` option.

1.2 SNM Database Trap Forwarding

To ensure that *all* SunNet Manager database traps are sent to the local Sender daemon, you will need to make sure that the attribute `snm.console.DBMgrTrapAlways` is set to TRUE in your `.SNMdefaults` file. The default setting for this SNM attribute is FALSE. You can use `vi` or your favorite text editor to change the setting of this attribute. If there is no

`.SNMdefaults` file in your home directory, you can force SNM to create one by making a minor change to some Console attribute from the Console Properties menu.

Cooperative Consoles Configurations

2

Cooperative Consoles (CC) provides the flexibility to implement a variety of possible cooperative relationships between multiple SunNet Manager Consoles. Three possible configurations are described below.

2.1 Peer-to-Peer Forwarding

A peer-to-peer relationship exists among SunNet Manager Consoles when the SNM machines both send and receive event and topology information to each other. The peer-to-peer configuration implies that each SNM Console machine has both a Sender and Receiver software installed, and thus functions as both a receiving and sending station.The following example illustrates peer-to-peer forwarding relationships.

Company PQR has separate SNM Consoles to manage regional networks. A network management station in San Francisco is responsible for managing a west coast region and a station in New York is responsible for an east coast region. The network manager in New York wants to know about all changes to PQR's backbone network (routers, WAN link) and critical servers (financial database server) in the west coast region. Events are filtered by the Sender on the San Francisco machine on the basis of host name (selecting the servers) and component type (selecting the routers) and forwarded to the SNM machine in New York. The relevant elements reside in a view on the San Francisco machine called “WestNet.” The Receiver process on the New York machine will place forwarded topology information under a view also called “WestNet” on the New York machine.

The network management station in San Francisco also wants to receive this type of information from the network management console in New York. The Receiver process on the San Francisco machine places forwarded topology information under a view called “EastNet,” mirroring the view name on the New York machine. A Sender and a Receiver will therefore be installed on both machines to implement this peer-to-peer relationship. Figure 2-1 illustrates this example.

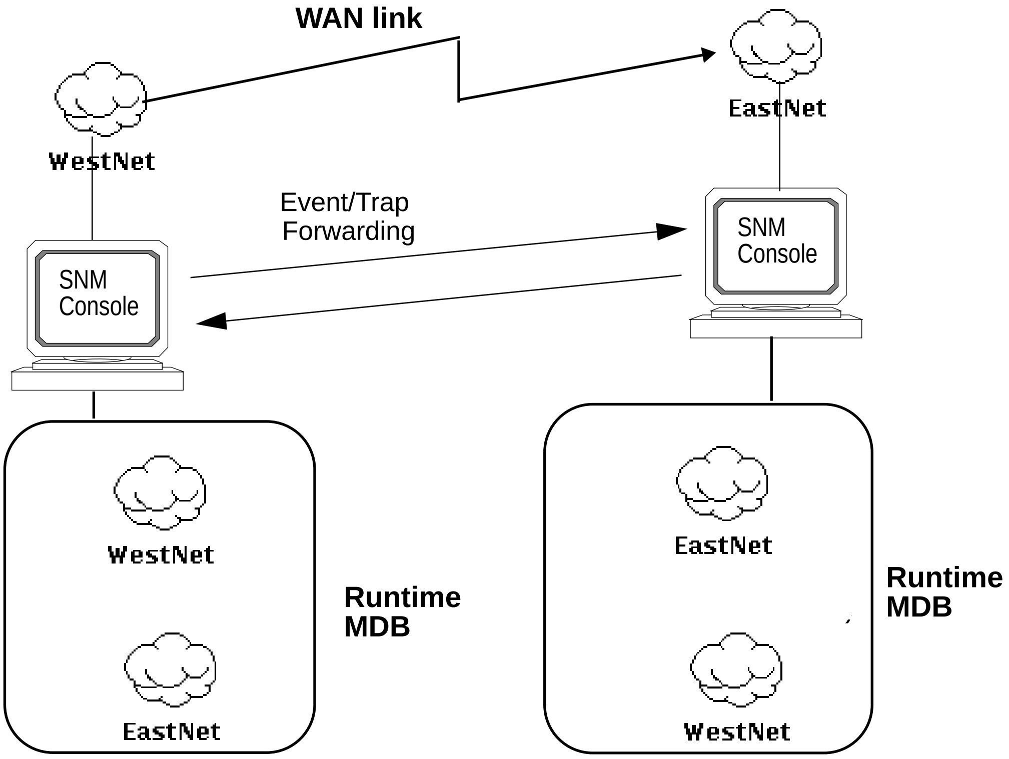


Figure 2-1 Example of Peer-to-Peer Relationship between Regional Stations

2.2 Periphery-to-Center Forwarding

In a periphery-to-center configuration, the flow of forwarded alarm and topology information is from distributed SNM Consoles to a central SNM Console. For example, an organization may have multiple SNM Consoles responsible for regional components of its network while also having a Central SNM Console that needs to display and monitor the global network topology and state of critical devices. The flow of information, in this configuration, is one-directional, from the regional management stations to the central management station. This configuration is illustrated in Figure 2-2.

In this configuration, the Sender daemon only needs to be installed on the regional SNM Console machines. The Receiver only needs to be installed on the central SNM Console machine. Start-up and shutdown of Cooperative Consoles would be initiated from the central SNM Console.

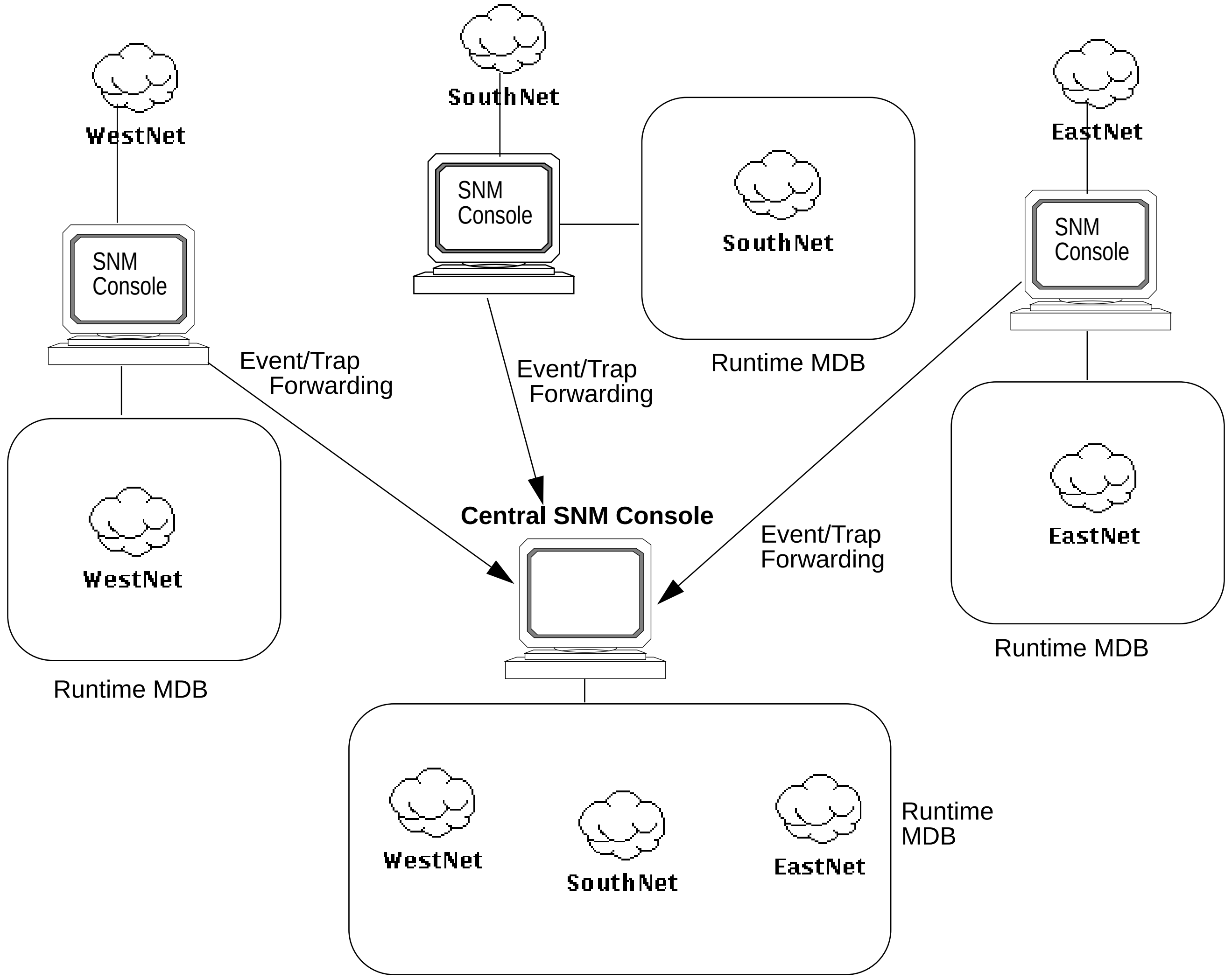


Figure 2-2 Forwarding of Information to Central Management Station

.

2.2.1 Distributed Management with Functional Specialization

A variation on periphery-to-center forwarding of information is a situation where regional SNM Consoles have management responsibility for a type of network link (for example, X.25, Frame Relay, ISDN) or type of network device (routers, database servers, T1 links) for the enterprise-wide network. In this configuration, illustrated in Figure 2-3, there is not a central SNM Console machine that functions as a single network management center. Rather, each SNM Console receives topology and event information from other SNM Consoles about devices of a particular type it is responsible for, in addition to managing its own regional network. Each SNM Console functions as the “center” only for a particular type of network link or type of network device.

As in a peer-to-peer configuration, both a Sender and Receiver is needed on each SNM Console machine in this configuration. The Sender daemon is required to forward information that pertains to the types of devices the other SNM Consoles are interested in. The Receiver application is needed in order to function as a receiving station, to receive information about the particular device types the local SNM Console is managing for the enterprise-wide network.

In the example in Figure 2-3, the Receiver process on SNM#2, which manages the western regional network, would register with the Sender daemons on each of the other SNM Console machines to receive information about X.25 links. Each of the sending stations would need to have a Filter Table available that forwards only X.25-related event and topology information. The Receiver process on SNM#2 would pass the name of this filter file when it registers with the Sender daemons on the other stations. SNM#2 would function as the “central” SNM Console only in regard to the X.25 network links.

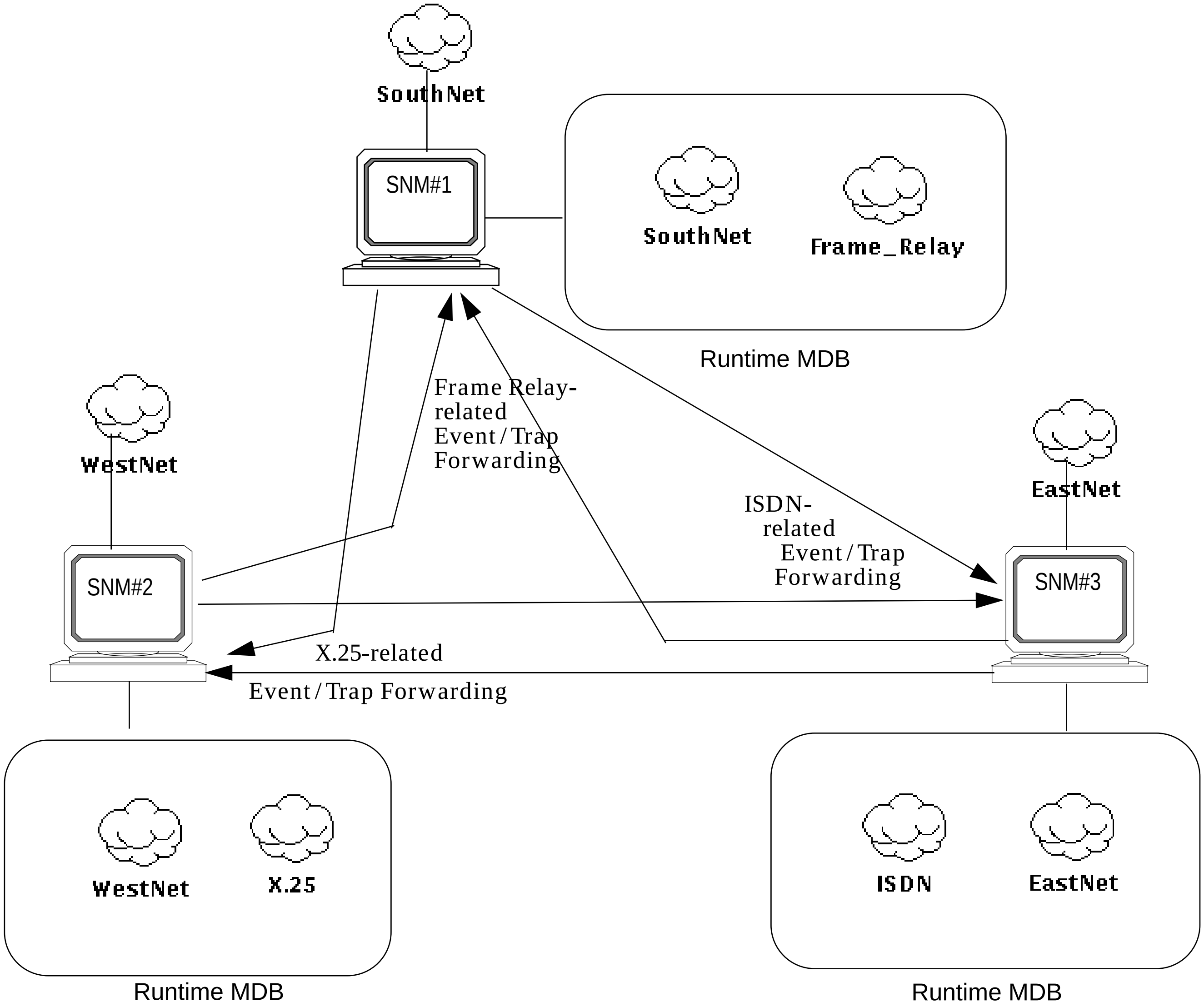


Figure 2-3 Functional Specialization in Multiple SNM Consoles

2.3 Center-to-Periphery Forwarding

In a center-to-periphery configuration, topology changes and alarms are propagated from one or more central SNM Consoles to distributed SNM Consoles in order to off-load responsibility for management of particular regions, type of network, or type of device. This scenario is illustrated in Figure 2-4.

In this configuration, the central SNM Console would require a Sender daemon but not a Receiver. Each peripheral SNM Console machine would have a Receiver installed but would not require a Sender daemon.

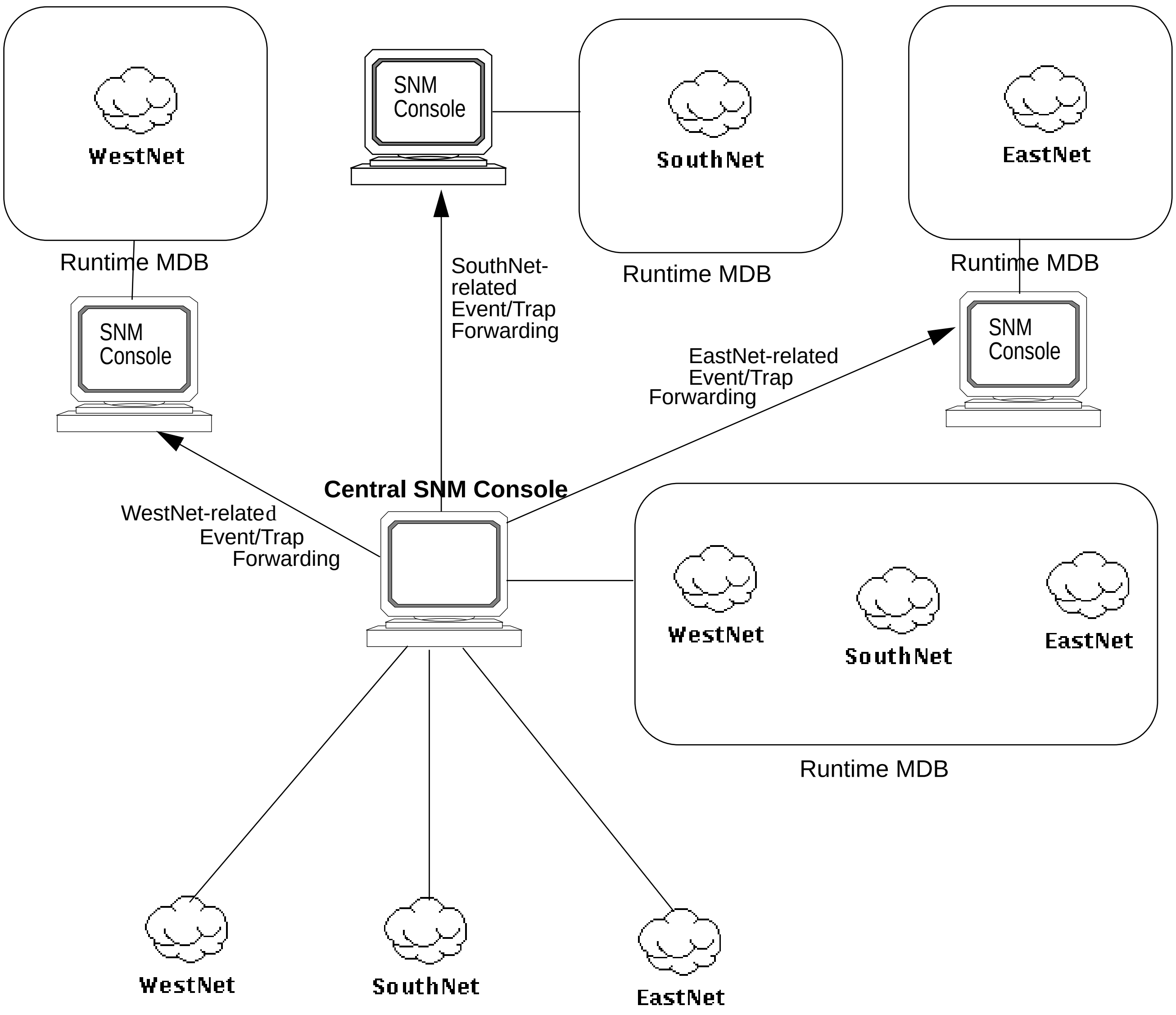


Figure 2-4 Forwarding of Information from Center to Periphery

Cooperative Consoles Operation 3

3.1 Cooperative Consoles Start-up

The Cooperative Consoles Receiver is the user entry point for starting operation of Cooperative Consoles. Installation of the Cooperative Consoles Receiver on a management station adds the Receiver to the SNM Console's Tools Menu on that machine, as shown in Figure 3-1.

.

