Defining and Executing Role-Based ServiceControl Manager Tools

DONALD SUIT
HEWLET PACKARD
3404 EAST HARMONY ROAD, MS 99
FORT COLLINS, CO 80528-9599
(970) 898-0327
(970) 898-2151 FAX
donald_suit@hp.com

FEBRUARY 28, 2001

This page intentionally blank

Biographical Sketch

Donald Suit is a software engineer at Hewlett-Packard's Manageability Solutions Lab in Fort Collins, Colorado. He has 20 years of software development experience developing various commercial and government applications. He has a B.S. degree in computer science from the University of Maryland and an M.S. degree in computer science from Johns Hopkins University.

Mailing address:

Donald Suit
3404 East Harmony Road, MS 99
Fort Collins, CO 80528-9599

E-mail address:

donald_suit@hp.com

1 Introduction

The ServiceControl Manager (SCM) tool feature provides a user-configurable, role-based mechanism for defining procedures (tools) that system administrators can use to manage from a central management server (CMS) and direct to execute across a cluster of SCM managed nodes. System administrators can define tools to copy files and/or execute processes. In the ServiceControl environment, system administrators can define roles, such as database administrator or web administrator, which the system administrators can associate with SCM tools. The authorization facility of the SCM system associates users with roles and nodes within an SCM managed cluster. With authorizations in place, SCM users can execute SCM tools from the CMS, and the tool facility can automatically determine on which SCM managed nodes the tool has authorization to execute and send the tool to execute on those managed nodes.

2 Purpose

The SCM tool feature provides a mechanism to distribute routine or special-purpose tasks across a managed cluster of HP-UX nodes.

The roles and authorizations, defined in the SCM, provide a more secure distributed mechanism than remote shell. SCM users can only run an SCM tool with the user's authorized roles on the user's authorized nodes. When an SCM user runs an SCM tool, SCM audits the execution of the tools. SCM logs that it executed the tool and logs the SCM user that executed the tool. Optionally, the SCM can log the output of the tool. Additionally, by using the SCM authorization feature, an SCM user can define a tool to run as a privileged user, such as root, but the SCM user logs on to the CMS as him/herself. Using SCM, therefore, does not require that all SCM users have super user privileges to administer distributed systems. Because SCM identifies users as themselves, not root, the audit data in the SCM log more accurately portrays the activity occurring on the CMS and the managed nodes.

SCM users can package their favorite scripts or any other command line-based tasks in an SCM tool and use the SCM to distribute and execute the tool across the network of SCM managed nodes. Using the SCM tool feature, SCM users can build a collection of SCM toolkits for various purposes, for example trouble-shooting or system performance analysis.

Besides executing scripts and commands, the SCM tool feature provides the capability to copy files to the SCM managed nodes. Using the file-copying capability, users can define tools that distribute configuration files or scripts to managed nodes, then execute commands that use the copied files.

Additionally, HP provides a set of built-in SCM tools for the convenience of SCM users. The built-in tools package some of the capabilities of other HP system management products. These include Ignite-UX (IUX), Software Distributor (SD), System Administration Manager (SAM), System Configuration Repository (SCR), Event Monitoring Service (EMS), and general purpose UNIX commands, such as ls, mv, find, bdf, and ps. SCM users can view the definitions of HP-provided tools and use the HP-provided tools as templates for their own customized tools.

3 ServiceControl Manager Overview

The ServiceControl Manager is used to manage a collection of HP-UX nodes from a single CMS. The primary purpose of the SCM environment is to increase the efficiency of managing multiple HP-UX systems. The focus of SCM is the ongoing administration and configuration of HP-UX server systems. SCM provides the means for administrators to perform tasks on multiple HP-UX nodes in parallel. SCM 1.0 does not support s700 workstations.

SCM runs on a CMS and, from the CMS, manages the nodes. The CMS is an HP-UX 11.x server running the SCM software. Because the CMS does not use excessive resources, IT personnel may use the CMS server for other purposes. There are some specifics regarding the CMS system configuration that are required (including such things as kernel tunables, file system space, and the configuration of certain daemons). The SCM installer may find these system specifications in the SCM technical reference manual found at URL http://www.software.hp.com/products/scmgr/info.html.

The current version of SCM supports a single instance of SCM on a single CMS. The SCM administrator initiates all tasks performed on the SCM cluster from the CMS, although the CMS is accessible via a web interface. The workstation, at which an SCM user sits, must be WEB accessible. The CMS is the host for the SCM software, the SCM data repository, a web server that allows web access to SCM, an SD depot containing products used in the configuring of nodes, and an Ignite-UX server.

SCM supports multiple SCM users performing tasks. There are some limits in how many simultaneous tasks can be performed depending upon the number of target nodes performing the tasks and the disk and memory resources of the CMS. SCM users, therefore, may encounter error messages that indicate that the SCM cannot start a task due to resource constraints.

Additionally, on each SCM managed node, SCM agent software runs to complete the file-copying process and execute single system aware SCM tasks.

See the paper “Service Control Manager” also in these proceedings for more detailed information on ServiceControl Manager.

3.1 Users

SCM uses the HP-UX user authentication mechanism. To allow access to the SCM command line, SCM relies on the authentication done at login and does not require any additional passwords. For SCM users accessing SCM via the web, the user's normal HP-UX login name and password will be requested and validated in a way similar to login (1). This decision is a direct result of customer data indicating that the creation of SCM-unique user IDs and passwords is unacceptable. An SCM user is really just an SCM registered HP-UX user that has some SCM privileges and/or SCM authorizations (defined later). In order to start the SCM or execute any SCM tools, an existing privileged SCM user must add the other user to the SCM via the graphical user interface (GUI) or the command line interface (CLI). During the initial SCM configuration, the SCM adds two initial users to the SCM, root and a specified user. These initial users have the Trusted User privilege discussed later. The definition of an SCM user consists of:

- The SCM user's HP-UX login name,
- An SCM-specific description (optional), and
- A Trusted User privileges flag.

SCM is able to access the HP-UX user registry (/etc/passwd or NIS/NIS+) to get and display user information, including:

- The user's real name,
- Telephone number, and
- Office location.

An SCM user can, depending upon the assigned SCM authorizations, manage systems known to the SCM via the CMS. In addition, an SCM user can examine the SCM log, and scan the node, node group, role, authorization, and tool configurations. To manage the ServiceControl Manager itself, SCM institutes a Trusted User privilege.

An SCM Trusted User is an SCM user responsible for the configuration and general administration of the ServiceControl Manager. SCM provides the following capabilities to a Trusted User:

- Add/Modify/Delete a User
- Add/Modify/Delete an authorization
- Add/Modify/Delete a Node
- Add/Modify/Delete a Node Group
- Add/Modify/Delete a Tool
- Rename a Role

The granting of the Trusted User privilege implies a trust that the privileged SCM user will responsibly configure and maintain the overall structure of the ServiceControl Manager.

3.2 Managed Nodes

In SCM, the managed HP-UX systems are termed “managed nodes” or simply “nodes”. The managed node represents a single instance of HP-UX running on some hardware. HP designed the SCM to manage HP-UX system instances in any of these forms.

An SCM privileged user must explicitly add nodes other than the CMS to the set of SCM managed nodes. The process of adding a node to a SCM includes a minimum of installing an SCM agent and some SD products from the CMS onto the node and performing an “add node” operation on the CMS.

SCM nodes run a managed node agent that performs the management tasks sent to them from the CMS. In addition, there will likely be some client-side software installed on each node associated with the various management applications supported in SCM. For example, this includes the SD agent and the Desktop Management Interface (DMI) agent. These are all items that are included when SD installs the SCM agent product on a new node.

The user interface runs only on the CMS, although its display is accessible over the network via the web or X windows. No SCM node, other than the CMS, is capable of executing remote tasks, accessing the repository, or any other SCM operations. The managed nodes are capable of contacting the CMS after the agent install process. The agent notifies the CMS that it is accessible. However, the managed node cannot initiate actions on the CMS.