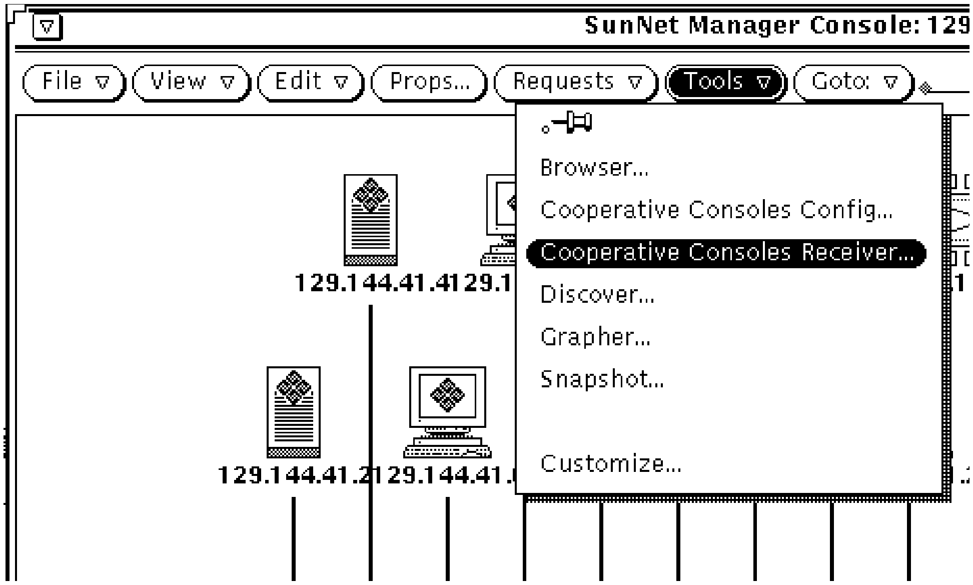
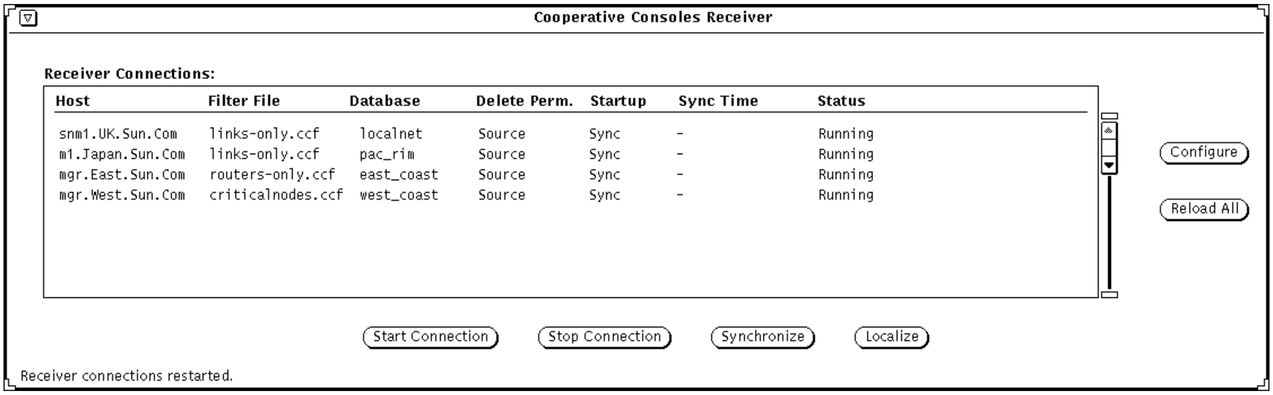


Figure 3-1 Starting the Cooperative Consoles Receiver

If you pull down the SNM Console's Tools menu and select the Cooperative Consoles Receiver, the Receiver window will appear, as shown in Figure 3-2. When launched, the Receiver process attempts to register with the Sender process on those SNM Console machines on the Receiver's Registration List. The Registration List also specifies the specific database and event forwarding criteria (Filter Table) to be used by the remote Sender processes. (You build the Receiver's Registration List using the Configuration tool — see Section 4.1, "Configuring the Cooperative Consoles Receiver.") The Receiver window provides information about the status of the connection with remote sending stations.



Receiver Connections:

Host	Filter File	Database	Delete Perm.	Startup	Sync Time	Status
snm1.UK.Sun.Com	links-only.ccf	localnet	Source	Sync	-	Running
m1.Japan.Sun.Com	links-only.ccf	pac_rim	Source	Sync	-	Running
mgr.East.Sun.Com	routers-only.ccf	east_coast	Source	Sync	-	Running
mgr.West.Sun.Com	criticalnodes.ccf	west_coast	Source	Sync	-	Running

Receiver connections restarted.

Figure 3-2 Receiver Window

The six buttons on the Receiver window have the following uses:

- **Start Connection** — This button enables you to reregister with selected sending stations. If you click SELECT a target host in the Receiver window to highlight it and then click SELECT on the Start Connection button, the Receiver process attempts to register with the Sender process on the remote management machine that you have selected.
- **Stop Connection** — This is button allows you to shutdown forwarding of information from selected hosts. for forwarding to the local SNM Console. If you click SELECT on one of the hosts listed in the Receiver window to highlight it and then click SELECT on the Stop Connection button, the Receiver process unregisters with the Sender on that remote host.
- **Synchronize** — This button allows you to do a manual synchronization of the local runtime database with the selected sending station. If you click SELECT a target host in the Receiver window to highlight it and then click

SELECT on the Synchronize button, all database information from the selected connection will be re-initialized in a two step process: First, all existing elements in the database that match the selected connection on their `Created by cc` field (<*hostname*>:<*filter-table*>:<*database-name*>) will be deleted. (The `Created by cc` field is included in the properties sheet for that element; see Figure 3-4.) Second, the Receiver will send a synchronization request to the target sending station to send all topology information that passes the pertinent filters.

- **Localize** — If you select one of the connections in the Receiver window and then click on the Localize button, every element in the local database that has a `Created by cc` field that matches the selected connection in its Created by cc field (<*hostname*>:<*filter-table*>:<*database-name*>) will have its `Created by cc` field blanked out, as if that element had been created by the local console rather than via the CC Receiver. This will prevent the Receiver from deleting these elements when a synchronization is executed.
- **Configure** — If you click SELECT on the Configure button, this launches the Configuration Tool. For information on how to use the CC Configuration Tool, refer to Chapter 4, “Using the Configuration Tool.”
- **Reload All** — If you click SELECT on the Reload All button, the Receiver unregisters with all remote Senders, rereads the Receiver’s Registration List, and then reregisters with the remote Senders on the Registration List. If you change the Receiver’s Registration List while the Receiver is running — for example, by adding new stations — you will need to use the Reload All button to force the Receiver to use the updated Registration List.

You shut down the Receiver by pressing MENU over the control bar at the top of the window and selecting Quit from the control bar menu.

3.2 Receiver Operation

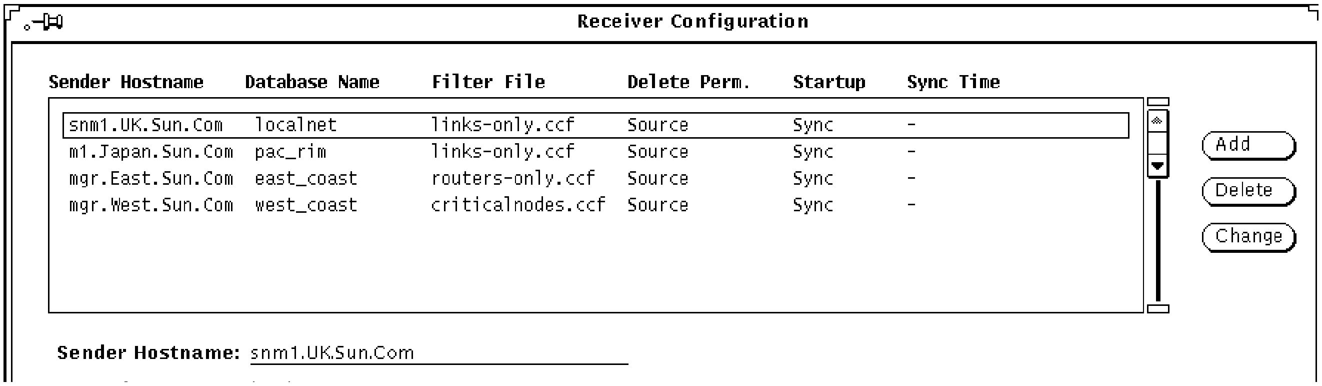
When the user first starts the Receiver, the Receiver process attempts to register with the Sender process on each machine specified on a list of target management stations. When registering with a Sender on a remote station, the Receiver can pass a user database name (the SNM_NAME value) that the Sender process should use to select database information when forwarding database traps from that machine. Since multiple user databases can exist on the SunNet Manager machine, a user database name is needed to specify which database the Sender process is to access. The Sender will access the database file named

`db.`<*database-name*>

where <*database-name*> is passed by the Receiver at the time of registering with the Sender.

The Receiver process also passes to each Sender the name of the filter file to use for selecting event and topology information for forwarding to the receiving station.

An example of a Receiver's list of target stations for registration is shown in Figure 3-3.



Sender Hostname	Database Name	Filter File	Delete Perm.	Startup	Sync Time
snm1.UK.Sun.Com	localnet	links-only.ccf	Source	Sync	-
m1.Japan.Sun.Com	pac_rim	links-only.ccf	Source	Sync	-
mgr.East.Sun.Com	east_coast	routers-only.ccf	Source	Sync	-
mgr.West.Sun.Com	west_coast	criticalnodes.ccf	Source	Sync	-

Sender Hostname: snm1.UK.Sun.Com

Figure 3-3 Sample Registration List

With the list shown in Figure 3-3, the Receiver process will attempt to register with the four machines listed in the "Sender Hostname" column when Receiver is first launched from the Console Tools Menu. As the Receiver attempts to register with the Sender at mgr.West.Sun.Com, it will pass the SNM_NAME value west_coast to indicate that the Sender should access the database file `db.west_coast`. The Receiver will also pass the file name "criticalnodes.ccf," which contains the Filter Table that mgr.West.Sun.Com is to use for selecting information to forward.

Registration with a remote SNM Console machine will be successful only if the requesting Receiver is on the target Sender's list of stations that are authorized to receive forwarded information. (You use the Configuration Tool on the sending station to define the list of remote stations authorized to register with the local Sender process, as described in Section 3.4.1, "Access Control.")

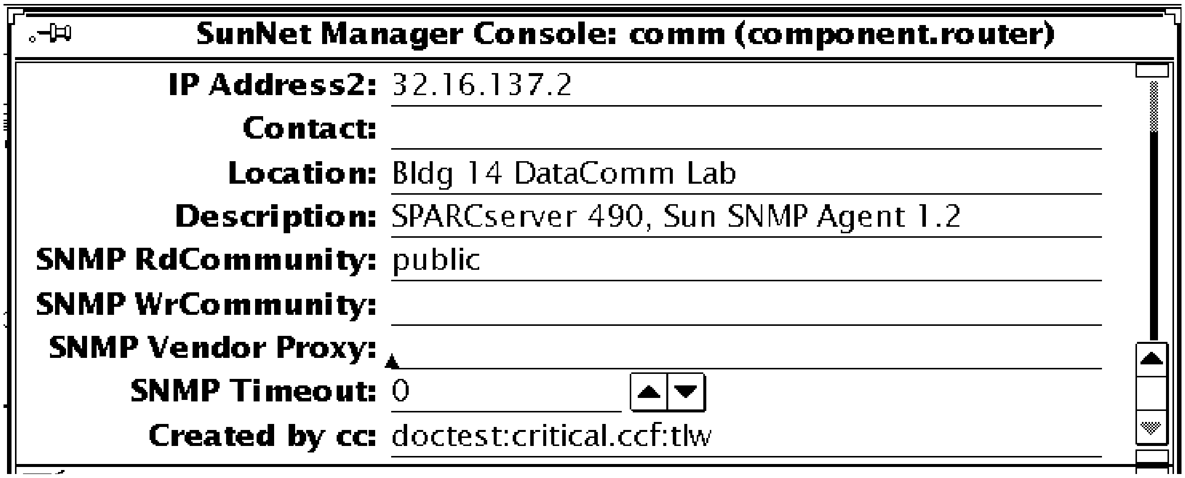
A Receiver's attempt to register with a Sender process will fail if the Receiver passes the name of a nonexistent filter file or passes a database name it is not authorized to access.

The Filter Tables include the names of Database Template files that are used by the Sender daemon to select database information to be forwarded to the Receiver process that has requested use of that Filter Table. Receivers running on different hosts may request use of different Filter Tables when registering with the same Sender daemon.

3.2.1 *Receipt of Topology Information*

Once a Receiver process successfully registers with a Sender process on a remote station, SNM database traps will be forwarded from the Sender process to the Receiver process according to user-configurable criteria. (You use the Configuration Tool on the sending station to define the Sender's processing of forwarded information.) Multiple Receivers on various hosts can register with the Sender on a given SunNet Manager Console machine and the criteria used for forwarding can be configured separately for each receiving station.

The Receiver process does not receive information about network events (such as traps or SNM events) that are to be forwarded from remote Sender processes. Rather, this event-related information is forwarded by the Sender process directly to the Event Dispatcher on the receiving station after the Receiver's registration with the Sender has been completed successfully.



SunNet Manager Console: comm (component.router)

IP Address2: 32.16.137.2

Contact:

Location: Bldg 14 DataComm Lab

Description: SPARCserver 490, Sun SNMP Agent 1.2

SNMP RdCommunity: public

SNMP WrCommunity:

SNMP Vendor Proxy:

SNMP Timeout: 0 ▲▼

Created by cc: doctest:critical.ccf:tlw

Figure 3-4 `Created by cc` Field in an Element's Properties Sheet

As topology traps are received by the Receiver process, the Receiver uses the database Application Programming Interface (API) of the local SNM Console to update the local database to reflect the topology information received from sending stations. The local Receiver is able to track the source of the elements that it adds to the local database due to the `Created by cc` field. This field is blank for all elements created by the local SNM Console.

Whenever the Receiver creates elements in the local database, the `Created by cc` field is filled in, as shown in Figure 3-4, to indicate the connection that was the source of that topology information. The `Created by cc` field matches the connection on hostname, filter table, and database name.

User-configurable filters are used at the sending station to determine what topology information should be forwarded. Forwarded information may include view memberships, agents and proxy system names on the element's "properties sheet," glyph color, screen position (X,Y coordinates), and attributes (as specified in the element's schema).

When the Receiver process receives a topology trap from a sending station, it reads the local SNM database to determine if this element already exists. If the element does not exist, then it is added to the local database. If an element already exists in the local database, the Receiver process determines whether the forwarded features of the element match those already attributed to the element. If the element's characteristics in the local database already match the features reported in the trap, the trap is ignored. If the element's characteristics in the database differ from the forwarded characteristics, the element information in the local database is changed to match the information forwarded from the sending station.

Note – If an element forwarded from the sending station already exists in the local runtime database and the existing record does not match the information passed from the remote sending station, the Receiver will *overwrite* the existing record with the information forwarded from the Sender process on the remote station. Only the view membership information in the existing record will be retained. If you want to protect local database records from being overwritten, you should configure the Sender daemons to not forward database traps to the local Receiver for those elements.

3.2.1.1 Synchronization of Shared Topology Information

The filters on the sending station define the area of information shared with remote receiving stations. When the databases exactly match in this targeted information, they are said to be "synchronized." If a Receiver should be down for a period of time, the database information on the sending machine may change during that interval. "Synchronization" is the process of updating the topology information on the receiving station so that the two stations have exactly the same picture of the segment of network information that is to be shared between them.

Synchronization can be executed manually by clicking on the Synchronize button (see Figure 3-2) or by configuring the Receiver to request synchronization automatically at start-up or at scheduled times. The Receiver initiates synchronization by sending a synchronization request to a selected Sender. In response to a synchronization request from an authorized Receiver, the Sender reads through its local database and forwards to the Receiver all topology data that passes the specified filters.

Whenever the Receiver adds elements to the local database to reflect information passed by remote Senders, it indicates in the element's `Created by cc` field (as shown in Figure 3-4) the hostname, filter table name, and database name that was the source for that element.

Whenever a synchronization is initiated for a selected sending station, the Receiver removes all the existing records in the database with a `Created by cc` field that match the target host, filter, and database names. Replacement records are then added to the local database as new information is received from the sending station in response to the synchronization request. For more information about database synchronization, see Section 3.5, "Database Synchronization."

3.2.1.2 Holding Area Views

By default, a "holding area" view is created at the receiving console for each connection to a remote sending station. Holding area views created by the Receiver are indicated in the Home view by the Cooperative Consoles icon, as shown in Figure 3-5. The name of a "holding area" view has the following form:

snm:<*host-name*>:<*database-name*>

where <*host-name*> is the name of the sending station and <*database-name*> is the database name (SNM_NAME value) being used by the Sender daemon at that station.

In Figure 3-5 Cooperative Consoles holding area views for sending machines named “gatoloco” and “montecarlo” are present in the Home view.

When information about an element’s view memberships is passed from a sending station, the Receiver adds the element to the specified views, if they already exist. The “holding area” is used to store elements whose view location is unknown to the Receiver. This situation can occur if elements are created at the sending station between synchronizations. If the element is a member of <*viewname*> where <*viewname*> is not a view that already exists on the receiving station, the Receiver will place the new element in the “holding area” view that corresponds to that sending station.

Topology information that is passed to a receiving station during synchronization will never go into holding area views, however. During synchronization viewname information is always passed before element information, so the views required for the elements will already exist in the database at the receiving station. Thus, the holding area views will typically be empty during normal operation.