An SCM Trusted User may also remove an SCM managed node from the SCM domain. The following steps occur during the removal of a node from the CMS:

- Remove the node from any node groups of which it is a member
- Remove any authorizations for the node
- Remove the node from the repository

This action removes no software SCM-related or otherwise from the managed node. Rather, the SCM deletes the node's CMS context, i.e. its presence in the repository, node groups and authorizations.

3.3 Node Groups

ServiceControl Manager node groups are collections of SCM managed nodes. Node groups can have overlapping memberships, such that a single node can be a member of more than one group. SCM node groups provide a convenient way to specify multiple targets for distributed tasks and multi-system aware applications. The node group mechanism allows flexible partitioning of the SCM. Customers can use the node group mechanism to reflect the way they logically associate nodes in their environment.

A node group definition consists of a group name, a list of members, and a description.

Any SCM user (defined below) may view node groups. Only privileged SCM users may configure node groups.

Additionally, an SCM Trusted User may authorize an SCM user to assume a specified SCM role on a set of SCM managed nodes represented by an SCM node group.

3.4 Roles

The ServiceControl Manager role provides a mechanism to logically group SCM tools with a system management function. One may think of an SCM role as a toolkit. The SCM may associate each role with one or more tools and may associate each tool with one or more roles. SCM authorizes an SCM user to perform some limited set of functionality on one or more nodes or node groups. The SCM bases an authorization upon roles and not on tools. The SCM role divides the sum total of functionality represented by all the tools into logical sets that correspond to the responsibilities that would be given to the various administrators.

For example, one SCM role might be that of performing backups. The "backup" role would be associated with tools that perform backups, manage scheduled backups, view backup status, etc. In this example, an SCM user may have the "backup" role for a group of systems that run a specific application. Later, when new backup tools are created and associate the tools with the "backup" role, the authorized users have immediate access to the new tools on those systems.

ServiceControl Manager supports a fixed number of roles, currently set at 16. There is one immutable role, called "Master Role". However, SCM privileged users can rename or disable the other 15 SCM roles. This allows granting of authorizations to an SCM user who only has periodic access to the systems. Thus, the trusted user could enable roles for auditors or an HP field engineer when needed.

"Master Role" is the default role assignment for all newly created tools. In fact, the SCM assigns all tools the "Master Role" and does not allow its removal. The "Master Role" is analogous to someone who knows the root password on a node and hence can do anything she wants. SCM users that have the "Master Role" on one or more nodes can run any tool on managed nodes for which the SCM user is authorized the "Master Role".

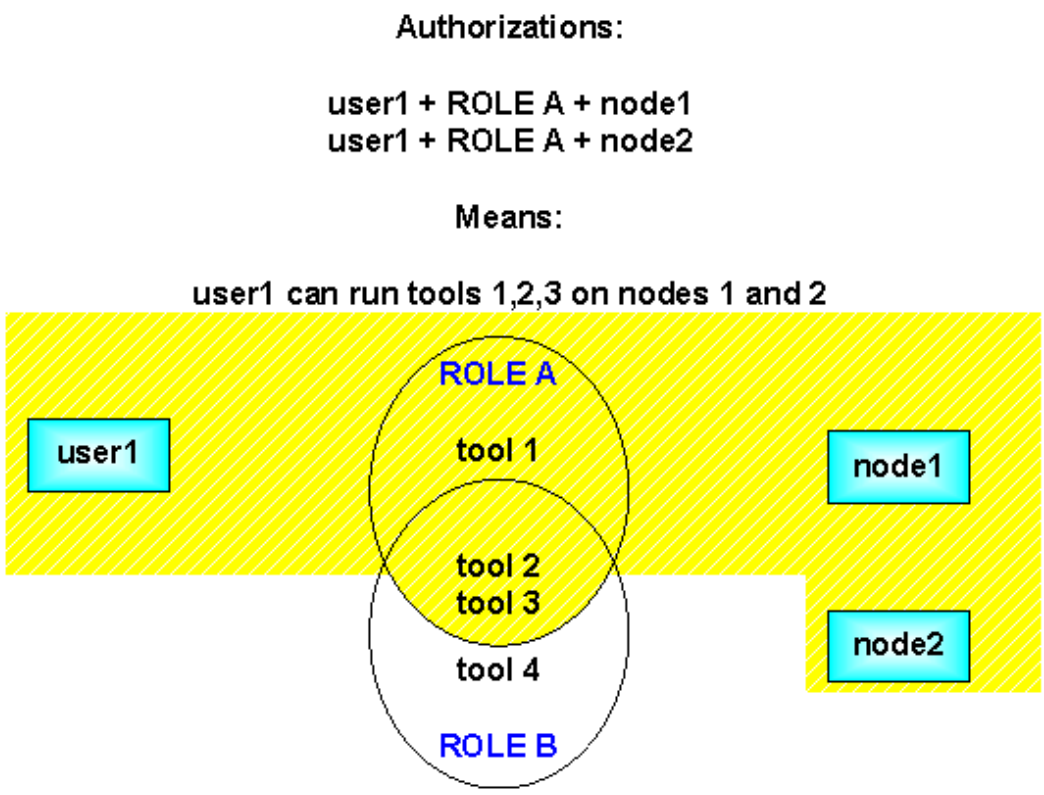
3.5 Authorizations

The ServiceControl Manager authorization model supports the notion of assigning to SCM users the ability to assume a specified role on some set of nodes or node groups. An SCM authorization is an association that links a user to a role on a managed node or node group. The role allows the IT trusted user to divide the total functionality represented by all SCM tools into logical sets that correspond to administrator responsibilities.

Each authorization links a single SCM user to a single role and to a single node or node group. Each role corresponds to one or more tools and each tool can be associated with one or more

roles. The trusted user can assign each SCM user multiple authorizations, as can each role and each node or node group.

In the following diagram, the SCM user “user1” has two authorizations. The first authorization assigns the SCM role “ROLE A” to “user1” on node “node1”. The second authorization assigns the same role to “user1” on node “node2”. Because tools named “tool 1”, “tool 2”, and “tool 3” are associated with “ROLE A”, the authorizations permit “user1” to execute those three tools on the managed nodes “node1” and “node2”.



The SCM authorization model for determining if an SCM user can execute a tool on a set of nodes is an “all or none” model. Therefore, the SCM user must have a valid authorization for each managed node or node group to execute the tool associated with the authorization’s role. If an SCM user specifies a list of nodes or node groups on which to run a tool, the tool execution fails for all of the nodes and node groups if one of the nodes or node groups does not have an authorization that allows the SCM user to run the tool on the node or node group.

The SCM simplifies the authorization process somewhat by allowing the SCM Trusted User to group logically related nodes into SCM node groups. An SCM Trusted User can then authorize the set of nodes by specifying the node group in the SCM authorization. The previous example could be simplified, somewhat, by assigning “node1” and “node2” to the node group “group1” and creating one authorization associating “user1”, “ROLE A”, and “group1”.

3.6 Tools

A central part of ServiceControl Manager is the ability to execute various management commands or applications on one or more managed nodes. The SCM users must wrap the commands or applications in an SCM tool. SCM users can define SCM tools that run simple commands like bdf (1) or mount (1M), or launch single system interactive applications such as SAM or EMS.

Additionally, SCM users can define tools to copy files from the CMS to the managed nodes. This provides the SCM user with the ability to push out copies of configuration files that need she needs to keep consistent across a specific set of nodes. This feature is not a replacement for the Software Distributor and is limited in the number (16 files) and type of files (only regular files - no device files, symlinks, etc.) copied. Users can only copy files from the CMS to the managed nodes - not from the nodes back to the CMS.

The combination of being able to copy a small number of files to a managed node and then to execute a specific command allows for SCM tools that copy a script to a managed node and then execute the script on the managed node. Another way to use the combination would be to copy one or more data files to a node (such as a crontab file) and then execute a command that uses the file contents. When the SCM copies the files from the CMS to a managed node, it also copies the ownership and permissions of the file on the CMS to the copy placed on the managed node. If the file in question already exists on the managed node in the destination location, then the SCM agent removes that file. If the specified directory does not exist on the managed node in the destination location, the file copy and the rest of the task fails.

Because the execution command string is not required for tools that specify one or more file copies, SCM users can create tools that only distribute files.

SCM users cannot define tools with neither a command line nor file-copy pairs.

3.7 Log

An integral part of the SCM functionality is the ability to record and maintain a history of events. Customers have expressed the need to track which users have initiated which events, with details such as launch time, managed nodes, and results. In response to this need, SCM logs configuration changes and task execution events.

SCM configuration changes include adding, modifying and deleting users, nodes, node groups, tools, authorizations or renaming roles in the CMS.

Task execution events include the details and intermediate events associated with the running of a tool. The details include the identity of the SCM user who launched the task, the task identifier, the task start time, the actual tool and command line with arguments, and the list of targeted managed nodes. The SCM logs intermediate events that include:

- The beginning of a task on a managed node
- Any exceptions that occur in attempting to run a tool on a node,
- The final result (if any) of the task
- The exit code, stdout and stderr (if they exist).

3.8 Repository

ServiceControl Manager requires a persistent centralized data repository for general management information. The ServiceControl Manager data repository uses the Lightweight Directory Access Protocol (LDAP) and the Netscape Directory Service (NDS) as its backing store.

All definitions of tools, users, nodes, node groups, roles, and authorizations are stored in the SCM repository.

4 ServiceControl Manager Tools In Depth

4.1 Tool Attributes

To identify and characterize a tool, SCM tools consist of the following attributes.

4.1.1 Name

The tool name is a unique identifier used to address the tool for addition, modification, removal, listing, or execution. SCM requires a unique name for every tool. The tool name is case insensitive; therefore, the name must differ from other tool names by characteristics other than case.

4.1.2 Category

The tool category is a name with which the operator relates this tool with other tools. This allows an operator to logically associate tools to aid in managing the tools. If the SCM user does not specify a category, the category defaults to “Local Tools”.

4.1.3 Description

The description is a short one-line description of the tool. The description is optional.

4.1.4 Owner

The owner attribute defines the SCM user that owns the tool. The value of the owner attribute determines who may modify the tool and the enabled roles for the tool.

If the trusted user specifies the owner, SCM enables only the “Master Role” role. SCM temporarily disables all other assigned roles. The SCM user, identified by the owner attribute, can modify the tool. If the current tool owner is not a privileged SCM user, however, the owner cannot modify roles or change the owner field. Privileged SCM users can modify all tool attributes.

If the owner field is empty, the SCM enables all roles assigned to this tool. Only the SCM trusted user can modify the tool. Enabling and disabling the tool’s roles controls affects the authorizations that will allow the tool to execute on the managed nodes. (See Roles below.)

4.1.5 Comment

The comment may be a multi-line text field to provide instructions, comments, warnings, etc. The comment is optional.

4.1.6 Command

The command attribute represents the invariant portion of the command line that the SCM will execute as the tool. If the SCM user provides no parameters to the tool definition, the command represents the entire command line of the tool. If the SCM user provides parameters to the tool definition, the SCM concatenates the parameters and argument values to the command to construct the tool’s complete command line.

When defining a tool definition, the SCM user must specify either a command or a file-copy pair or both (see below).

4.1.7 Parameters

Parameters are a list of argument entries. These are composed of an optional prefix, an optional prompt, and a flag indicating whether the argument is optional or required. The SCM user must define either the prefix or the prompt to create a valid parameter.

The prefix is an invariant string that SCM will concatenate to the command string as the SCM is building the command string for execution.

The prompt is an informational string used by the SCM GUI to describe a value that the SCM concatenates to the command string.

When building the execution command line, the SCM iterates through the list of specified parameters. If a parameter is required, the SCM looks for the prefix and, if it is present, appends the prefix to the command line. If a prompt is present, then the SCM looks for an operator-provided argument value. The SCM appends the argument value to the command line. If a parameter is not required, the SCM only concatenates the parameter if the operator provides a value for the argument. Because of this, SCM does not allow parameters specifications that are optional and do not provide a prompt.

Parameters are optional. However, to specify parameters, the user must specify a command.

4.1.8 File–copy pairs

File–copy pairs specify the list of files that the SCM will copy from the CMS to the managed nodes during tool execution. The SCM tool builder specifies file–copy pairs as a pairing of a source pathname on the CMS and a destination pathname on the managed node. The ServiceControl Manager does not allow more than one file–copy pair with the same destination in one tool. For the source and destination files, the user must specify an absolute file path starting at the root directory. The SCM user may specify no more than 16 file–copy pairs per tool definition.

When defining a tool definition, the SCM user must specify either a command or a file–copy pair or both (see Command).

4.1.9 Execution user

When the tool is executed on a managed node, the tool runs under the UID specified as the execution user. The execution user is optional. If the execution user is not specified, the tool runs under the UID of the current SCM user.

4.1.10 Roles

The role attribute provides the list of SCM roles for this tool. Whether or not the SCM enables these roles for this tool depends on the value of the owner attribute (see Owner). If the trusted user does not specify the owner attribute then the SCM enables all roles. If the trusted user specifies the owner attribute, then the SCM enables the “Master Role” role. For an SCM user to run the tool on a managed node, the SCM user must be authorized with one of the tool’s enabled roles on the managed node.

All tools contain the “Master Role” by default. The SCM user cannot remove the “Master Role” from a tool. Only an SCM user with the Trusted User privilege can modify a tool’s roles.

4.1.11 Default Targets

The default targets attribute specifies the possible managed nodes on which the tool may run if the SCM user does not specify a target managed node. The possible values are:

- None specified, i.e. no defaults,
- “ALL”, all managed nodes for which the operator is authorized,
- “CMS”, only run on the managed node specified as the CMS, or
- A single node on which the trusted user must have authorized the executing SCM user.

If SCM user specifies no default targets then she must specify the target nodes at execution time. If the default targets attribute is set to “ALL”, at execution time, the SCM will determine all of the authorized managed nodes on which the tool will run. If the SCM user specifies the default targets attribute as “CMS”, the SCM will run the tool only on the CMS. If the SCM user sets the default targets attribute to a specific node, the SCM will only run the tool on that specific node. For the CMS or a specific node, the SCM user must have an authorization to run the tool on that node. If an SCM user specifies a list of nodes when preparing to execute a tool, that list overrides the default targets attribute.

4.1.12 Log flag

The log flag indicates whether to save the stdout and stderr results from the execution of the tool to the SCM log.

By default, the results are saved to the SCM log. Regardless of the value of this flag, the SCM always logs the tool execution event..

4.1.13 Launch–only flag

The Launch–only flag indicates whether the agent should wait for the completion of the command and capture the exit code, stdout, and stderr results. If the Launch–only flag is set, the agent does not wait for the completion of the command.

By default, the agent waits for the task to complete, captures the results, and returns the results to the CMS.

4.2 Managing Tools

The ServiceControl Manager provides two interface categories for managing tools, a graphical user interface (GUI) and a command line interface (CLI). Both interfaces provide the same basic capabilities, to define, add, modify, remove, and execute tools. The SCM GUI provides a windowing look–and–feel for managing tools. The GUI provides windows for defining tools and menu items for adding, modifying, removing, and executing tools. The GUI is more interactive and dynamic than the CLI. The CLI provides access to tools at a terminal prompt and allows tool management in a shell script.