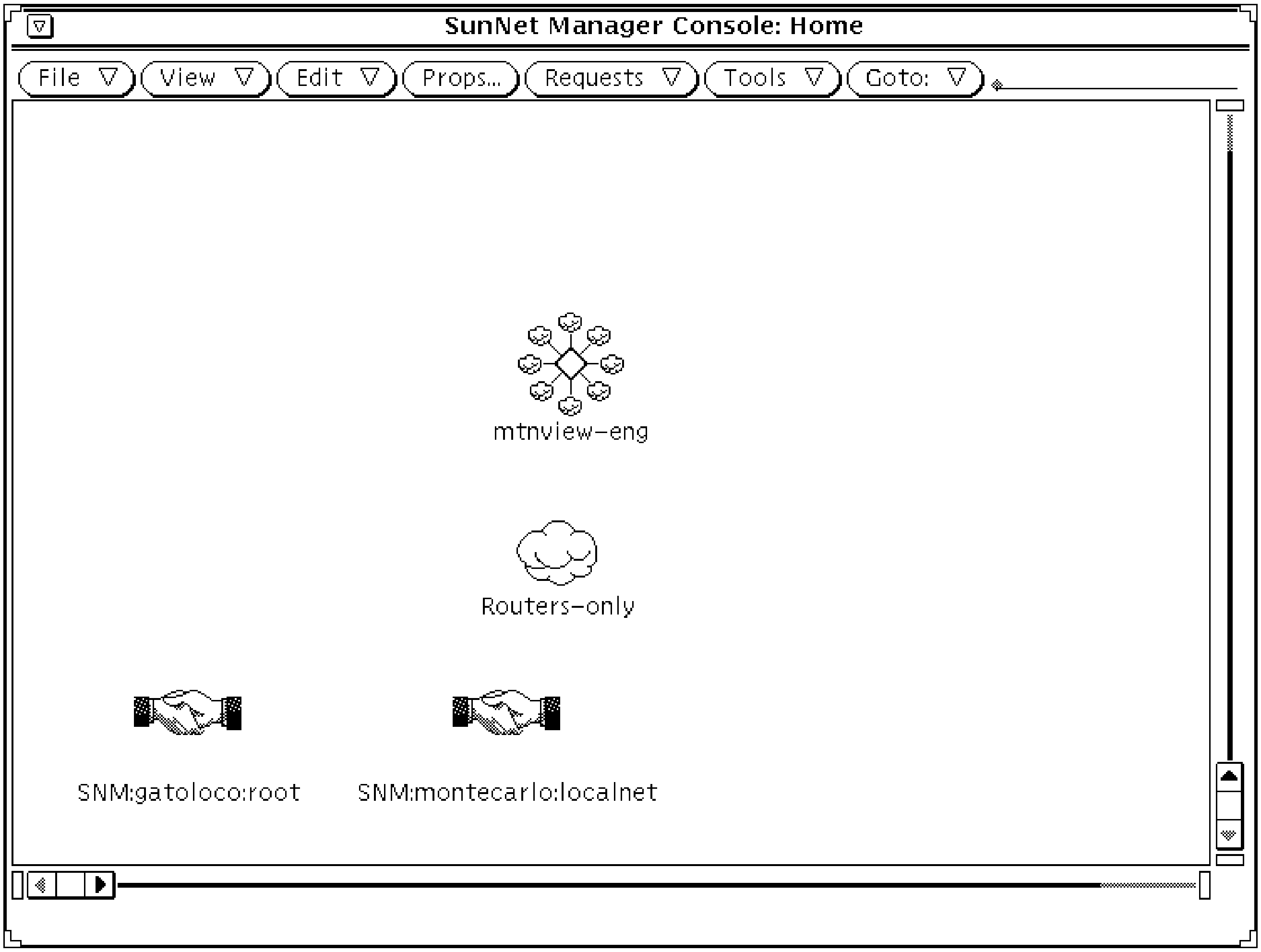
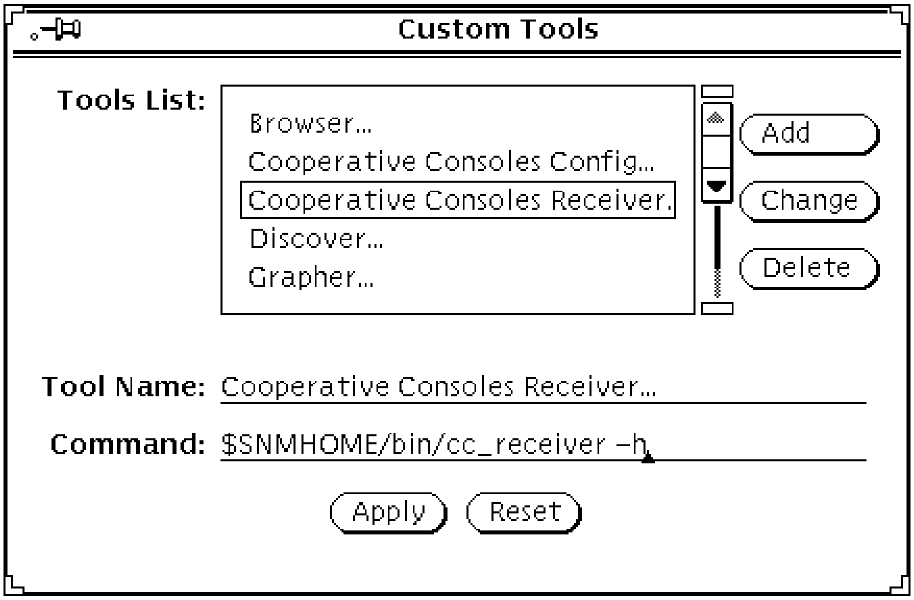


Figure 3-5 Holding Area Views in the Console Home View

3.2.1.3 *Using the Receiver* `-h` *Option*

If you want the Receiver to leave elements (components, views, etc.) that it adds to the local database in the “holding area” view, the Receiver’s `-h` runtime option allows you to implement this. If you specify the `-h` option, any views and elements added to the local runtime database by the Receiver will be under the “holding area” view. This approach is useful if the local Console is receiving forwarded topology information from a number of sending stations. The topology information forwarded from each machine will be grouped under its respective holding area view in the Console's Home view.

To implement the `-h` option, you will need to select Customize from the SNM Console Tools menu, which will invoke the Custom Tools window, as shown in Figure 3-6. Select the Cooperative Consoles Receiver and type in the `-h` runtime option, as shown below. Click SELECT on the Change button and then click SELECT on the Apply button to save the change.



Tool Name: Cooperative Consoles Receiver...

Command: $SNMHOME/bin/cc_receiver -h

Figure 3-6 Adding the Receiver `-h` Option

Note – Using the `-h` option will not prevent the Receiver from overwriting element records if information for that same element is forwarded from a sending station and it does not match the existing record. The `-h` option merely provides a convenient way of grouping forwarded information by sending machine. If you want to prevent overwriting of certain element records in the local SNM database, you will need to configure the Sender daemons to not forward topology information about those elements.

3.3 The Role of the SunNet Manager Event Dispatcher

A building block used by Cooperative Consoles is the facility for forwarding of events and traps provided by the SunNet Manager Event Dispatcher (`na.event`). The Event Dispatcher permits network management applications to register with it to receive traps, SNM events, and SunNet Manager database traps. Each instance of the SunNet Manager Console running on that machine is itself a network management application that registers with the Event Dispatcher to receive all traps and events. The Event Dispatcher spawns a child process for each network management application that registers with it.

When the Sender daemon on a given host is first launched, it registers with the Event Dispatcher on that machine in order to receive all traps, SunNet Manager events, and SunNet Manager database traps. The Event Dispatcher forwards this information to the Sender daemon on the local machine for filtering and forwarding to receiving stations.

On each receiving station, the local Event Dispatcher receives event-related traps from remote Sender processes. The Event Dispatcher then forwards this information to the SNM Console on the local machine. The Event Dispatcher on the receiving station does not receive SNM database traps forwarded from remote Sender daemons — topology information is sent to the Receiver process on the receiving station.

3.4 Sender Daemon Operation

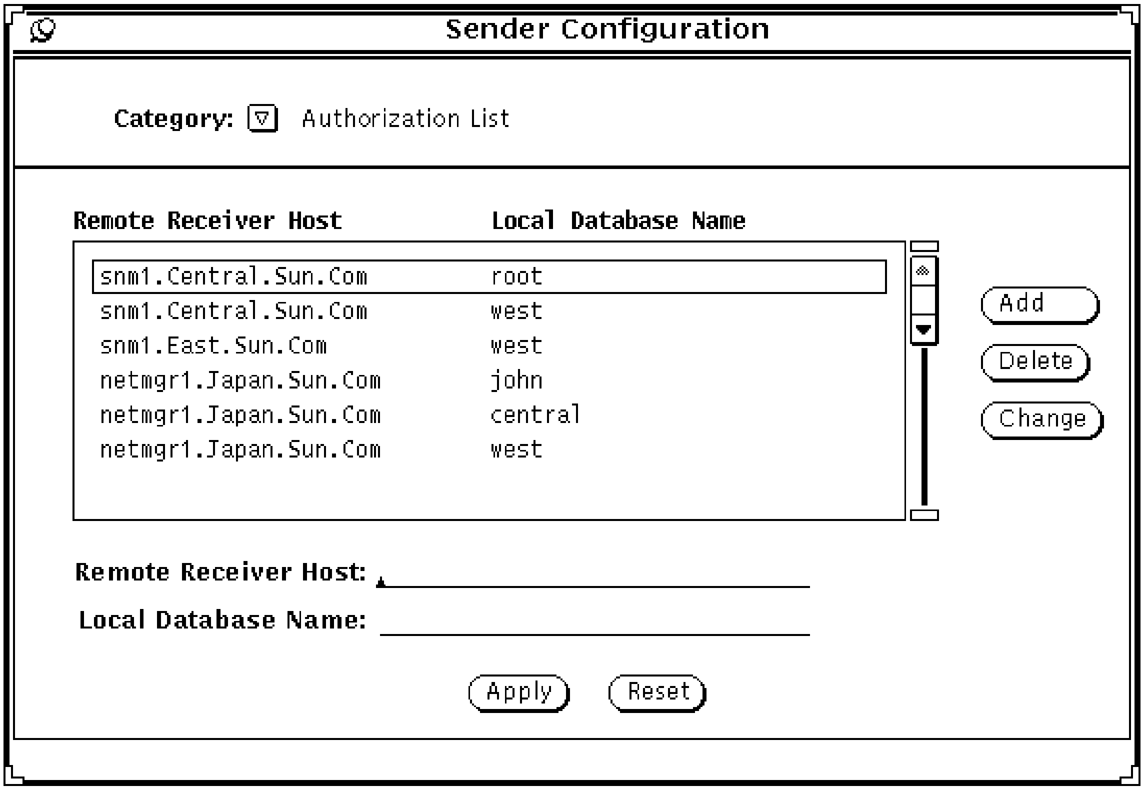
The Sender process on a SunNet Manager Console machine is launched in response to the first registration request from a remote Receiver process. When the Sender process is first created, it registers with the local SunNet Manager Event Dispatcher to receive all traps, SNM events, and SNM database (topology) traps.

A separate child Sender process is spawned for each receiving station that registers with the Sender. A Receiver process unregisters with the Sender if the user selects Quit from the Receiver window control menu on the receiving station. The Sender process unregisters with the Event Dispatcher and exits if every Receiver that had registered with it has unregistered.

3.4.1 Access Control

Access of remote receiving stations to local event and topology information is controlled by the Sender daemon. A remote Receiver process can successfully register with the Sender, and access a given local database, only if that receiving station is authorized to register with this Sender and access the specified database. To make this determination, the Sender daemon uses an authorization list that you define using the Configuration Tool on the sending station. This authorization list provides you with the ability to prevent unauthorized eavesdropping and discourage unnecessary forwarding of network event information.

The authorization list allows you to specify a list of authorized databases for each receiving station. The example in Figure 3-7 illustrates the format of an authorization list.



Remote Receiver Host	Local Database Name
snm1.Central.Sun.Com	root
snm1.Central.Sun.Com	west
snm1.East.Sun.Com	west
netmgr1.Japan.Sun.Com	john
netmgr1.Japan.Sun.Com	central
netmgr1.Japan.Sun.Com	west

Remote Receiver Host:

Local Database Name:

Figure 3-7 Sample Authorization List

In this example, the machine where this list is located is responsible for the western region of an enterprise network and this list defines access for machines in three other regions. The names listed in the Local Database Name column define SNM_NAME values. In this example, the machine netmgr1.Japan.Sun.Com has access to `db.west`, `db.central`, and `db.john`.

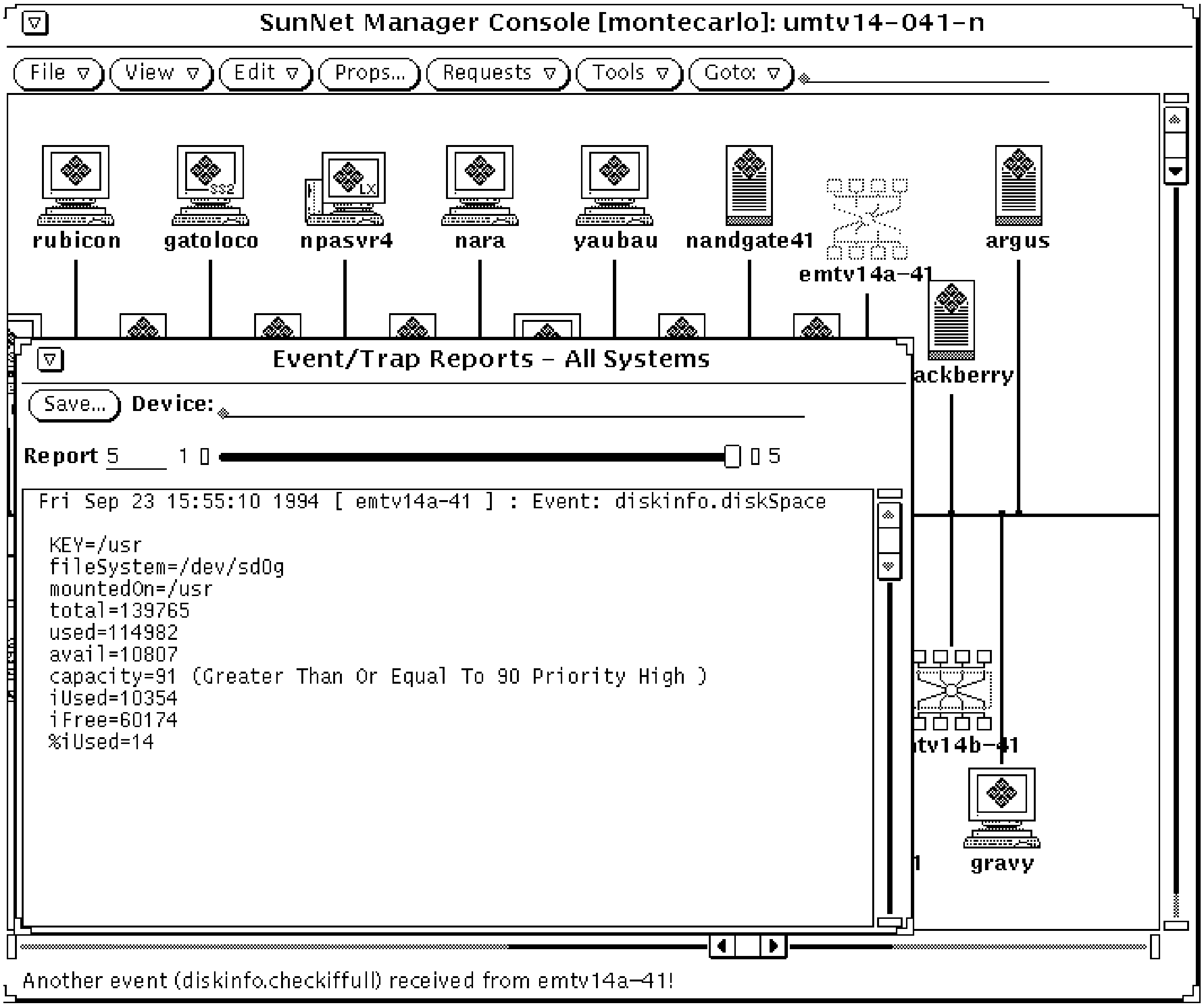
3.4.2 *Information Forwarding*

SNM agents send SNM events to the Event Dispatcher on the sending station in response to event requests launched by the SNM Console on that machine. The Event Dispatcher forwards these events to the Sender daemon.

The Sender daemon reformats all traps and SNM events into SNM traps before forwarding them to the Event Dispatcher on the receiving station. The Event Dispatcher passes the traps to the local SNM Console. An SNM event must be converted into a trap before being sent to a remote SNM Console because an SNM Console will ignore an SNM event that it cannot match to one of its own event requests.

3.4.2.1 *Event-Forwarding Example*

The example in Figure 3-8 shows an SNM event as viewed in the Event/Trap Reports window on the SNM Console running on the machine “montecarlo.” As indicated in the Console footer, the event has been generated by the SNM `diskinfo` agent on the router emtv14a-41 in response to an event request “diskinfo.checkiffull” launched from this Console machine. The condition that defined the occurrence of this event was any file system on emtv14a-41 having more than 90% of its capacity used. The glyph representing emtv14a-41 has dimmed to indicate an alarm has occurred.



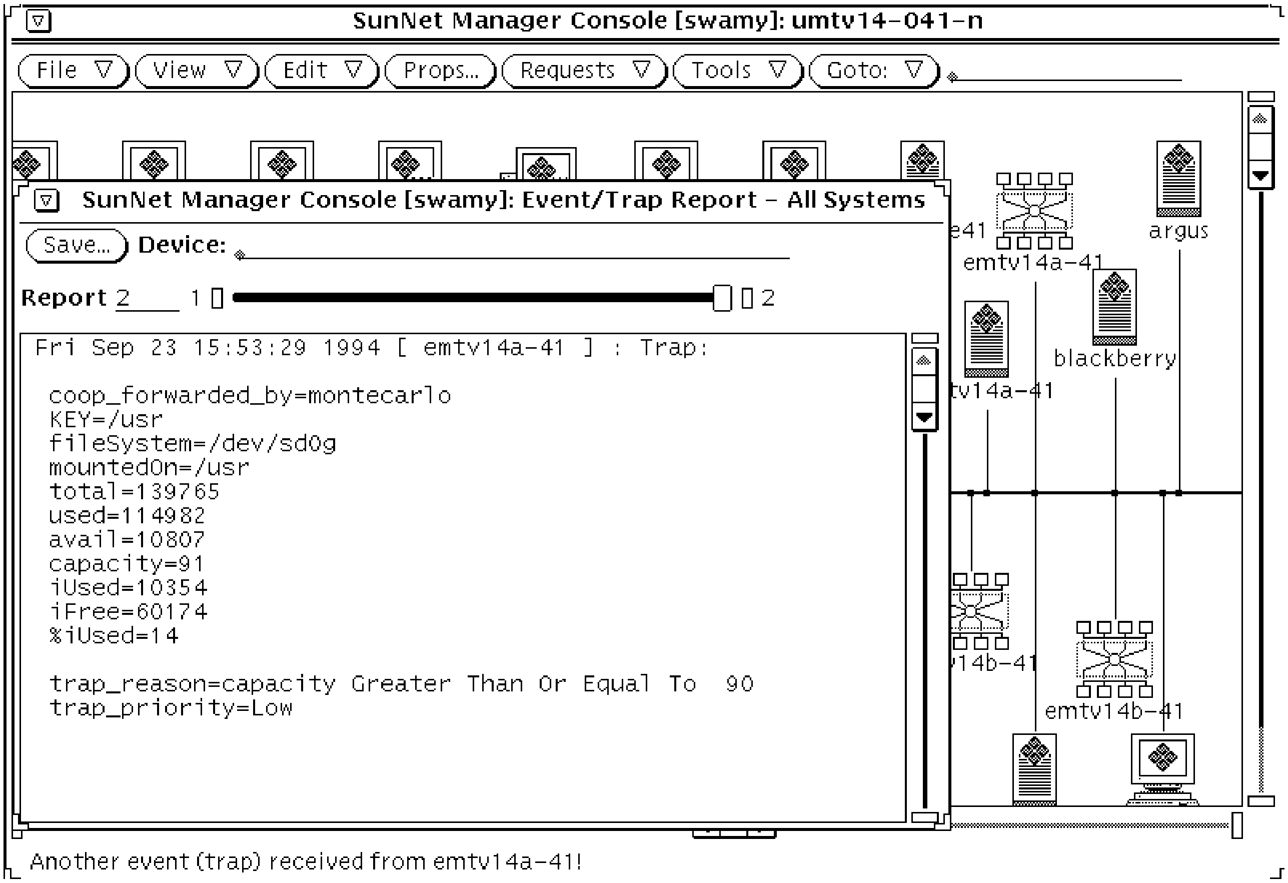
```
Fri Sep 23 15:55:10 1994 [ emtv14a-41 ] : Event: diskinfo.diskSpace

 KEY=/usr
 fileSystem=/dev/sd0g
 mountedOn=/usr
 total=139765
 used=114982
 avail=10807
 capacity=91 (Greater Than Or Equal To 90 Priority High )
 iUsed=10354
 iFree=60174
 %iUsed=14
```

Another event (diskinfo.checkiffull) received from emtv14a-41!

Figure 3-8 Example of Event as Viewed on Sending Station

The Sender daemon on montecarlo will reformat this event as a trap and then forward it to the appropriate receiving stations. Figure 3-9 shows this trap as viewed in the Event/Trap Reports window on the receiving station.

The line “coop_forwarded_by=montecarlo” in the Event/Trap Reports window in Figure 3-9 indicates that this trap is an event or trap that is forwarded by the Cooperative Consoles Sender process located on the SNM Console machine named “montecarlo.” Also, notice that the priority of the event has been changed by the Sender from high to low, as indicated in the last line in the Event/Trap Report window in Figure 3-9. This action is configured by the user when defining the Filter Table that will be used by the Sender for forwarding of events to this receiving station.



```
Fri Sep 23 15:53:29 1994 [ emtv14a-41 ] : Trap:

  coop_forwarded_by=montecarlo
  KEY=/usr
  fileSystem=/dev/sd0g
  mountedOn=/usr
  total=139765
  used=114982
  avail=10807
  capacity=91
  iUsed=10354
  iFree=60174
  %iUsed=14

  trap_reason=capacity Greater Than Or Equal To  90
  trap_priority=Low
```

Another event (trap) received from emtv14a-41!

Figure 3-9 Event as Viewed on Receiving Station

3.4.2.2 *Filtering of Events for Forwarding*

The Sender process uses one or more user-configurable, table-driven filter files in processing events and traps for forwarding to the Event Dispatcher on receiving stations. The filters also specify separate database template files which define the database information forwarded to the receiving station's Receiver process in response to SNM database traps.

Separate filter files can be selected for each receiving station. When the Receiver registers with a Sender process, it passes the name of the filter file that the Sender is to use for selecting information forwarded to that receiving station.

Each filter file contains a Filter Table that defines the action to be taken for a given event or trap. The action to be taken can be selected on the basis of host name of the device originating the event or trap, or by database component type name (for example, events from any device of type `component.router`). The action to be taken can also be further selected on the basis of event priority.

An event that is considered a high priority event for one management station may have a lower priority for another management station. For this reason, you can also use the filters to change the priorities of events when forwarded from a sending station to a receiving station.

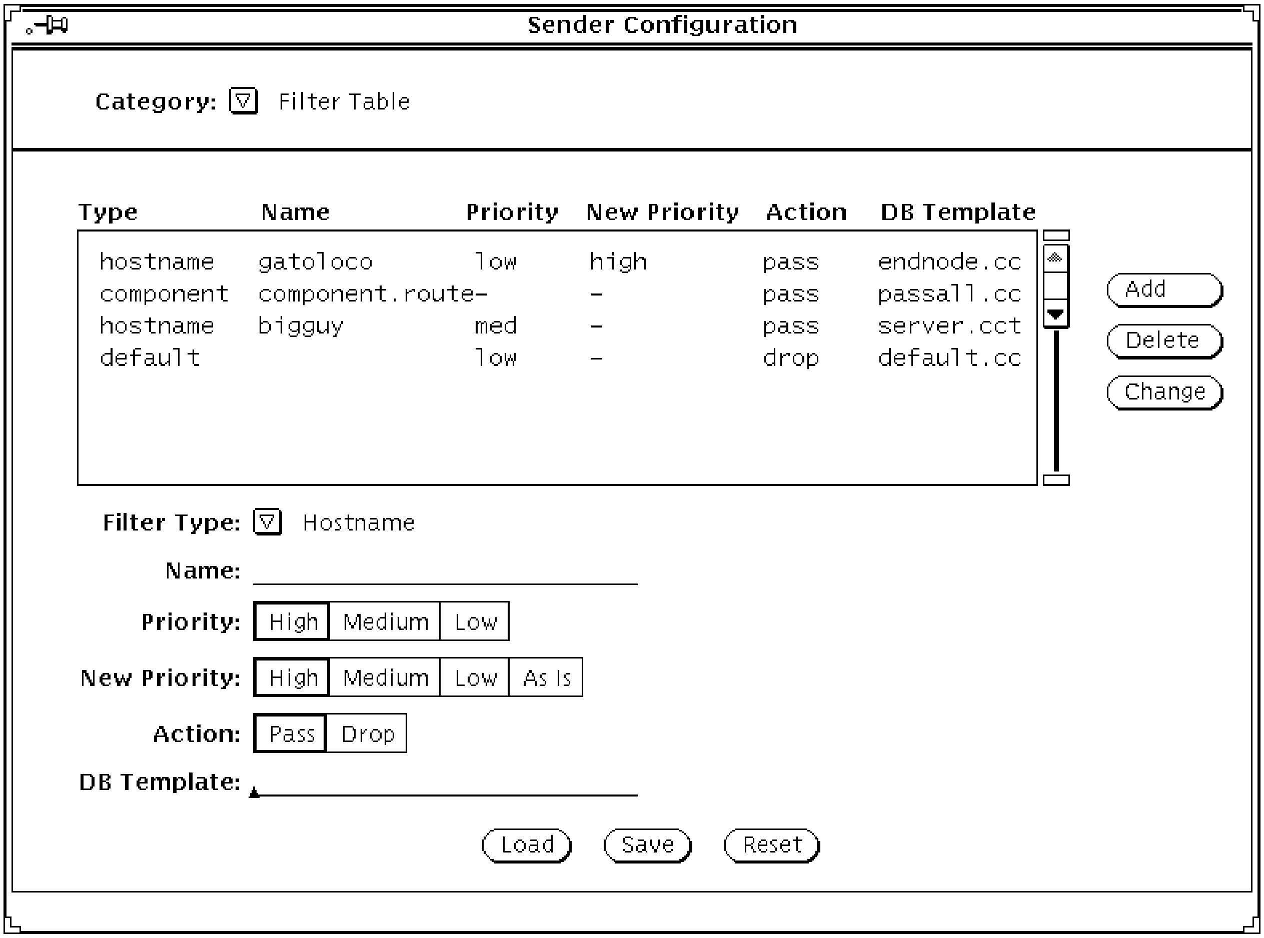
Multiple receiving stations can use the same filter file if the same criteria are to be used in processing events for forwarding, or separate filters can be defined for different receiving stations.

Each filter in a filter file has the following fields:

- **Type** — This determines the primary selection criterion for processing a trap or event. Only three values are possible here: Hostname, Component, or Default. There can be only one filter in a filter file with the type Default; this is the filter used if an event or trap does not match the Type criterion of any of the other filters. If the type is Hostname, this selection criterion is satisfied if the value of the Name field matches the host name associated with the event or trap. If the type is Component, this selection criterion is satisfied if the SNM component type in the **Name** field (for example, `component.router`) matches the component type associated with the event or trap.

- **Name** — If the Type is Hostname, the **Name** field value must be either an IP address or the host name of a glyph in the SNM database. If the Type is Component, the Name field value must be the name of a component type (for example, `component.router`), view type (for example, `view.subnet`), or bus type (for example, `bus.rs232`).
- **Priority** — This field specifies the *lowest* priority event or trap that this filter will process. For example, if “low” is specified, then events of *any* priority will satisfy this criterion. This filter will be selected if the event or trap satisfies both the Name field criterion and the Priority field criterion.
- **New Priority** — If a value is specified here, this will be the priority of the trap when forwarded.
- **Action** — This field specifies whether an event or trap selected by this filter should be either forwarded or ignored.
- **DB Template** — This field specifies the file name of the database template that should be used in selecting information from the database when forwarding SNM database traps.

An example of a filter table is shown in Figure 3-10.



Category: Filter Table

Type	Name	Priority	New Priority	Action	DB Template
hostname	gatoloco	low	high	pass	endnode.cc
component	component.route-		-	pass	passall.cc
hostname	bigguy	med	-	pass	server.cct
default		low	-	drop	default.cc

Filter Type: Hostname

Name:

Priority: High | Medium | Low

New Priority: High | Medium | Low | As Is

Action: Pass | Drop

DB Template:

Figure 3-10 Sample Filter Table

Each row in this table defines a distinct filter. The order of the filters defines which filter selection criteria are examined first to determine a match. The first filter in the table whose Name and Priority selection criteria match the event or trap will be used in processing that event or trap. The Default filter is used if none of the other filters is selected.

In the Figure 3-10 example, the Sender process will first check for host name match when it receives a trap, SNM event, or SNM database trap. If an event or trap for gatoloco is received from the Event Dispatcher, it is forwarded to the receiving station's Event Dispatcher as an SNM trap with a high priority. If an SNM database trap for gatoloco is received, the file `endnode.cct` is used to determine the type of information that the Sender retrieves from the local database for forwarding to the Receiver process on the receiving station.

3.4.2.3 *Forwarding of Topology Information*

The local SNM Console can generate six types of SNM database traps:

- **Add** — Generated when a new element is added to the database via the database Application Programmatic Interface (API).
- **Background** — Generated when a background image is added to a view.
- **Create** — Generated when a new element is created via the SNM Console.
- **Change** — Generated when attributes of the element (such as the agents list or the screen coordinates) are changed.
- **Delete** — Generated when an element is deleted from the database.
- **Load** — Generated when a new management database is loaded.

When the Sender daemon receives SNM database traps from the local Event Dispatcher, it uses the filter file specified for the target receiving station to determine which database template file to use in processing the database trap. The DB Template field in the filter contains the template file name.

The name of the element is the minimum information that is passed when a database trap is forwarded to the Receiver process on the receiving station. The database template file allows you to specify additional topology information that should be forwarded.

For database traps of type **Delete,** the only information that can be passed to the Receiver process, for such traps, is the element's name. For traps of type **Delete**, you can indicate whether such traps should be dropped, but no other information needs to be selected for that trap type. The DB template file allows you to specify the types of database information to forward for each of the other types of SNM database trap.

Because the filter file allows you to specify different DB Template files in each filter, you can use the filter selection criteria to specify different types of topology information to forward for different devices (by host name or element type), if you define different DB template files. (You use the Configuration Tool to define both event/trap filters and DB templates.)

A DB Template file has the following format:

Table 3-1 Database Template Format

Trap Type	**Keywords (one or more can be specified)**
Add	membership, color, agents, attributes, connections, drop
Create	membership, color, agents, attributes, connections, drop
Change	membership, color, agents, attributes, connections, drop
Delete	drop
Load	membership, color, agents, attributes, connections, drop
Background	drop

The keywords that are used will determine the forwarded content for each of these four trap types. The keywords have the following interpretation:

- **Membership** — If specified, the view name and coordinates will be passed for each view the element belongs to.
- **Color** — If specified, the element's RGB values are passed.
- **Agents** — If specified, the name of each agent selected on the element's properties sheet is passed, and the proxy system name is also passed for each specified proxy agent.
- **Attributes** — If specified, the entries for each attribute specified in the schema file for that element (e.g., IP address or contact name) are passed.
- **Connections** — If specified, information about the simple connections to the element will be forwarded.
- **Drop** — If specified, traps of this type will not be forwarded.

To allow access of registered Receivers to a multiplicity of SNM databases on the machine where the Sender is located, the Sender daemon runs as root.

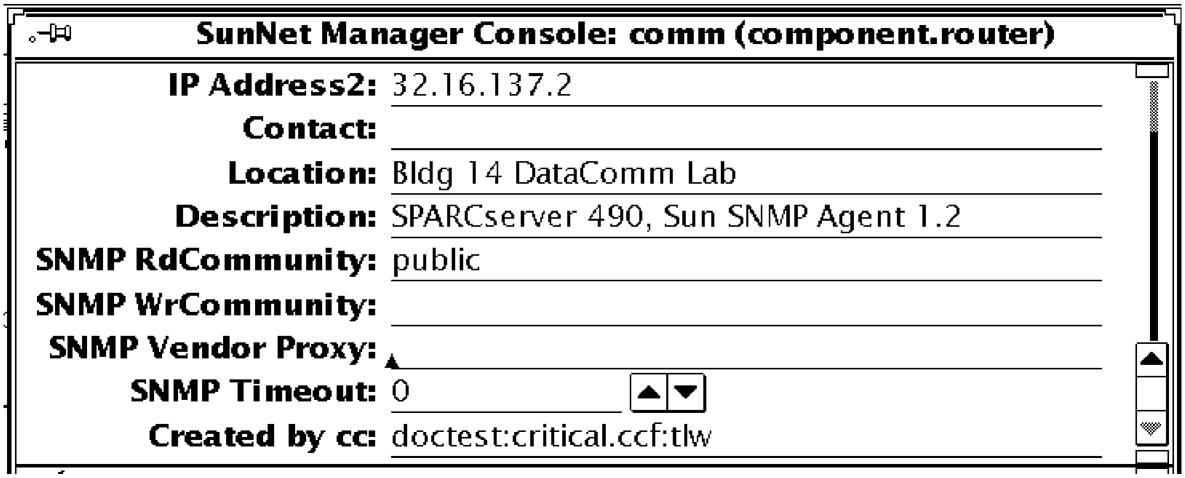
3.5 Database Synchronization

The runtime database on the sending and receiving station are said to be "synchronized" when they are in agreement on the area of network information that they are intended to share. For example, if Consoles on machines RegionWest and RegionEast are both intended to contain information about routers on Net_A, then the RegionWest and RegionEast consoles are synchronized when they have the same information about routers in Net_A. Synchronization is therefore always in regard to some defined set of data which two or more runtime databases are intended to share. This information may include everything in their respective runtime databases or only some of that information. The intended area of shared data is determined by the filters on the pertinent sending stations.

Synchronization is an action that is initiated by the Receiver. When synchronization is initiated, the Receiver sends a synchronization request to the selected sending station on its Registration List, that is, the list of target stations from which the Receiver requests event and topology information.

When a Sender daemon receives a synchronization request from a receiving station on its list of authorized Receivers, the Sender reads through the database and sends all topology information that passes the filters that are indicated for the target receiving station. As this information is received by the Receiver that has requested the updated information, the Receiver adds it to the local runtime database.

Whenever the Receiver adds elements to the local database to reflect information passed by remote Senders, it indicates in the element's `Created by cc` field (as shown in Figure 3-11) the hostname, filter table name, and database name that was the source for that element.



SunNet Manager Console: comm (component.router)

IP Address2:	32.16.137.2
Contact:	
Location:	Bldg 14 DataComm Lab
Description:	SPARCserver 490, Sun SNMP Agent 1.2
SNMP RdCommunity:	public
SNMP WrCommunity:	
SNMP Vendor Proxy:	
SNMP Timeout:	0
Created by cc:	doctest:critical.ccf:tlw

Figure 3-11 `Created by cc` Field in an Element's Properties Sheet

Whenever a synchronization is repeated for a target sending station, already existing elements with a `Created by cc` attribute that matches the target hostname, filter, and database name are deleted and replaced with the updated information received from the sending station. In this way, the Receiver insures that the local database reflects the latest information from the remote sending station. Since not all of the information in the local database may be shared information, the `Created by cc` attribute enables the Receiver to identify which portions of the local runtime database are shared with the remote sending stations.

3.5.1 Ways to Initiate Synchronization

There are three ways in which a Receiver can initiate synchronization with its target sending stations:

- **Manually** — Select the target station on the Receiver's Registration List and then click SELECT on the Synchronize button.
- **Automatically at Receiver start-up** — The Configuration Tool can be used to configure a local Receiver to request synchronization with a target sending station on any occasion when the Receiver starts up. If the runtime database on the sending station has changed since the Receiver was last

running, the receiving station will automatically have its database updated at start-up to reflect the correct picture of the area of information shared between the sending and receiving station.

- **Automatically at scheduled intervals** — You can use the Configuration Tool to schedule synchronization with a target sending station to occur automatically at a specified time of day on either a daily basis or weekly on a specified day of the week.

Refer to Chapter 4, “Using the Configuration Tool,” for information about configuring the CC Receiver for automatic synchronization.

3.5.2 Multiple Sources of Information

There are some scenarios where you may have derived an element in your local SNM database from a remote SNM Console on host A but not all of the attribute information for that element may have come from host A. An example is illustrated by the periphery-to-center configuration shown in Figure 3-12. In this scenario, information about the device `doctest` may be received at SNM Console `central` from either Console `snm1` or Console `snm2`.

In the situation depicted in Figure 3-12, the source of the instance doctest on `central` was `snm1`, as indicated by the `Created by cc` field (shown beneath the elements in this example). In a case such as this, the source is determined chronologically — the first station to send information about an element is considered as the source. But modifications to the properties sheet for `doctest` on `snm2` will also be reflected in the instance of `doctest` on `central`.

If the Receiver on `central` initiates a synchronization with `snm1`, the elements `jupiter` and `doctest` on central will be deleted and then replaced by updated elements that match the current information on `snm1`. If there is additional information about doctest that has snm2 as its source, then it will be necessary, with this configuration, to also do a synchronization with snm2 to update this information as well. When information about specific elements has its source in multiple connections, then it will be necessary to initiate a synchronization with all of these connections to update information about that element.

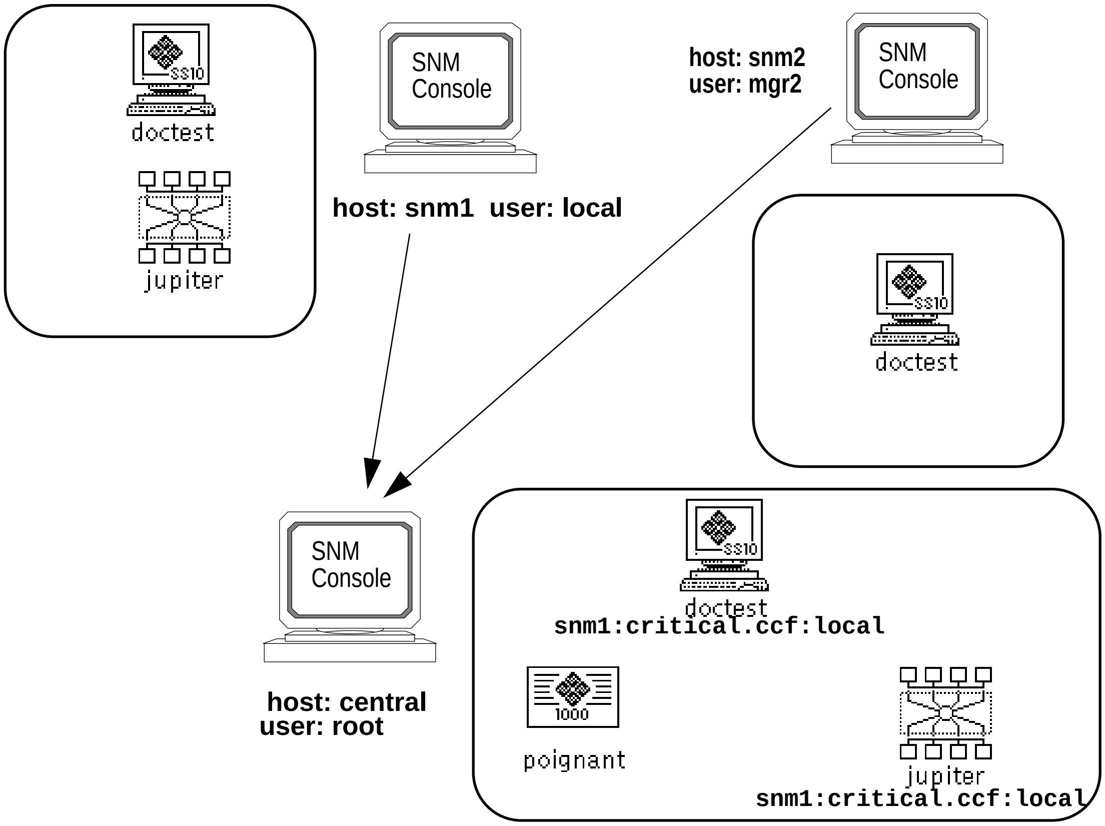


Figure 3-12 Information for Same Element from Multiple Sources

3.6 Delete Permission

The Receiver's Delete Permission option allows you to control the effect of deleting an element when multiple SNM Consoles are linked via CC. The "Created by cc" field on each element in a local database indicates to the local Receiver process the *source* of that element. If the `Created by cc` field is blank, this indicates that the local SNM Console created that element — for example, by running Discover to build a view of its local network.