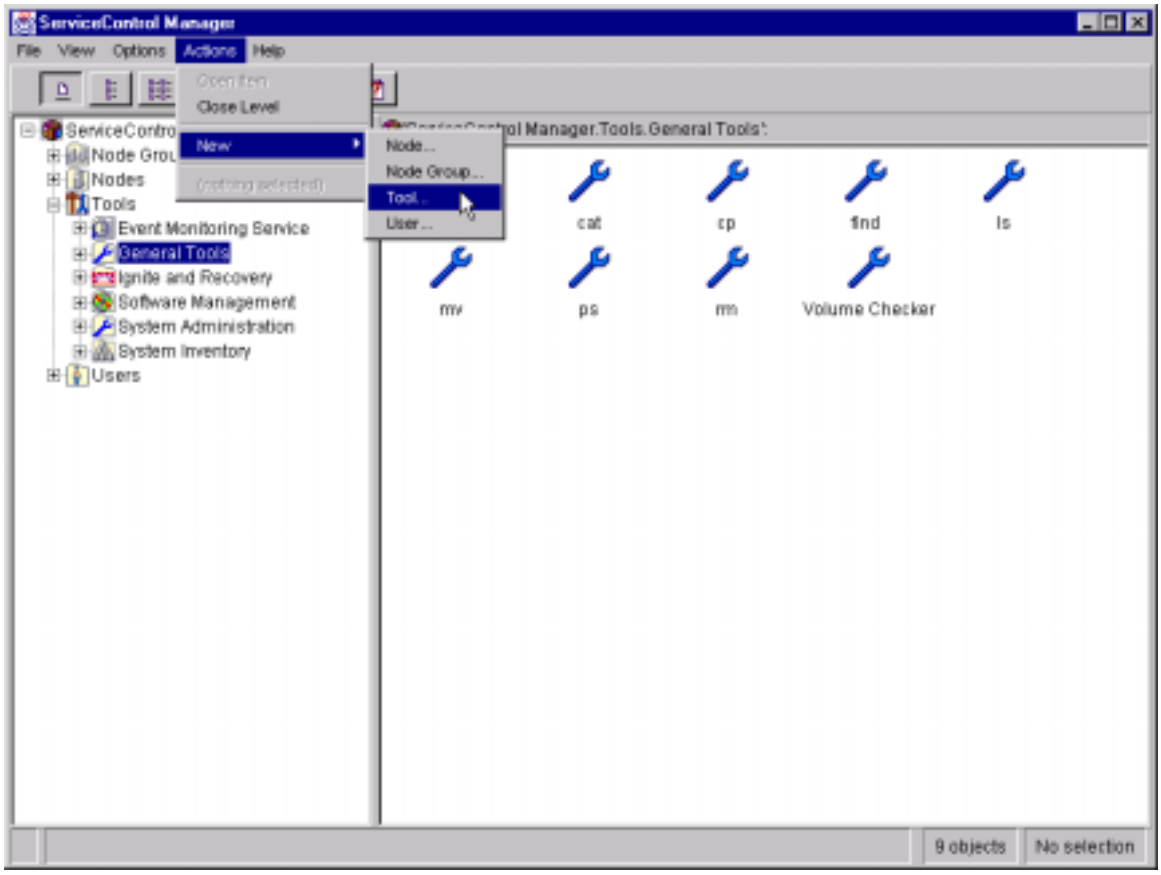
4.2.1 Defining Tools

The SCM restricts who may specify certain attributes of a tool, such as the owner or tool roles. The SCM only allows a SCM user with the Trusted User privilege to specify the owner of a tool or remove the owner field from a tool. If a non-privileged SCM user is defining a tool, the SCM will automatically set the tool owner to the SCM user's id. A non-privileged SCM user may only specify the SCM "Master Role" in a tool. The "Master Role" is the default role for all SCM tools, so assigning this role has no real effect on the tool definition. The SCM restricts assignment of these fields because the values of these fields affect the execution of the tools on the managed nodes. If the owner field is set, then the SCM only enables the "Master Role" in the tool. Therefore, the SCM user can only execute the tool on managed nodes for which the SCM has authorized the SCM user with the "Master Role" role. If the owner field is not set, the SCM enables all roles assigned to a tool. If SCM user has authorizations matching those in the tool to the appropriate managed nodes, the SCM user may execute the tool on those nodes.

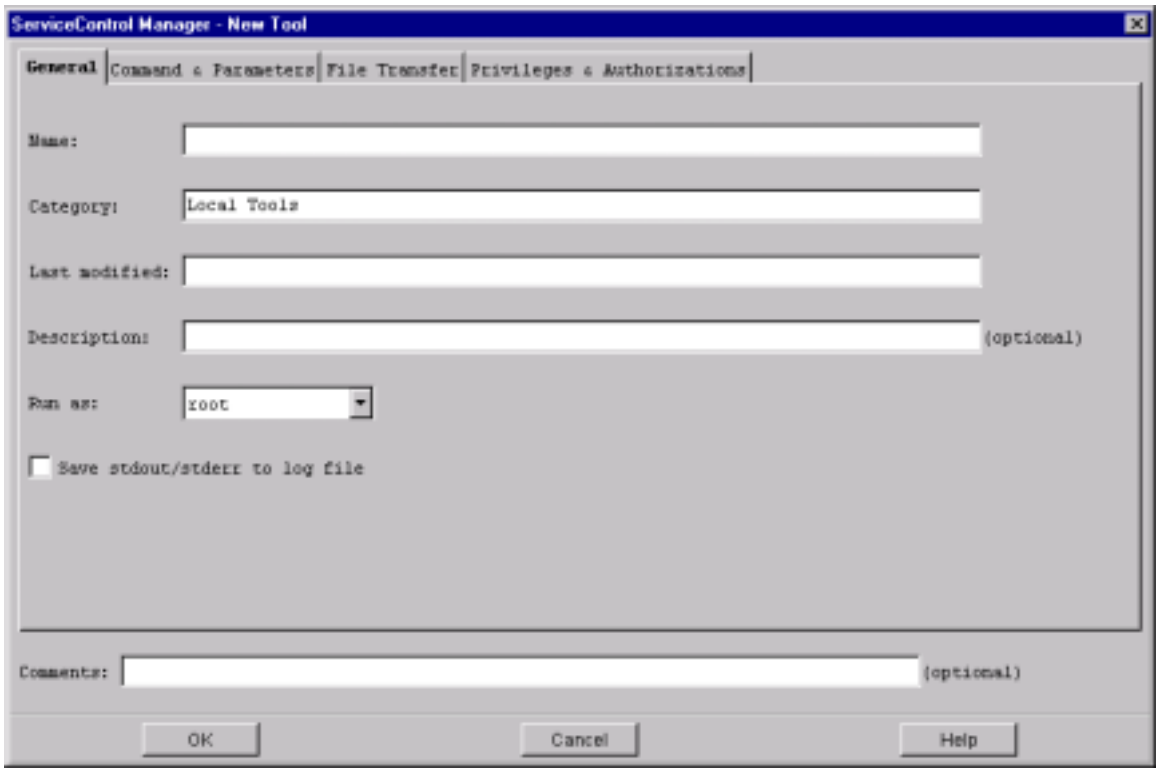
4.2.1.1 Graphical User Interface

To define tools, the SCM GUI provides a “New Tool” window consisting of tabbed property pages for general tool information, command and parameters, file transfer information, and owner and roles.