On the other hand, if the element was added to the local SNM runtime database by the Receiver, the `Created by cc` field indicates the particular connection with a remote sending station that was the source of that element.

The source information in the `Created by cc` field is in the following form:

<hostname>:*<filter-table>*:*<database-name>*

In the sample configuration in Figure 3-13, the arrows indicate the direction of information flow from sending stations to receiving stations. The two elements `doctest` and `jupiter` were originally created on the station `snm1`. The `Created by cc` fields that result from this configuration are shown beneath each element.

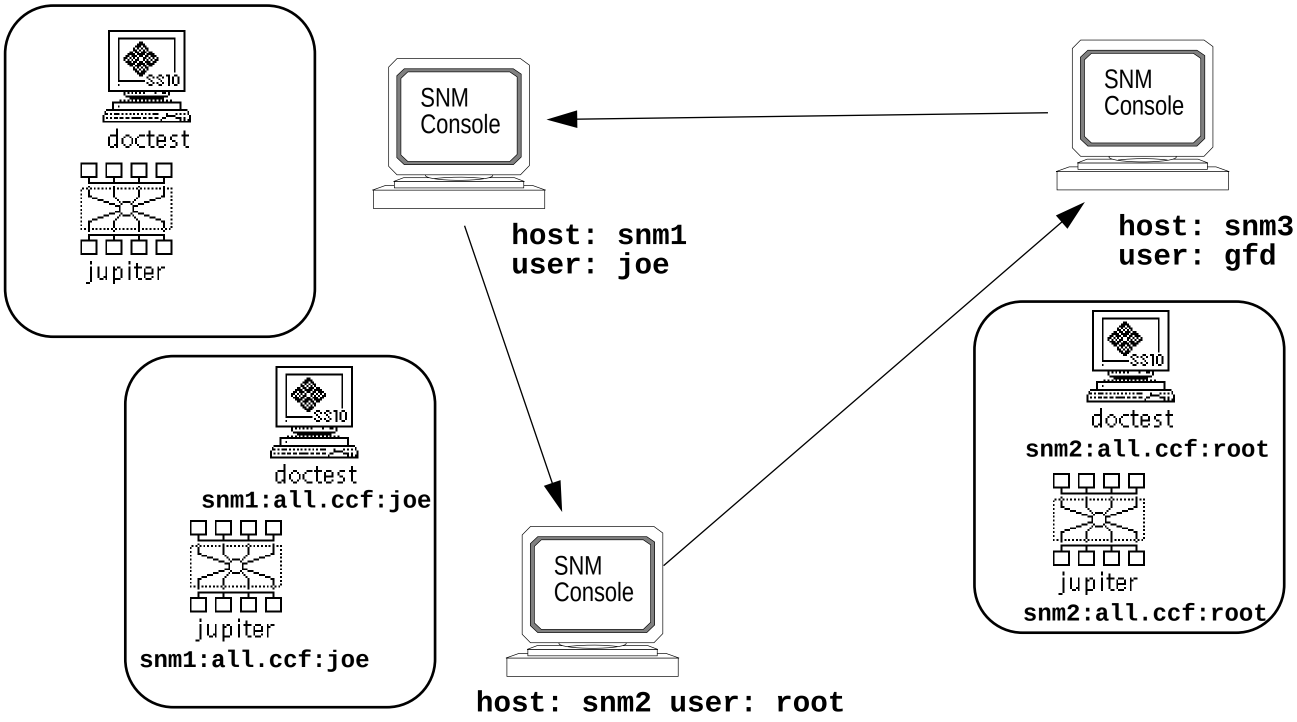


Figure 3-13 Sample Configuration Showing `Created by cc` Fields

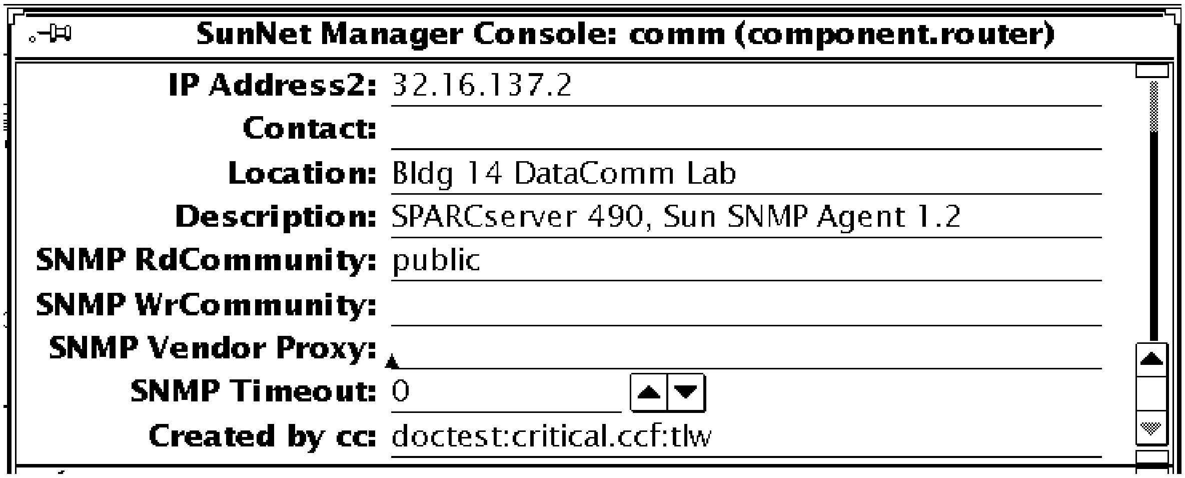
When an element is deleted on an SNM Console, the Sender will transmit a delete trap to registered Receivers if that element falls in the area of network information that is to be shared between them (as defined by the filters). If you have configured the Receiver's connection to that sending station with a Delete

Permission of “All”, then the Receiver will delete the indicated element, no matter whether that remote sending station was the source of the element or not. If the Delete Permission for that connection is set to “Source,” however, then only elements that had that connection as their source could be deleted as the result of a deletion of an element on that remote station.

In the case of the configuration in Figure 3-13, if the Receiver on `snm1` has its connection with `snm3` set to a Delete Permission of “All”, then if a user on snm2 or snm3 deletes the element, this would result in its deletion on `snm1` as well. On the other hand, if snm1’s connection to snm3 has a Delete Permission set to “Source,” then the deletion of `jupiter` on either snm2 or snm3 would not result in the deletion of `jupiter` on `snm1` since `snm1` is itself the source of that element on that console. Setting the Delete Permission to “Source” for a connection means that only an element whose `Created by cc` field matches that connection can be deleted as the result of a delete trap received from that connection.

3.7 Localizing Elements Received from Remote Stations

If SNM Console A has received an element from another SNM Console, B, then the `Created by cc` field for the instance of that element on Console B will be filled in on the element’s properties sheet to indicate that host A was the source for that information. For example, Figure 3-14 shows part of the properties sheet for a router element named “comm.” The `Created by cc` field on this properties sheet indicates that this element was received from the host `doctest`.



SunNet Manager Console: comm (component.router)

IP Address2:	32.16.137.2
Contact:	
Location:	Bldg 14 DataComm Lab
Description:	SPARCserver 490, Sun SNMP Agent 1.2
SNMP RdCommunity:	public
SNMP WrCommunity:	
SNMP Vendor Proxy:	
SNMP Timeout:	0
Created by cc:	doctest:critical.ccf:tlw

Figure 3-14 `Created by cc` Field in an Element's Properties Sheet

Elements that are created locally on B (for example, by running Discover), on the other hand, will have a blank `Created by cc` field. An element with a blank Created by cc field is said to be "local" to that Console. Elements that are not local will be deleted by the Receiver when a synchronization is initiated with the sending station that was the source for that element.

3.7.1 Localizing Selected Elements

There may be situations where you do not want an element that was received via CC from a remote station to be deleted when a synchronization is performed. You can accomplish this by blanking out the Created by cc field in the properties sheet for that element.

To do this, select Properties… from the element's icon pulldown menu to invoke the element's properties sheet. Delete the information in the `Created by cc` field and then select Apply. An element that has had its Created by cc field blanked out this way is said to be "localized." You can localize selected elements manually, as just described. But the CC Receiver also provides a facility for doing a global localization of all elements that were received from a selected connection.

3.7.2 Global Localization

Clicking on the Localize button in the Receiver window wipes out the `Created by cc` field for all elements that had the selected connection as their source. The CC Receiver then regards these elements as having the local SNM console as their source — as if they had been created locally by running Discover, for example.

You will probably want to use the Localize button only in certain special situations. If parts of the local network topology on your console have been built up by receipt of topology information from remote sending stations, there may be situations where you do not want this information to be wiped out by a new synchronization. If you localize the information received from selected stations, a new synchronization with those stations will not erase that previously acquired topology data.

An example of such a situation is illustrated in Figure 3-15. Figure 3-15 shows two SNM consoles arranged in a peer-to-peer configuration. (The peer-to-peer configuration is defined in Chapter 2, “Cooperative Consoles Configurations.”) The host `snmA` is the source of the database elements jupiter and doctest on `snmB`. If the database on `snmA` should be wiped out (for example, by starting SNM with the `-i` option), the database could be recovered if `snmA` were to synchronize with `snmB`. However, the database information that is passed from `snmB` to `snmA` will now indicate `snmB` as its source, as shown in Figure 3-16.

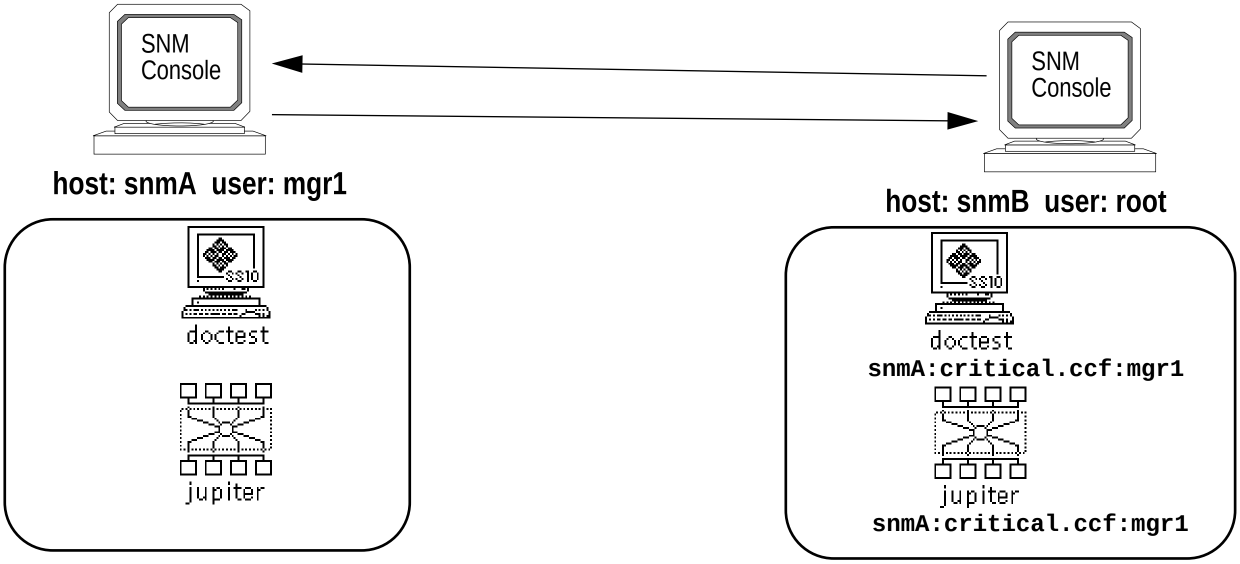


Figure 3-15 Peer-to-Peer Configuration Example Showing Created by cc Fields

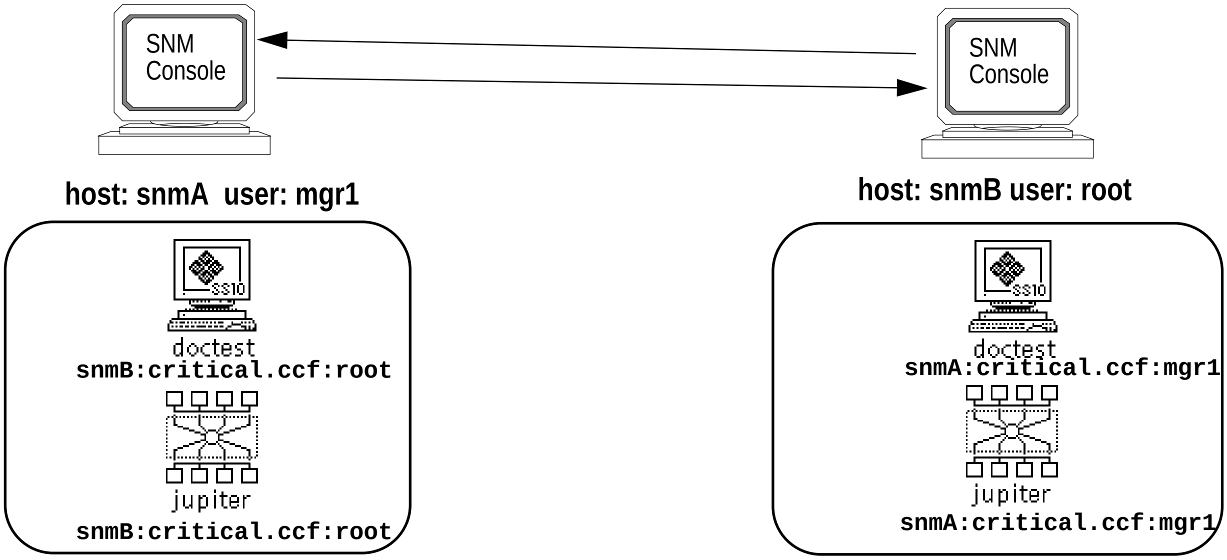


Figure 3-16 Peer-to-Peer Configuration with No Underived Elements

Note – In the situation depicted in Figure 3-16, the area of shared information is derivative on both machines — there is no underived database source for this information. In this situation it is recommended that you recreate a base or underived source for the data by localizing one of the copies of the information. Otherwise, a synchronization on either `snmA` or `snmB` for this connection will result in the data being deleted on both machines since there is no underived data source from which the information can be updated. Localizing the connection at the Receiver for `snmA` will allow you to make the local Console `snmA` be the source for the data, thus restoring the situation depicted in Figure 3-15.

3.8 Shutting Down Cooperative Consoles

The Receiver window provides the method for shutting down operation of Cooperative Consoles. If you press the MENU button over the Receiver window control panel, the control panel popup menu will appear. If you select Quit, the local Receiver process unregisters with the Sender processes at each of its target sending stations. (Alternatively, you can quit from the Receiver by pressing Meta-Q.)

When all of the Receiver processes that have registered with the Sender process on a given SunNet Manager machine unregister, that Sender process on that machine unregisters with the local Event Dispatcher and exits. Thus, all Cooperative Consoles operations — Sender and Receiver processes — will cease after Quit has been selected for the Receiver process at *each* receiving station.

3.9 Cooperative Consoles File Locations

The Cooperative Consoles executable binaries must be installed under the same path as the SunNet Manager executables ($SNMHOME). The default location for the executable binary files for Cooperative Consoles is:

- `/usr/snm` in the Solaris 1.x environment
- `/opt/SUNWconn/snm` in the Solaris 2.x environment

These files can be relocated or installed on a server and shared across a network. The file structure for the Cooperative Consoles product is shown in Table 3-2. SNMHOME here refers to the path where SunNet Manager has been installed.

Table 3-2 Cooperative Consoles File Structure

Directory	**Files**
$SNMHOME/agents	`cc_sender`
$SNMHOME/bin	`cc_config, cc_receiver`
$SNMHOME/filter	Sample filter files
$SNMHOME/help	CC on-line help files
$SNMHOME/icons	Two files added for CC
$SNMHOME/lib	Dynamic library for `libcoop`
$SNMHOME/man	CC `man` pages
$SNMHOME/struct	`cooptools.schema`

Please note the contents of the filter directory are copied to:

- `/var/adm/snm/cc_files` on Solaris 1.x environments
- `/var/opt/SUNWconn/snm/cc_files` on Solaris 2.x environment

The license file is located in:

- `/var/adm/snm/cc_files/license.dat` on Solaris 1.x environments
- `/var/opt/SUNWconn/snm/cc_files/License.dat` on Solaris 2.x environments

`libcoop` is a directory for all libraries required to operate CC.

By default, Filter Table files have names with a `.ccf` extension while DB Template files have a `.cct` extension.

Using the Configuration Tool

4

The Cooperative Consoles Configuration Tool allows you to configure the CC Sender and Receiver on the local SNM Console host. The Configuration Tool allows you to define the following:

- The list of remote stations authorized to register with the local Sender daemon and the databases they are authorized to access
- Filter files and database templates that determine the event and topology information forwarded by the Sender daemon to remote receiving stations
- The list of remote management stations the local Receiver process will attempt to register with and the database instance and filter file it will request at remote sending stations
- Schedule times for receiving stations to synchronize shared information with remote sending stations or configure a Receiver for automatic synchronization at start-up

You invoke the Configuration Tool from the SunNet Manager Console's Tools Menu, as shown in Figure 4-1.

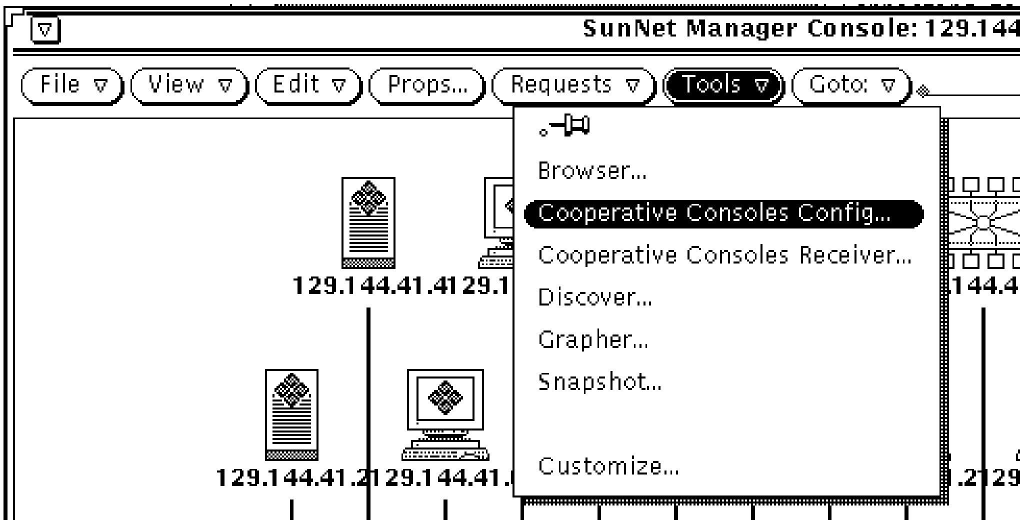
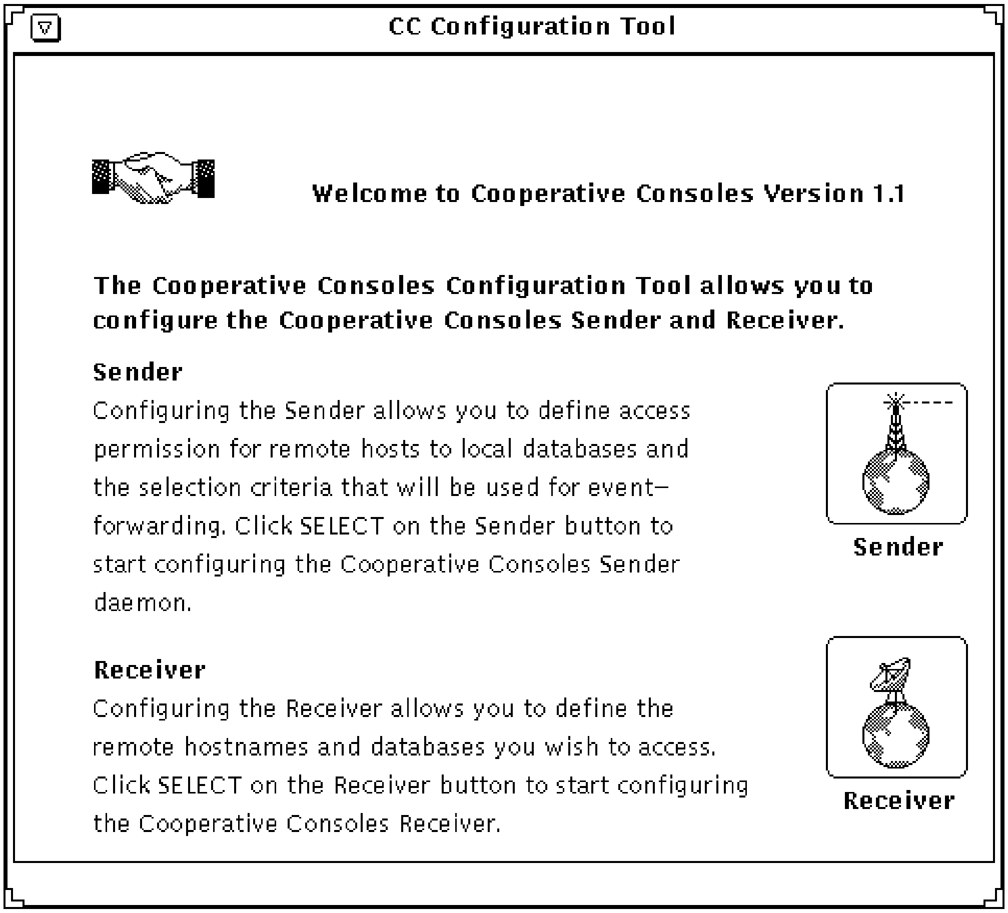


Figure 4-1 Selecting Configuration Tool from the Console Tools Menu

Invoking the Configuration Tool displays the window shown in Figure 4-2.



Welcome to Cooperative Consoles Version 1.1

The Cooperative Consoles Configuration Tool allows you to configure the Cooperative Consoles Sender and Receiver.

Sender

Configuring the Sender allows you to define access permission for remote hosts to local databases and the selection criteria that will be used for event–forwarding. Click SELECT on the Sender button to start configuring the Cooperative Consoles Sender daemon.

Receiver

Configuring the Receiver allows you to define the remote hostnames and databases you wish to access. Click SELECT on the Receiver button to start configuring the Cooperative Consoles Receiver.

Figure 4-2 CC Configuration Tool Main Menu

To configure the action of the local Sender daemon, click SELECT on the Sender button. To configure the Receiver application on the local host, click SELECT on the Receiver button.

4.1 Configuring the Cooperative Consoles Receiver

If you select the Receiver button from the CC Configuration main menu, the Receiver Configuration window appears, as shown in Figure 4-3.

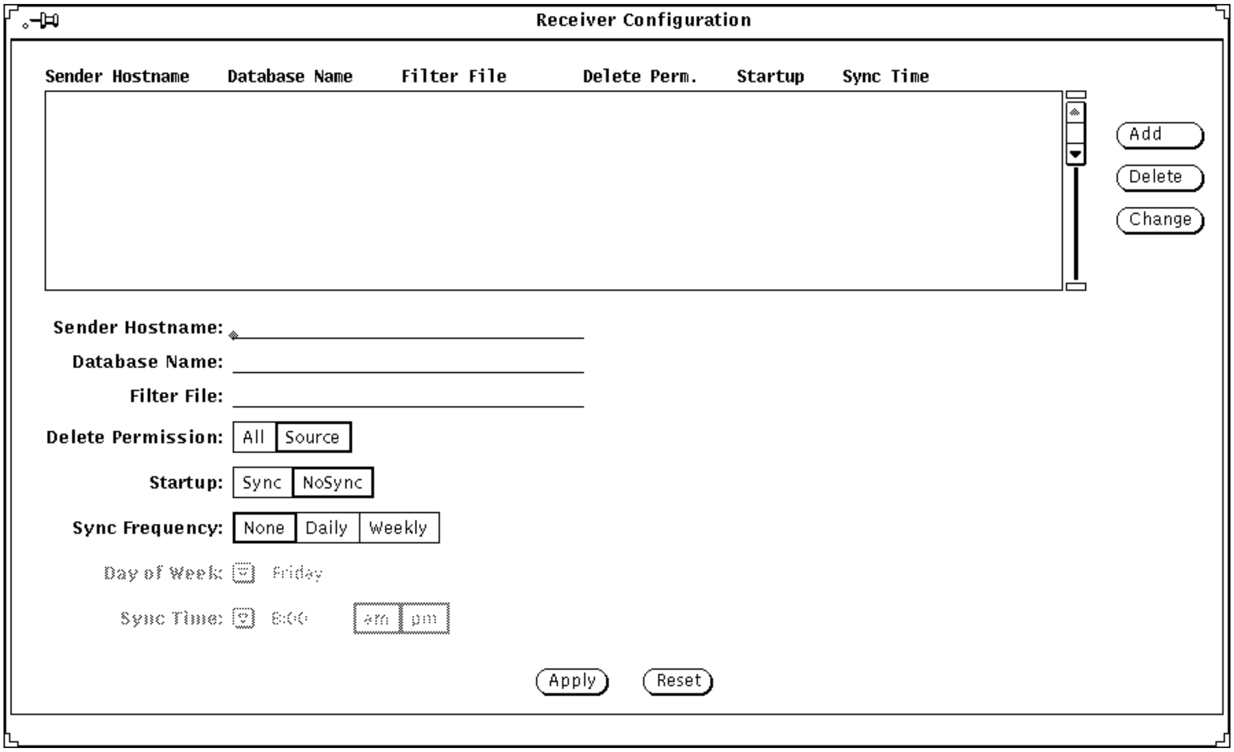


Figure 4-3 Receiver Configuration Window

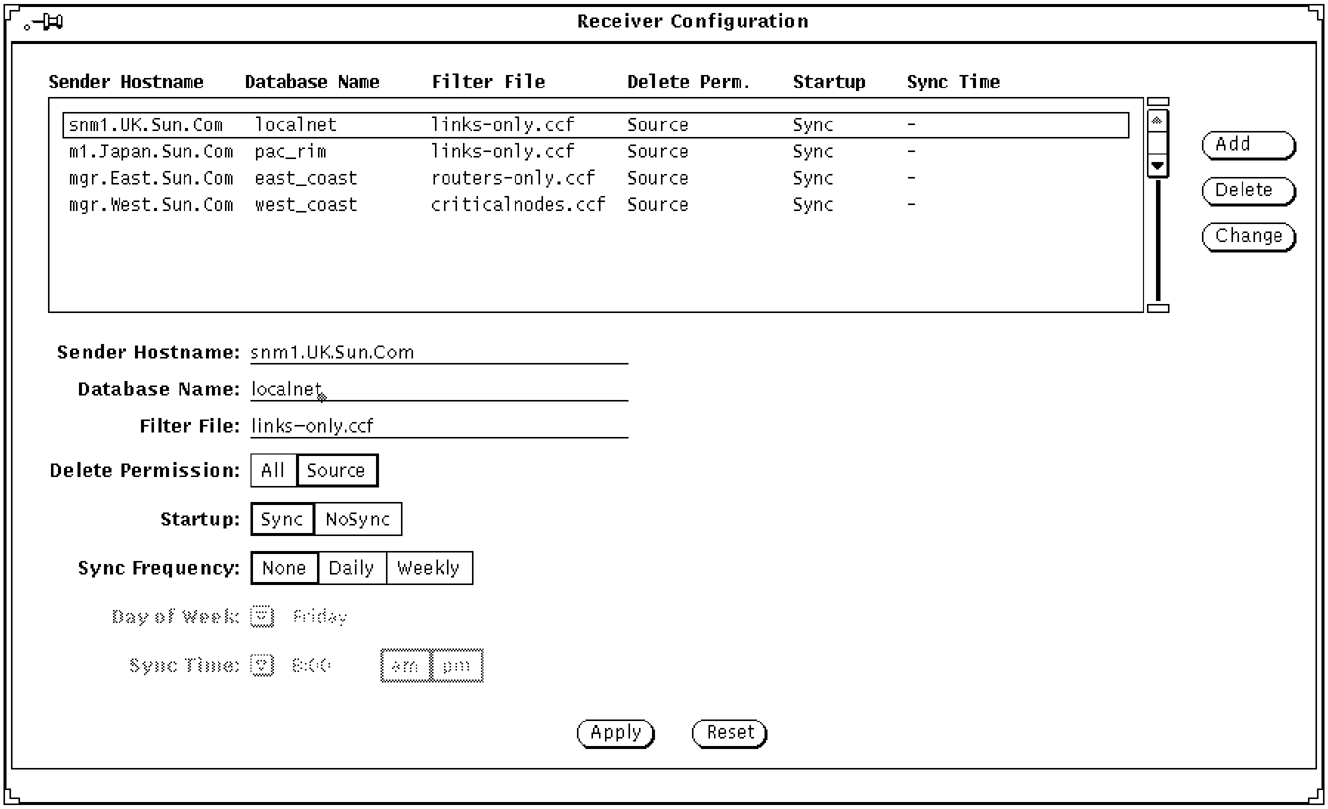
The Receiver Configuration window allows you to build the registration list that determines the remote management stations the local Receiver process will attempt to register with when it is launched. You can also specify the manner of automatic synchronization of the receiving station with remote sending stations.

4.1.1 Adding a New Entry to the Registration List

To add a new entry to the Registration List, you will need to first type in the appropriate information in the three fields at the bottom of the list window:

- **Sender Hostname** — This is the host name of a remote SNM Console machine where a Sender daemon has been installed.
- **Database Name** — This is the instance name (SNM_NAME value) of the runtime database on the remote machine that the Receiver requests access to when it registers with the Sender daemon on the remote machine. For example, if the database file is named `db.localnet`, then the name that should be entered in this field is “localnet”.
- **Filter File** — This is the name of the file on the remote machine that contains the Filter Table that is to be used to select event and topology information for forwarding.
- **Delete Permission** — This toggle allows you to control deletion of elements in response to deletions of element instances on remote sending stations. If Delete Permission is set to Source, the Receiver will delete an element in response to a delete trap from a remote station only if that connection was the source of the local element instance (as indicated by the `Created by cc` field on the local element’s properties sheet). If Delete Permission is set to All, then a delete trap received from a remote station will result in the deletion of that element on the local Console even if that connection was not the source of the local instance of that element.
- **Startup** — This is a toggle that determines whether the Receiver automatically executes a synchronization with the selected sending station whenever the Receiver is started.
- **Sync Frequency** — This option allows you to enable automatic synchronization of the selected connection at scheduled intervals. Automatic synchronization can be scheduled to occur either Daily or Weekly. If None is selected, scheduled synchronization is disabled.
- **Day of Week** — A pulldown menu enables you to select the day of week for scheduled automatic synchronization. This selection takes effect only if you have selected Weekly for Sync Frequency.
- **Sync Time** — A pulldown menu enables you to select the time of day for scheduled automatic synchronization. This selection takes effect only if you have selected Daily or Weekly for Sync Frequency.

After you have added this information, click SELECT on the Add button to add the new entry to the Registration List. A sample Registration List is shown in Figure 4-4.



Receiver Configuration

Sender Hostname	Database Name	Filter File	Delete Perm.	Startup	Sync Time
snm1.UK.Sun.Com	localnet	links-only.ccf	Source	Sync	-
m1.Japan.Sun.Com	pac_rim	links-only.ccf	Source	Sync	-
mgr.East.Sun.Com	east_coast	routers-only.ccf	Source	Sync	-
mgr.West.Sun.Com	west_coast	criticalnodes.ccf	Source	Sync	-

Sender Hostname: snm1.UK.Sun.Com

Database Name: localnet

Filter File: links–only.ccf

Delete Permission: All Source

Startup: Sync NoSync

Sync Frequency: None Daily Weekly

Day of Week: Friday

Sync Time: 8:00 am pm

Figure 4-4 Sample Registration List

4.1.2 Modifying an Entry in the Registration List

If you want to change one of the existing entries in the Registration list, do the following:

1. **Click SELECT on the entry that you want to change. The entry should now be highlighted.**

2. **The values for the three fields in the selected entry should now be displayed in the Sender Hostname, Database Name, and Filter File fields at the bottom of the screen. Edit these fields to make your desired changes.**
3. **Click SELECT on the Change button to make your changes take effect.**

4.1.3 Deleting an Entry from the Registration List

To delete one of the entries from the Registration List, click SELECT on the entry to highlight it and then click SELECT on the Delete button.

4.1.4 Saving the Registration List

After you have finished adding or modifying entries in the Registration List, be sure to click SELECT on the Apply button to save your changes.

4.1.5 Abandoning Changes to the Registration List

If you want to abandon the changes you have made to the Registration List and revert to the previously saved version of the Registration List, click SELECT on the Reset button.

4.2 Configuring the Sender Daemon

If you select the Sender button from the CC Configuration main menu, the Sender Configuration window appears, as shown in Figure 4-5.

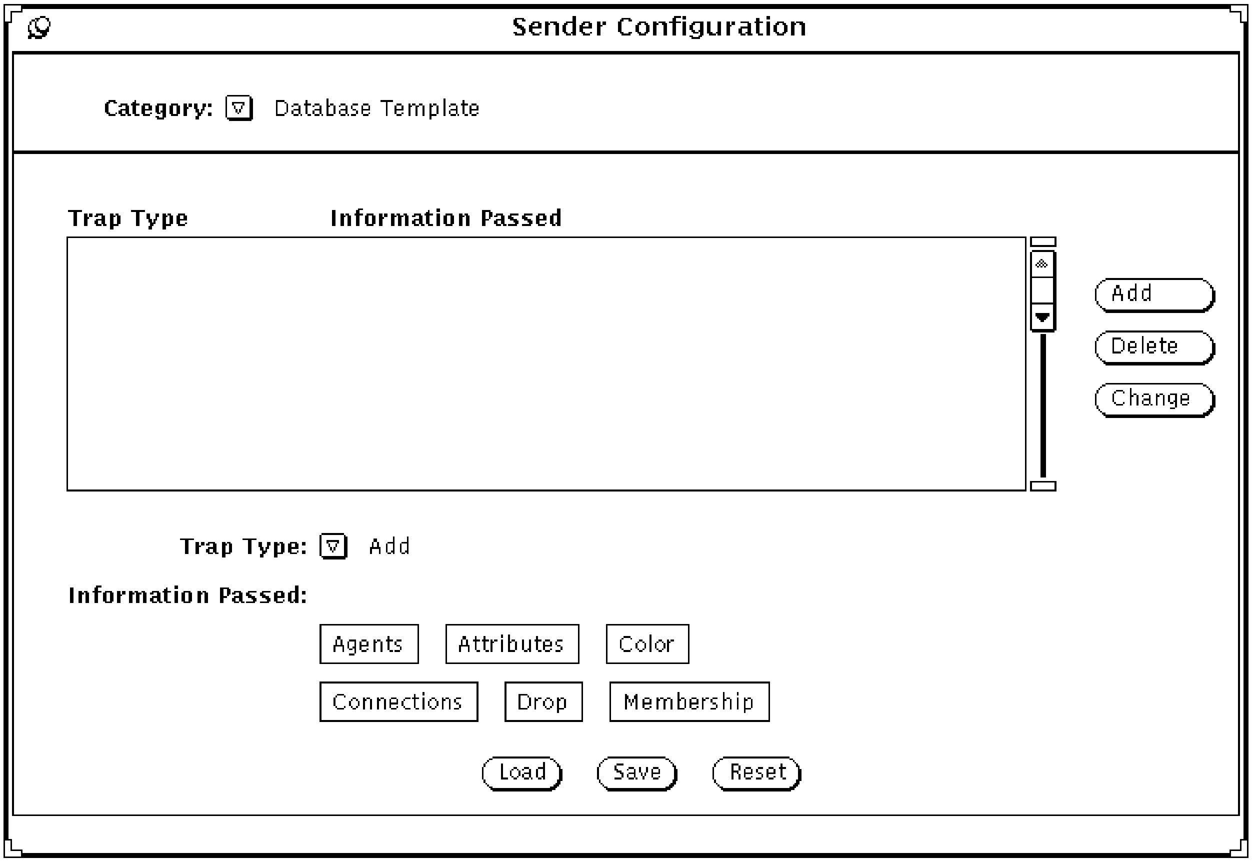


Figure 4-5 Sender Configuration Window

If you press the MENU button over the Category button, as shown in Figure 4-6, a menu appears with three options:

- **Database Template** — Select this Category if you want to define the topology information that the Sender daemon is to forward to specified Receiver processes. Multiple DB Templates can be defined**.**

- **Authorization List** — Select this Category if you want to define the list of remote receiving stations that are authorized to register with the local Sender and the database instances they are authorized to access.

Note – If you do not set up a list of SNM Console hosts that are authorized to register with the local Sender daemon, then any host will be allowed to register.

- **Filter Table** — Select this Category if you want to define the selection criteria to use when forwarding event and topology information to receiving stations. Multiple filter tables can be defined.