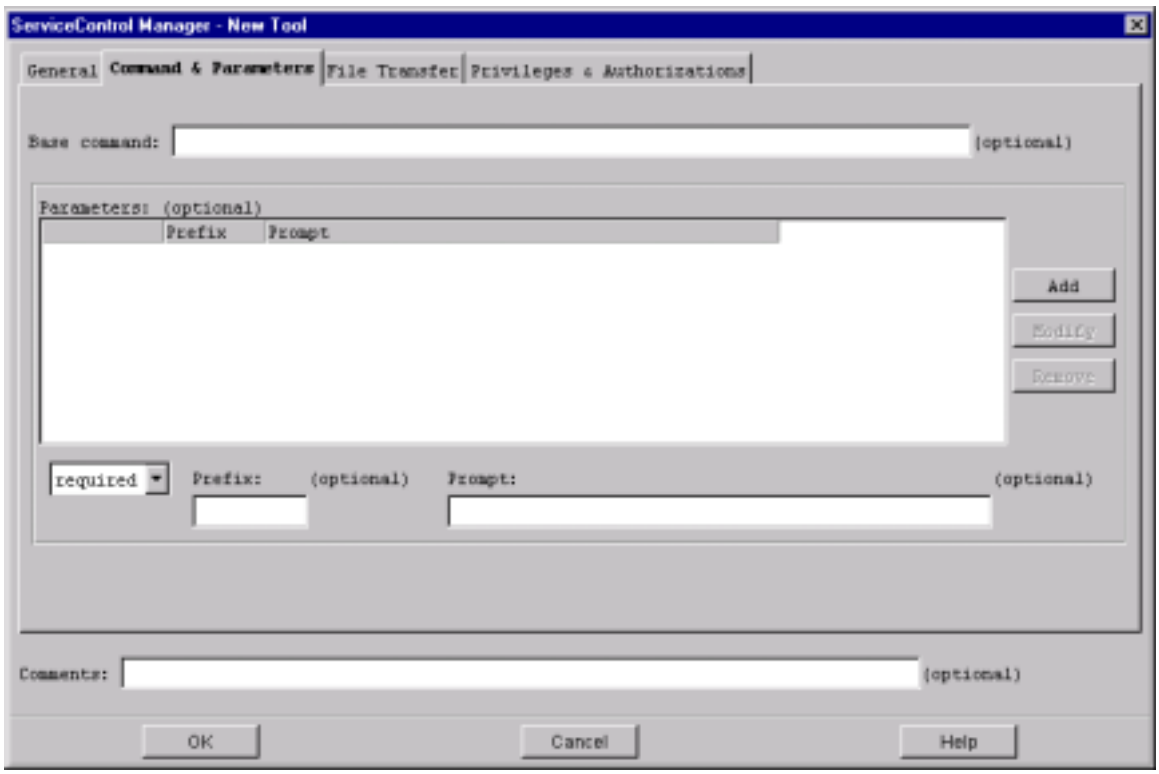
To display the “New Tool” window, the SCM user clicks on the “Action” item on the main window, selects the “New…” item, and the “Tool…” item.



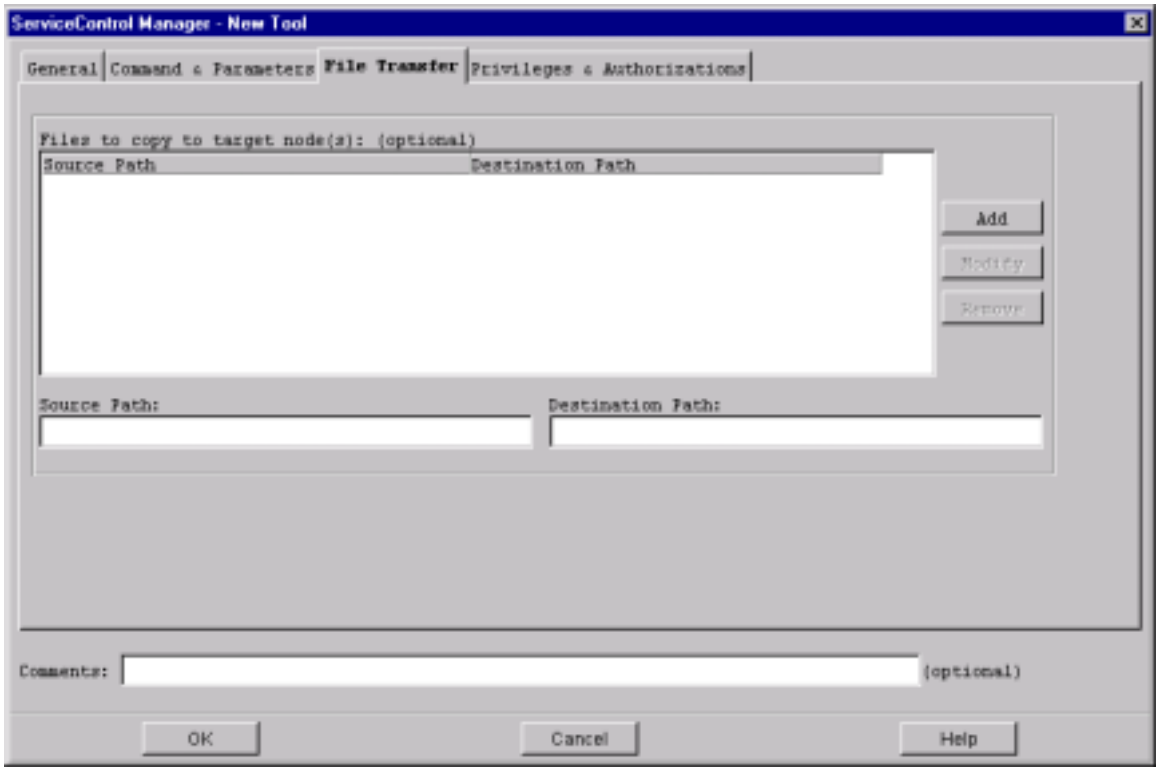
The “New Tool” window consists of several tabbed property pages for defining the tool attributes The “General” page provides entry fields for the tool name, category, description, and comments. Additionally, there is a list box for the “run as” user and a check box to indicate whether to save the tool output to a file. If the SCM user selects to run the tool as a specific user, the GUI will display an entry field in which to provide the specify user value. The GUI does not allow specification of the “Last modified” field.



The “Command & Parameter page provides entry fields to define the command line and the command parameter information. The parameters consist of a prefix, a prompt, and a flag to indicate if the parameter is required or optional. The SCM does not automatically put spaces between the commands and the parameters. If the SCM user desires spaces, the SCM user must either specify the command with a trailing space or specify spaces in the prefix field.

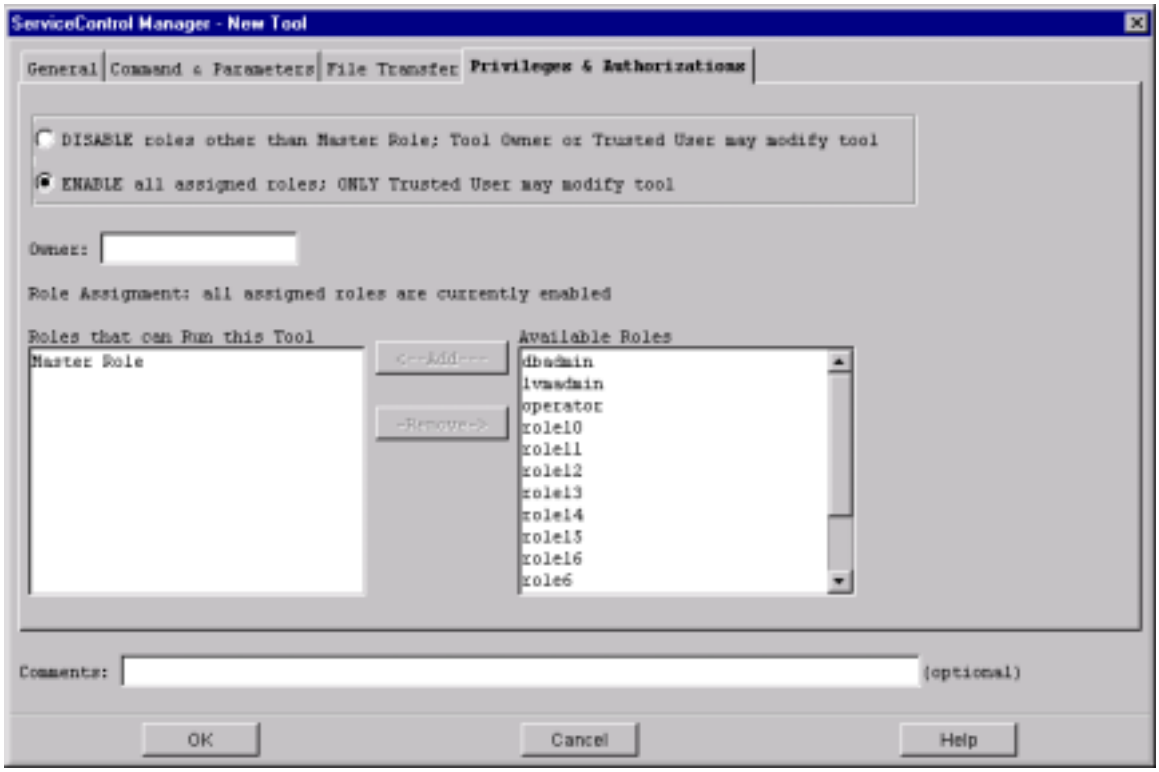


The “File Transfer” page provides fields to define the file-copy pairs for the tool.



The “Privileges & Authorizations” page provides the fields necessary to define the owner of the tool and the tool’s roles.

Note that specifying a name in the “Owner” field causes the GUI to set the “DISABLE” radio button on the page. Clicking the “ENABLE” radio button causes the GUI to blank out the value in the “Owner” field. The GUI does not allow non-privileged SCM users to modify these fields.


ServiceControl Manager - New Tool
General
Command & Parameters
File Transfer
Privileges & Authorizations
DISABLE roles other than Master Role; Tool Owner or Trusted User may modify tool
ENABLE all assigned roles; ONLY Trusted User may modify tool
Owner:
Role Assignment: all assigned roles are currently enabled
Roles that can Run this Tool
Master Role
<--Add---
-Remove->
Available Roles
dbadmin
lvmadmin
operator
role10
role11
role12
role13
role14
role15
role16
role6
Comments:
(optional)
OK
Cancel
Help

4.2.1.2 Tool Definition File

SCM provides a mechanism for translating a tool definition file into an SCM tool via a command line interface. The tool definition file may contain one or more tool definitions. The SCM tool definition grammar description uses the EBNF syntax, where "|" means a choice, "?" means optional, and "*" means zero or more. The tool definition grammar is as follows:

```
file           := toolDefinition*

toolDefinition := "SSA" "tool" name "{" toolStmts "}"

toolStmts      := toolGlobals* copyStmt execStmt defTargets? roleList?
                | toolGlobals* execStmt defTargets? roleList?
                | toolGlobals* copyStmt defTargets? roleList?

toolGlobals    := "description" string
                | "comment" string ( “,” string)*
                | "revision" string
                | "owner" name
                | "category" name

copyStmt       := "copy" "{" fileCopySpec ( "," fileCopySpec )* "}"

fileCopySpec   := filePath ":" filePath

execStmt       := "execute" "{" commandSpec toolOpts* "}"

commandSpec    := "command" cmdString ( argStmt )*

argStmt        := "arguments" "{" promptStmt ( "," promptStmt )* "}"

promptStmt     := cmdString
                | ":" string ( "optional" )?
                | cmdString “:” string ( “optional” )?

toolOpts       := ( launchOnly | logOutput | executeUser )

launchOnly     := ( "launch" | "nolaunch" )

logOutput      := ( "log" | "nolog" )

executeUser    := "user" name

defTargets     := "default" "targets" ( "CMS" | "all" | name )

roleList       := "roles" "{" name ( "," name )* "}"
```

There are four special grammar definitions used that are not defined above. They require special treatment to properly describe.

string: must be enclosed in double quotes and begin with an alphabetic character (upper or lower case)

cmdString: must be enclosed in double quotes. Also, this type of string can be made up of a number of white space separated quoted substrings that will be later concatenated into a single long string. This allows arbitrarily long strings to be represented unlimited by the line length of the tool definition file.

filePath: must begin with "/", contain no white space, and represent a legal HP-UX file path name

name: this may either be a string as defined above or an unquoted single identifier containing no white space, starting with an upper or lower case alphabetic character,

and containing only alphabetic characters (upper or lower case), numerals (0-9), the underscore(_), the dash (-), or the period (.) characters.

When an SCM user sets logOutput to “log”, the SCM logs output from the execution of the tool to the CMS log file /var/opt/mx/logs/scmgr.log.

Example tool definition:

```
SSA tool "Volume Checker" {
    description "Check the volume usage of nodes"
    category "General Tools"
    copy {
        /home/dsuit/bdfscript : /tmp/bdfscript
    }
    execute {
        command "/tmp/bdfscript"
        arguments {
            " " : "Enter a bdf option:" optional
        }
        nolaunch
        user root
    }
    default targets all
    roles {
        "Master Role",
        "operator"
    }
}
```

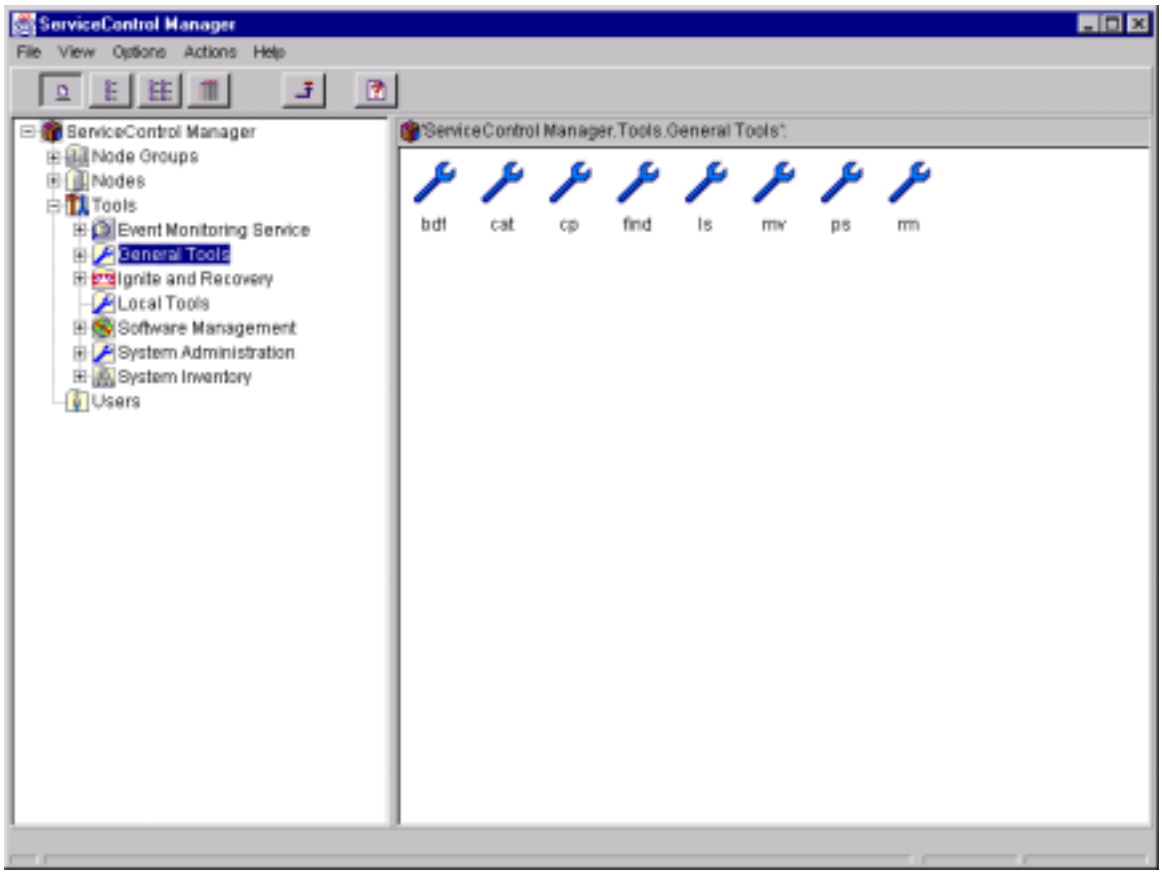
The above definition defines a tool named “Volume Checker”. The SCM puts this tool in the “General Tools” category. The tool copies the file “/home/dsuit/bdfscript” from the CMS to “/tmp/bdfscript” on the target nodes. After copying the file, the tool executes the bdfscript. Optionally, the SCM user may specify a bdf option for the command. When the SCM executes this tool, it waits until the tool completes on all nodes and provides the output directly to the SCM user. The SCM writes the tool execution output to the SCM log. The SCM executes the tool as the “root” user on the target nodes. If the SCM user does not provide a list of target nodes, the tool will run on all user authorized nodes. To run the tool, the SCM user must posses the operator or “Master Role” role on each of the target SCM nodes.