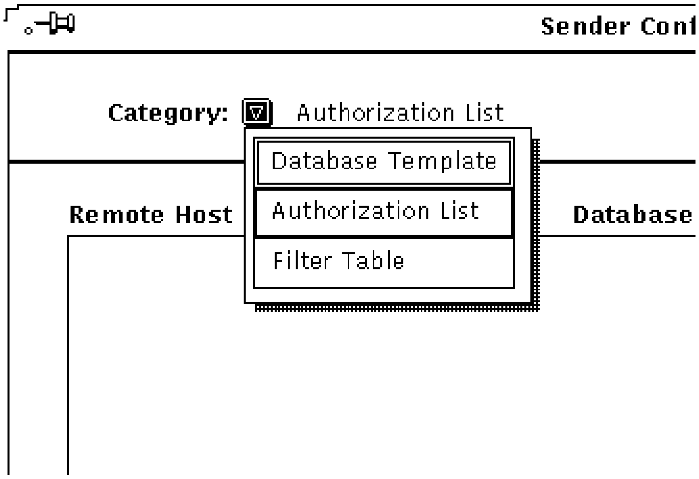


Figure 4-6 Sender Configuration Category Options

4.2.1 Setting Up the Authorization List

As shown in Figure 4-7, the local Sender's authorization table contains two columns:

- **Remote Receiver Host** — Names of SNM host machines that are authorized to register with this Sender
- **Local Database Name** — A database name (SNM_NAME value) that the remote SNM host is authorized to use in accessing the runtime management database on the local machine

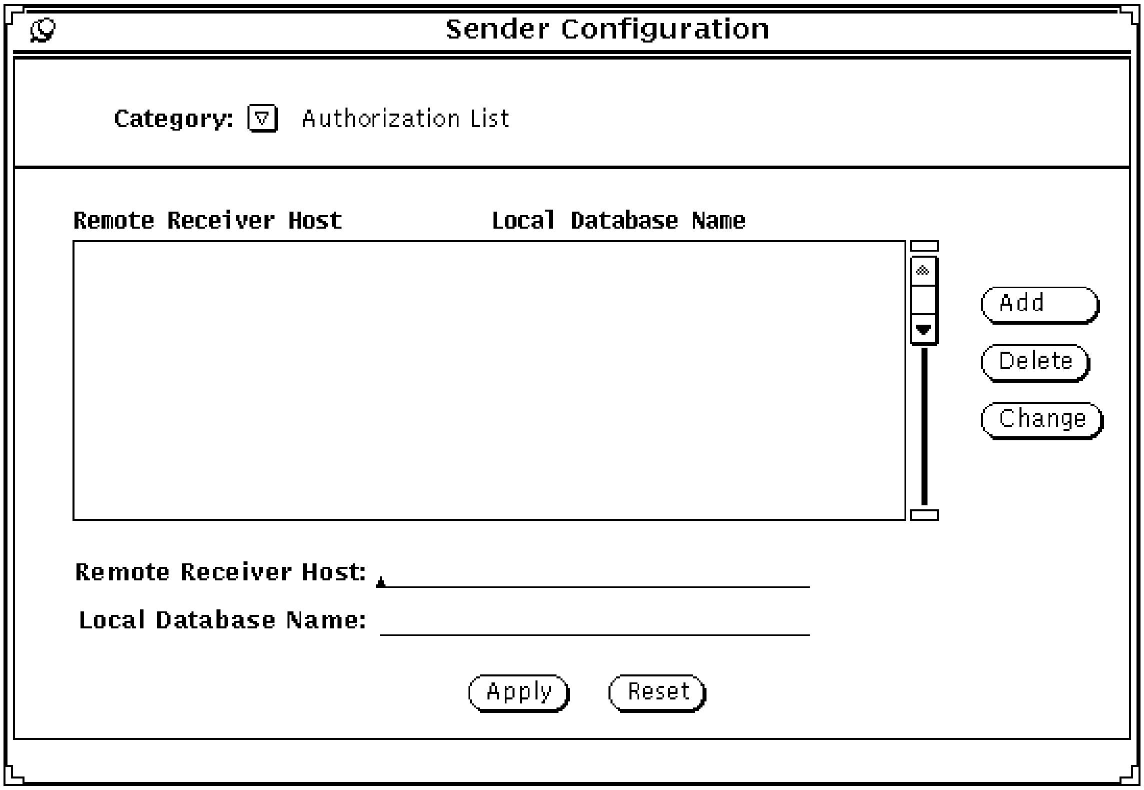


Figure 4-7 Authorization List Properties Sheet

To enable you to create and modify rows in the Authorization List, there are two fields at the bottom of the window:

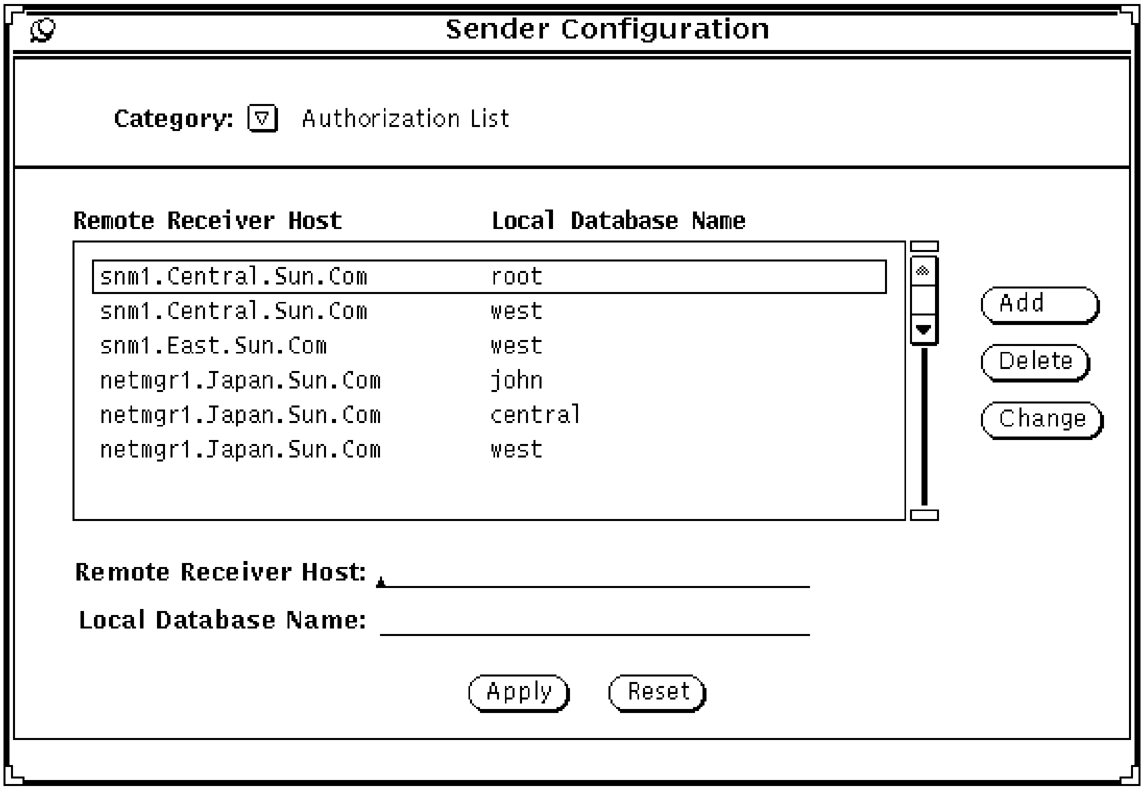
- Remote Receiver Host
- Local Database Name

4.2.1.1 *Adding a Remote Host to the Authorization List*

If you want to enter a new row in this list, type in the host name of the receiving station in the Remote Receiver Host field. Also type a database name (SNM_NAME value) in the Local Database Name field. For example, if you want to authorize a remote host to access a database named `db.localnet`, you would enter "localnet" in the Local Database Name field.

When you have entered the information that you wish to specify for the receiving station, click SELECT on the Add button to add the specified host to the authorization list.

You can make multiple entries in the authorization list for the same remote host if that host is authorized to access more than one database, as shown in Figure 4-8.



Category: Authorization List

Remote Receiver Host	Local Database Name
snm1.Central.Sun.Com	root
snm1.Central.Sun.Com	west
snm1.East.Sun.Com	west
netmgr1.Japan.Sun.Com	john
netmgr1.Japan.Sun.Com	central
netmgr1.Japan.Sun.Com	west

Remote Receiver Host:

Local Database Name:

Figure 4-8 Sample Authorization List

4.2.1.2 Changing an Entry in the Authorization List

If you want to *modify* one of the entries (rows) in the authorization table, click SELECT on that row to highlight it. The information for that host is now displayed in the Remote Receiver Host and Local Database Name fields at the bottom of the window. After you have changed these values, click SELECT on the Change button to enter the changes into the table.

4.2.1.3 Removing a Remote Host from the Authorization List

If you want to *delete* one of the entries (rows) in the authorization list, click SELECT on that row to highlight it, and then click SELECT on the Delete button.

4.2.1.4 Abandoning Changes to the Authorization List

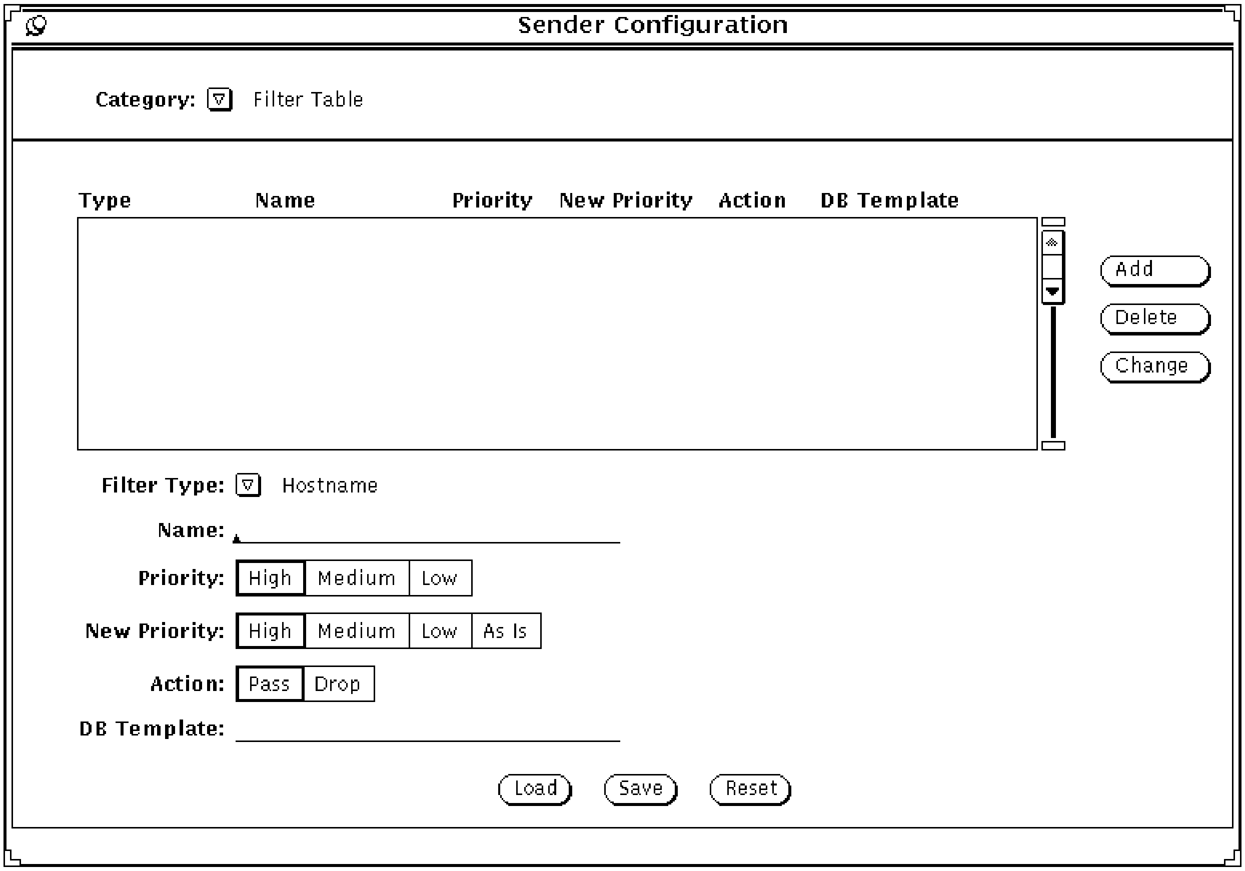
If you want to abandon the changes you have made to the Authorization List and revert to the previously saved version, click SELECT on the Reset button.

4.2.1.5 Saving the Authorization List

After you have added new entries or made other changes in the Authorization List, click SELECT on the Apply button to save your changes.

4.2.2 Setting Up the Filter Table

The selection criteria used by the local Sender in forwarding event, trap, and topology information to registered receiving stations is contained in one or more filter files. The entries in each filter file constitute a single Filter Table. When you first select Filter Table from the Sender Configuration Category menu, the Filter Table properties sheet will be blank, as shown in Figure 4-9.



Category: Filter Table

Filter Type: Hostname

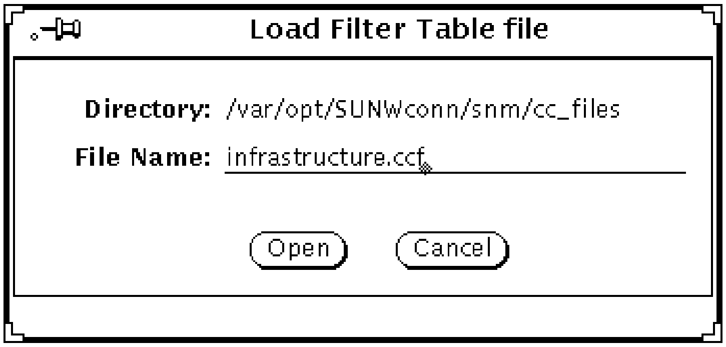
Priority: [✓ High] Medium Low

New Priority: [✓ High] Medium Low As Is

Action: [✓ Pass] Drop

Figure 4-9 Sender Filter Table Window

You can either load an existing filter file or define a new Filter Table. To load an existing filter file, click SELECT on the Load button. You will receive a load window as shown in Figure 4-10.



Directory: /var/opt/SUNWconn/snm/cc_files

File Name: infrastructure.ccf

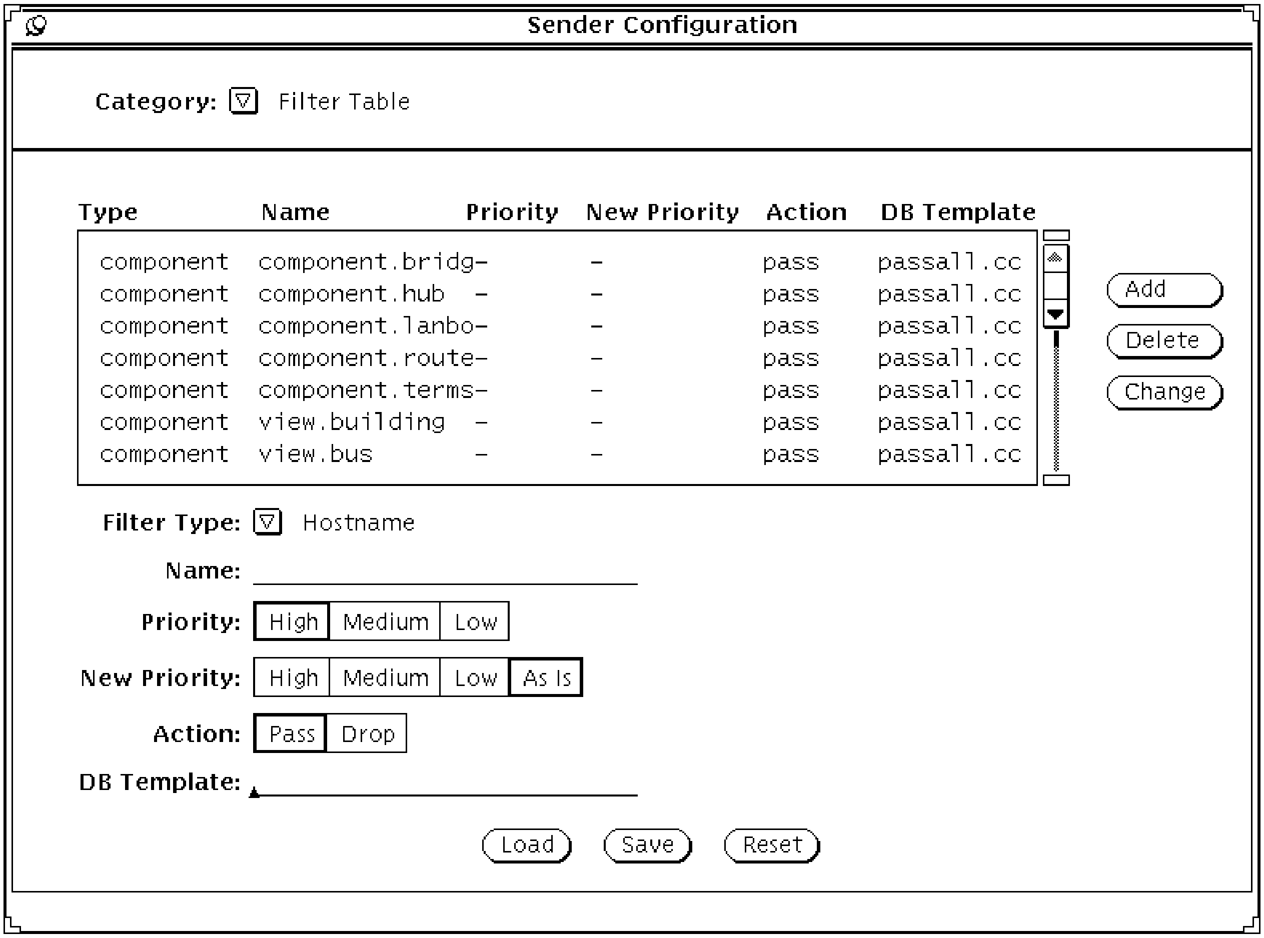
Figure 4-10 File Load Window

A sample Filter Table is shown in Figure 4-11. Three sample filter tables are included with the Cooperative Consoles product:

- `routers-only.ccf`
- `infrastructure.ccf`
- `filter_none.ccf`

These files are located in the following directory:

- `/var/adm/snm/cc_files` on Solaris 1.x environments
- `/var/opt/SUNWconn/snm/cc_files` on Solaris 2.x environments



Sender Configuration

Category: Filter Table

Type	Name	Priority	New Priority	Action	DB Template
component	component.bridg-		-	pass	passall.cc
component	component.hub	-	-	pass	passall.cc
component	component.lanbo-		-	pass	passall.cc
component	component.route-		-	pass	passall.cc
component	component.terms-		-	pass	passall.cc
component	view.building	-	-	pass	passall.cc
component	view.bus	-	-	pass	passall.cc

Filter Type: Hostname

Name:

Priority: [✓ High] Medium Low

New Priority: High Medium Low [✓ As Is]

Action: [✓ Pass] Drop

DB Template:

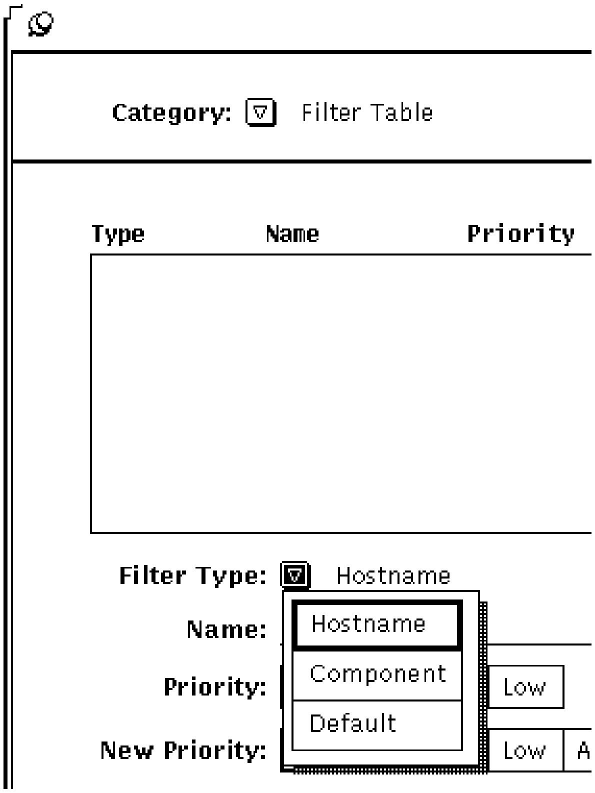
Figure 4-11 Sample Filter Table

Each row in the table is a filter that defines selection criteria and action to be taken if the event or trap matches the selection criteria. A device trap or SNM event will be selected by a filter if it matches the criteria defined by the Name and Priority fields in that filter. (The Type field determines the type of entry that is valid in the Name field — either a host name or the name of a component type.) Topology traps will be selected if they match the Name field in the filter.

Events and traps are checked against the filters in the table beginning at the top of the table. The event or trap will be processed by the method specified in the first filter whose selection criteria it matches.