4.2.2 Adding tools

Once the SCM user has defined a tool, the SCM user may add the tool to the SCM repository. Assuming the SCM user defines the tool properly and the tool does not currently exist in the SCM repository, the SCM adds the tool to the SCM repository.

4.2.2.1 Graphical User Interface

To add a tool from the GUI, the SCM user clicks on the main menu “Actions” item and selects “Add” and “Tool” from the “Add” cascade menu to bring up the “New Tool” property window. The SCM user defines the tool by filling in the appropriate fields. When finished, the SCM user clicks the “OK” button to add the tool to the SCM repository. The GUI displays the tool in the category defined by the tool’s category attribute.



4.2.2.2 Command Line Interface

To add tools from the command line interface, the SCM user must specify the add option and provide a tool definition file as a parameter to an mxtool command. For example to add tools from the “mytools” tool definition file, the SCM user would enter the following command line:

```
mxtool –a –f mytools
```

When the SCM user enters the mxtool command, the SCM parses the tool definition file and adds the resulting tools to the SCM tool repository. If SCM encounters any errors during tool definition parsing, the SCM will report the errors and will not add any tools to its repository. Before adding the parsed tool to its repository, the SCM will check if a tool by the same name exists. If a tool with the same name exists, the SCM will not add the parsed tool, but the SCM will continue to try to add the remaining parsed tools.

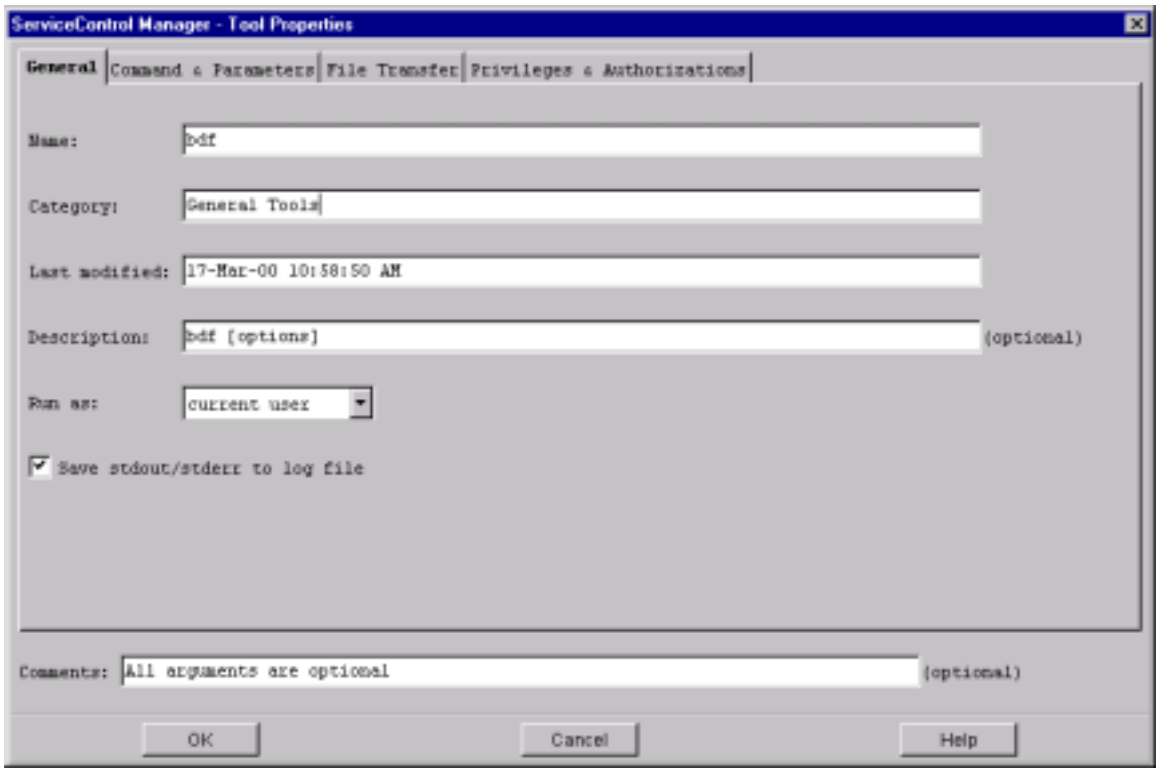
If the current SCM user is not a privileged SCM user, the SCM will specify the current SCM user as the tool owner and will only specify the “Master Role” in the tool’s roles list. If the current SCM user is a privileged SCM user, the SCM will use the owner ID and the role list specified in the tool definition to define the tool.

4.2.3 Modifying tools

An SCM user may modify existing tools in the SCM repository under the proper circumstances. SCM users with the Trusted User privilege may modify any tool in the SCM repository. Other SCM users may only modify tools for which they are the owner. For a non-privileged user to modify an SCM tool, a privileged SCM user must set the tool's owner field to the non-privileged SCM user. This action will disable the roles of the tool, except for the "Master Role", and allow the non-privileged SCM user to modify the attributes of the tool, with the exception of the owner field and the role list.

4.2.3.1 Graphical User Interface

To modify a tool, the SCM user navigates to the tool category window that displays the icon for the appropriate tool. The SCM user can right-click on the icon and select the "Properties" item to bring up the "Tool Properties" window. After modifying the appropriate fields, the SCM user can click the "OK" button to apply the changes to the SCM tool repository.



4.2.3.2 Command Line Interface

To modify tools from the command line interface, the SCM user must specify the modify option and provide a tool definition file as a parameter to an mxtool command. For example to modify tools from the mytools tool definition file, the SCM user would enter the following command line:

```
mxtool –m –f mytools
```

When the SCM user enters the mxtool command, the SCM parses the tool definition file and replaces the resulting tools in the SCM tool repository. If SCM encounters any errors during tool definition parsing, the SCM will report the errors and will not modify any tools to its repository. Before the SCM modifies the tool based on the parsed tool, the SCM must read the existing tool with the same name from its repository. If the SCM cannot find a tool by the same name, the SCM will skip the parsed tool, but the SCM will try to modify the remaining tools that were parsed from the given tool definition file. Note that tool names are case insensitive.