4.2.2.1 Adding a New Filter

To add a new filter to a Filter Table, you need to select the filter type. The filter types are listed if you press MENU on the Filter Type button, as shown in Figure 4-12.


Category: Filter Table

Filter Type: Hostname

Figure 4-12 Filter Type Menu

There are three possible filter types:

- **Hostname Filter** — If you select Hostname as the filter type, then the filter is to select events or traps on the basis of host name match. For example, you might want to use a Hostname type filter to select critical nodes by name in order to forward events or traps relating to those devices. If you have selected a Hostname filter, you will need to enter the host name in the **Name** field.
- **Component Filter** — If you select Component as the filter type, then the component type name (for example, `component.router`) is used to select events and traps. For example, you might want to use a Component type filter to forward events or traps pertaining to all devices of a certain type such as routers. If you have selected a Component filter, you will need to enter the component type name (the type name used in the element's schema file) in the **Name** field.
- **Default Filter** — If the event or trap is not selected by any of the filters in the table, the action specified by the Default filter will be used. A Filter Table can only have one Default filter.

In addition to the Host Name/Component Name field, the following fields can be used to complete the definition of a new filter:

- **Priority** — In the Priority field you specify the *lowest* priority of event or trap to be selected by this filter. For example, if you select Medium in the Priority field, then low priority events will not be selected by this filter.
- **New Priority** — When non-database traps are forwarded, their priority may be changed to a value specified in the **New Priority** field. This value will specify the priority of the trap at the receiving station. For example, if you want to change all events selected by this filter to be forwarded as low priority traps, click SELECT on Low. If you select the "As Is" button, the priority of a selected event when forwarded will be the same as the priority on the local Console.
- **Action** — You can use filters not only to select which events to forward but also which events or traps to ignore. The **Action** field in the filter may specify that the event or trap is to be forwarded or that it is to be dropped. Click SELECT to highlight the appropriate action in the **Action** field.
- **DB Template Name** — The **DB Template Name** field specifies the file name of a database template that is to be used in selecting the topology information to be forwarded when SNM database traps match the filter's **Name** field. The role of the DB Template is discussed in Section 4.2.3, "Specifying Topology Information for Forwarding.".

After you have you made all your selections for a single filter, click SELECT on the Add button if you want your selections to define a new filter (row) in the table.

4.2.2.2 *Changing a Filter*

To change the values of an existing filter in the filter table, do the following:

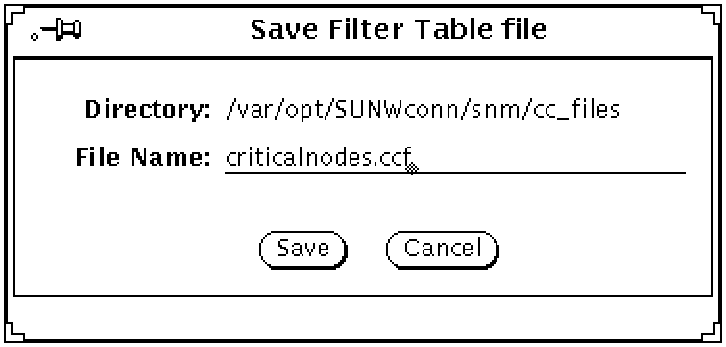
1. **Click SELECT to highlight the filter in the table.**
2. **Enter the values in the Filter Type, Host Name (or Component Name), Priority, New Priority, Action, and DB Template Name fields.**
3. **Click SELECT on the Change button to modify the values of the selected filter.**

4.2.2.3 *Deleting a Filter*

To delete a filter in the Filter Table, click SELECT on that filter to highlight it and then click SELECT on the Delete button.

4.2.2.4 *Saving the Filter File*

After you are done adding new filters, or deleting or modifying existing filters in the Filter Table, click SELECT on the Save button to save your changes to this filter file. You will be prompted to enter the name of the filter file, as shown in Figure 4-13.



Directory: /var/opt/SUNWconn/snm/cc_files

File Name: criticalnodes.ccf

Figure 4-13 File Save Window

4.2.2.5 Abandoning Changes to the Filter File

If you want to abandon the changes you have made to the Filter Table and revert to the previously saved version of this filter file, click SELECT on the Reset button.

4.2.3 Specifying Topology Information for Forwarding

The SNM Console generates traps in response to changes in the runtime database. When an SNM database trap is generated, the Sender daemon uses a the Filter Table specified by a remote Receiver process to determine whether the trap should be forwarded to that Receiver process. The Filter Table also specifies a Database Template file to determine the topology information that should be forwarded.

The Database Templates that are used for selecting topology information to forward to a specific Receiver process will depend on the filter file that was requested by the Receiver at the time it registered with the local Sender daemon.

Each filter in the filter file can specify a DB Template file to use for elements that match the selection criteria of that filter. The element is selected by a filter when one of the following conditions is met:

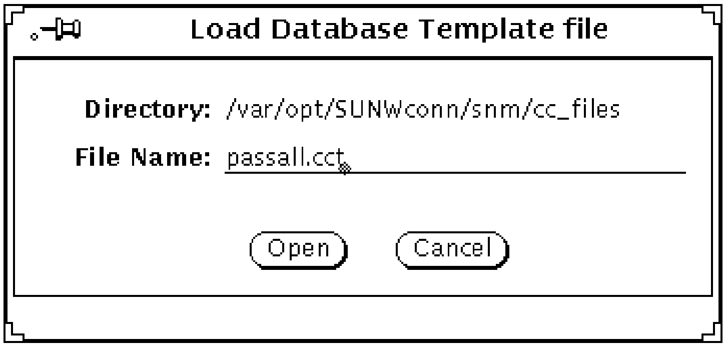
- the element has a specified host name (for filters of type Hostname)
- the element is of a specified component type (for filters of type Component)
- the element is not selected by any other filter in the filter file (for the filter of type Default).

DB Template files are only accessed in response to SNM database traps. Non-database traps or events do not cause the Sender to forward topology information to remote Receiver processes.

4.2.3.1 Setting Up the DB Template

To configure a DB Template file, select DB Template from the Sender Configuration Category menu. When initially selected, the DB Template window is blank, as shown in Figure 4-15. You can either load an existing DB Template file or create a new DB Template.

If you want to load an existing DB Template file, click SELECT on the Load button. A DB Template Load window will appear, as shown in Figure 4-14.



Directory: /var/opt/SUNWconn/snm/cc_files

File Name: passall.cct

Figure 4-14 DB Template Load Window

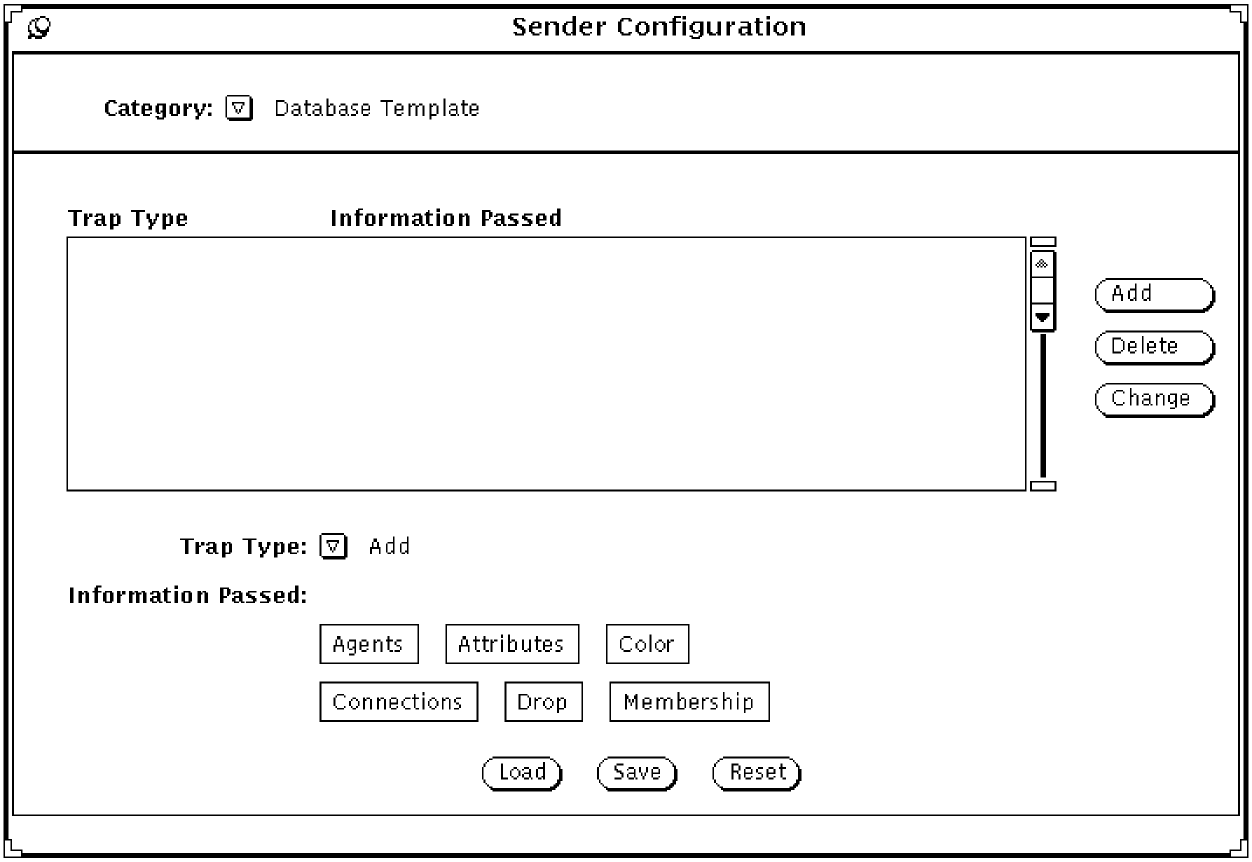


Figure 4-15 Database Template Window

A Database Template can contain six entries (rows). Each entry corresponds to one of the following types of SNM database trap:

- **Add** — Generated when a new element is added by the database Application Programmatic Interface.
- **Create** — Generated when a new element is created via the SNM Console.
- **Change** — Generated when attributes of the element (such as the agents list or the screen coordinates) are changed.
- **Delete** — Generated when an element is deleted from the database.
- **Load** — Generated when a new management database is loaded.
- **Background** — Generated when a background image is added to a view.

To specify the type of action to be taken for one of these SNM database trap types, there must be an entry for that type in the template.

4.2.3.2 Adding a Trap Type Entry

To add an entry for one of the SNM database trap types, press MENU over the Trap Type button, as shown in Figure 4-16, and select one of the four trap types.

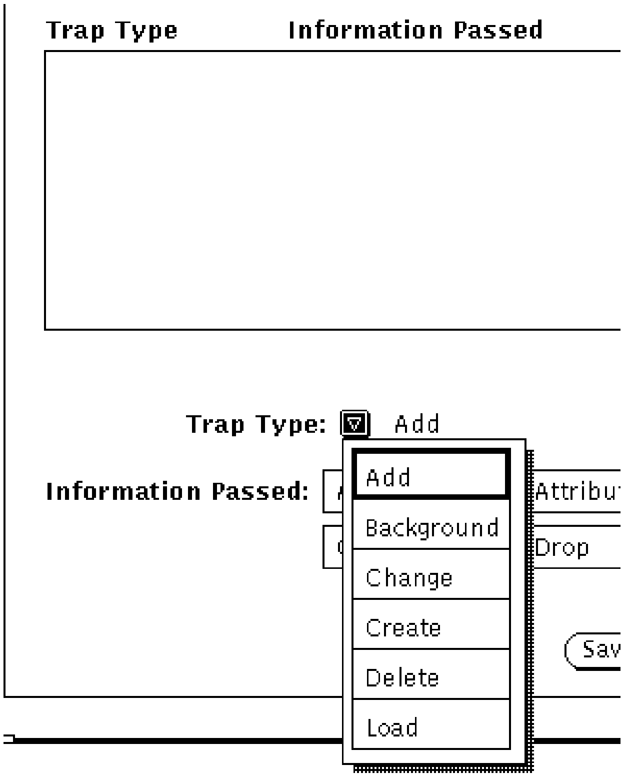


Figure 4-16 Trap Type Menu

The action to be taken for database traps of the selected type can be defined using the five buttons along the bottom of the template window:

- **Drop** — Select this button if you want all topology traps of this type to be ignored (not forwarded). If this button is selected, the other buttons are ignored.
- **Membership** — If selected, the Viewname of each view the element belongs to is forwarded as is the screen position (coordinates) of the element within that view.
- **Color** — If selected, the element's RGB values are forwarded.
- **Agents** — If selected, a list of the agents checked off on the properties sheet is forwarded, along with the proxy system name for proxy agents.
- **Attributes** — If selected, the attributes of the element as defined in the element's schema are also forwarded
- **Connection** — If selected, information about simple connections between the selected element and other elements is forwarded.

Each of these buttons is a toggle — click SELECT on the button to turn it off or on. The button is on if it is highlighted.

After you have selected the desired combination of values for these five toggles, click SELECT on the Add button to add an entry for this trap type to the template.

Note – Color, Agents, Connection, and Attributes buttons have no effect for database traps of types Background or Delete.

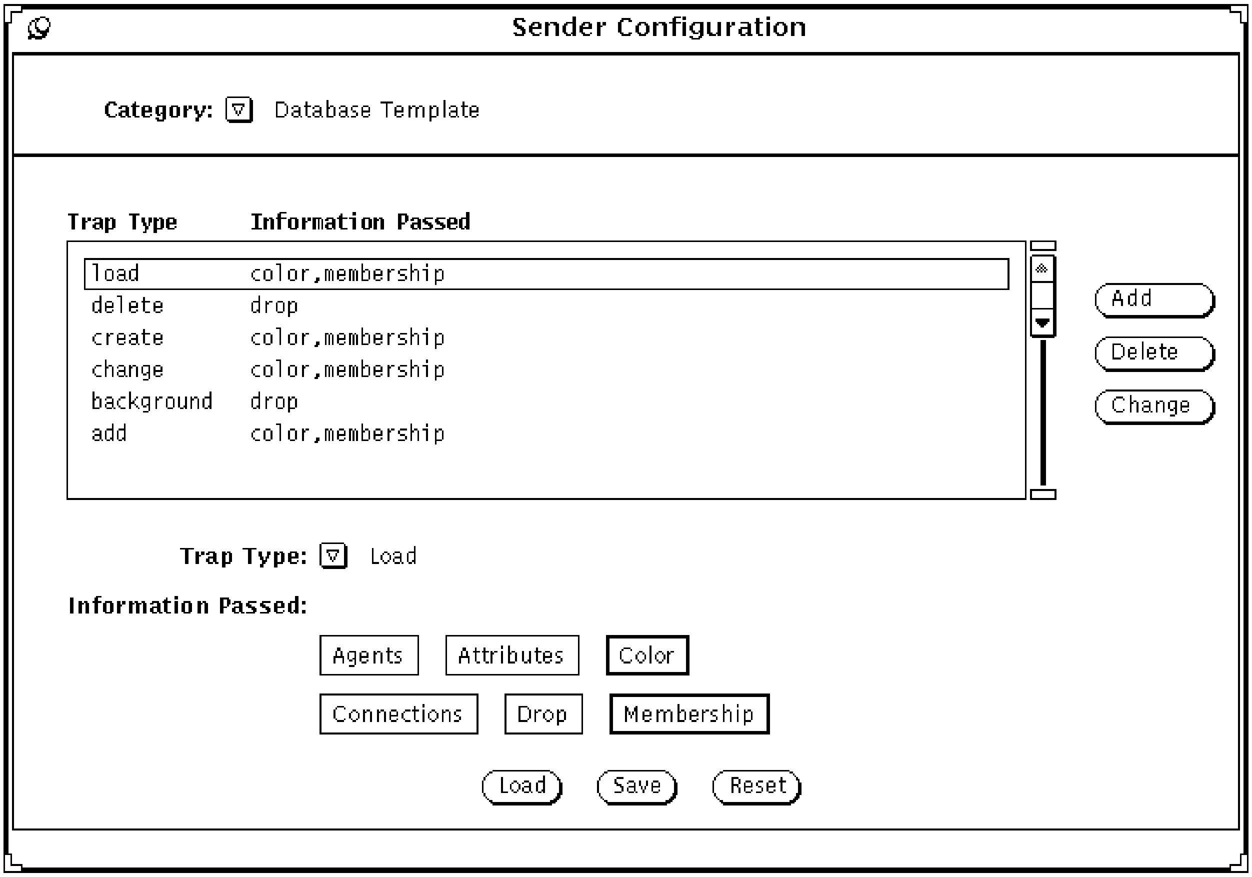
A sample DB Template is shown in Figure 4-17. Two sample DB Template files are shipped with the Cooperative Consoles product:

- `passall.cct`
- `views-only.cct`

These files are located in the following directory:

- `/var/adm/snm/cc_files` on Solaris 1.x environments
- `/var/opt/SUNWconn/snm/cc_files` on Solaris 2.x environments

If the DB Template shown in Figure 4-17 were specified by a filter row, then only color (RGB values) and view-membership information would be passed for the selected elements.



Sender Configuration

Category: Database Template

Trap Type	Information Passed
load	color,membership
delete	drop
create	color,membership
change	color,membership
background	drop
add	color,membership

Trap Type: Load

Information Passed: Color, Membership

Figure 4-17 Sample DB Template

4.2.3.3 *Changing the Forwarding Specified for a Trap Type*

If you want to modify the action specified for a trap type in the DB Template, do the following:

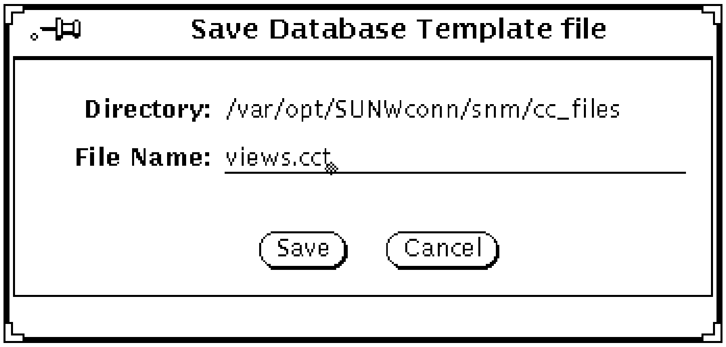
1. **Click SELECT on the trap type entry in the template to highlight it. The five buttons along the bottom of the window should be highlighted to indicate the selected values for that trap type.**
2. **Make the desired changes in the forwarding action for this trap type by clicking SELECT on the Drop, Membership, Attributes, Agents, or Color buttons.**
3. **Click SELECT on the Change button to apply your selections to the DB Template entry.**

4.2.3.4 *Deleting a Trap Type Entry*

You can delete the existing entry in the DB Template for a particular trap type if you click SELECT on that entry to highlight it and then click SELECT on the Delete button.

4.2.3.5 *Saving the DB Template*

After you are done adding new trap type entries, or deleting or modifying existing trap type entries in the DB Template, click SELECT on the Save button to save your changes to this DB Template file. You will be prompted to enter the name of the DB Template file, as shown in



Save Database Template file

Directory: /var/opt/SUNWconn/snm/cc_files

File Name: views.cct

Figure 4-18 DB Template Save Window*

4.2.3.6 Abandoning Changes to the DB Template

If you want to abandon the changes you have made to the DB Template and revert to the previously saved version of this DB Template file, click SELECT on the Reset button.

Index