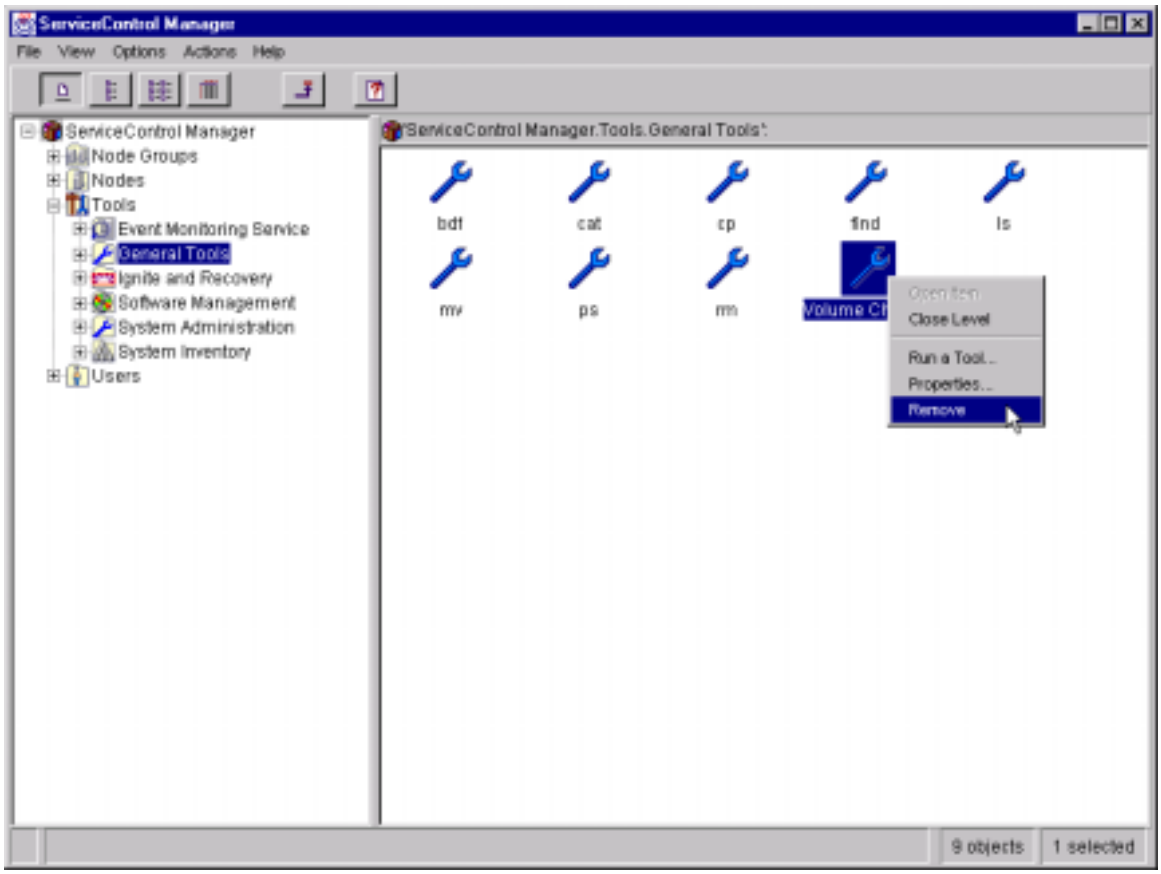
If the current user is not a privileged SCM user, the SCM will only allow the SCM user to modify the tools for which the current user is the owner. The SCM will ignore any changes made by a non-privileged SCM user to a tool's roles list.

4.2.4 Removing tools

Only privileged SCM users may remove tools from the SCM repository.

4.2.4.1 Graphical User Interface

To remove a tool, the privileged SCM user navigates to the tool category window displaying the appropriate tool, selects the icon for the appropriate tool, and, from the “Actions” menu, selects “Remove”. Optionally, after selecting the tool’s icon, the user can press the right mouse button to display a pop up menu with the “Remove” option.



4.2.4.2 Command Line Interface

To remove tools using the command line interface, the privileged SCM user must specify the remove option and provide a list of one or more tool names as a parameter to an mxtool command. For example to remove the tool named “Sample Tool” from the SCM tool repository, the privileged SCM user would enter the following command line:

```
mxtool –r –t “Sample Tool”
```

The privileged SCM user may enter one or more tool names on the command line. The SCM will attempt to remove the tools provided in the command line. If the SCM does not find a specified tool, the SCM will proceed to attempt to remove the next specified tool.

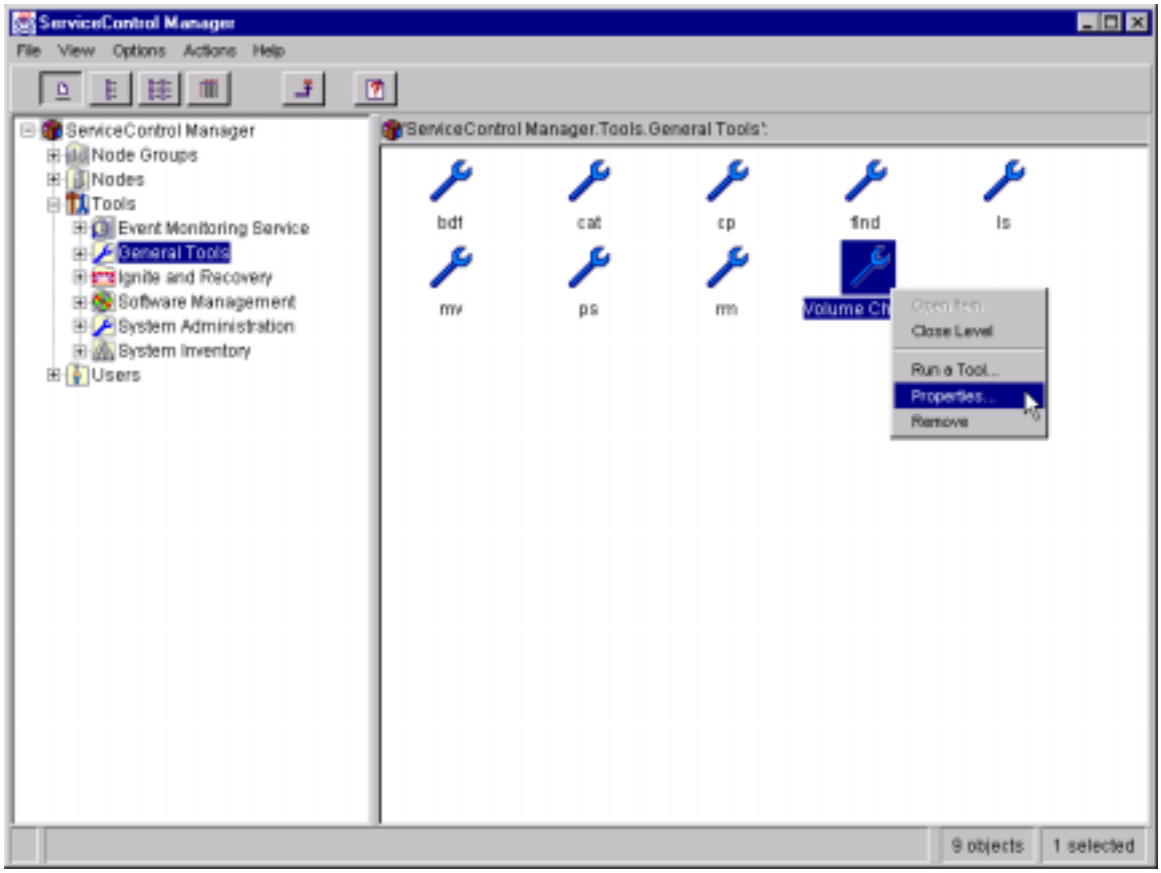
Note that tool categories only exist if the SCM user defines tools within the category; therefore, removing all tools from a category effectively removes the category as well.

4.2.5 Listing tool information

The SCM provides mechanisms for displaying the tools, the tool categories, and the tool definitions to the SCM users. Any SCM user may examine tool information.

4.2.5.1 Graphical User Interface

The GUI provides hierarchical displays of tool categories through which the SCM user can navigate to see lists of tools within the category. The SCM user can select a tool's icon, right-click the tool's icon, and select Properties from the popup menu to display the attributes of a tool (see Modifying Tools).



4.2.5.2 Command Line Interface

To display tool information using the command line interface, the SCM user must specify the list option, a “list option format argument”, and may optionally provide either a tool category or a list of one or more tool names as a parameter to an mxtool command. If the SCM user does not provide a category or list of tool names, the SCM will list the entire SCM tool repository in the specified format.

The mxtool CLI provides four list option arguments to control the listing format of a tool:

- n – lists just the tool names, equivalent to specifying the mxtool command without options
- t – lists the tool category, tool names, and description in a row, column format
- d – lists all of the tool’s attribute values in a single column, screen-viewable format
- f – lists the tools in a tool definition file format

For example, the following command lists the tool, “Sample Tool”, in a tool definition format and redirects the output to a file named “mySampleTool”:

```
mxtool –lf –t “Sample Tool” > mySampleTool
```

The SCM user can edit the “mySampleTool” file and use the file as input to the mxtool modify option.

4.3 Executing Tools

Executing tools across a set of managed nodes is the central feature of the ServiceControl Manager. The following sections describe the mechanisms for converting a tool into an executable task and distributing the task to the managed nodes for execution.

4.3.1 Generating a Task

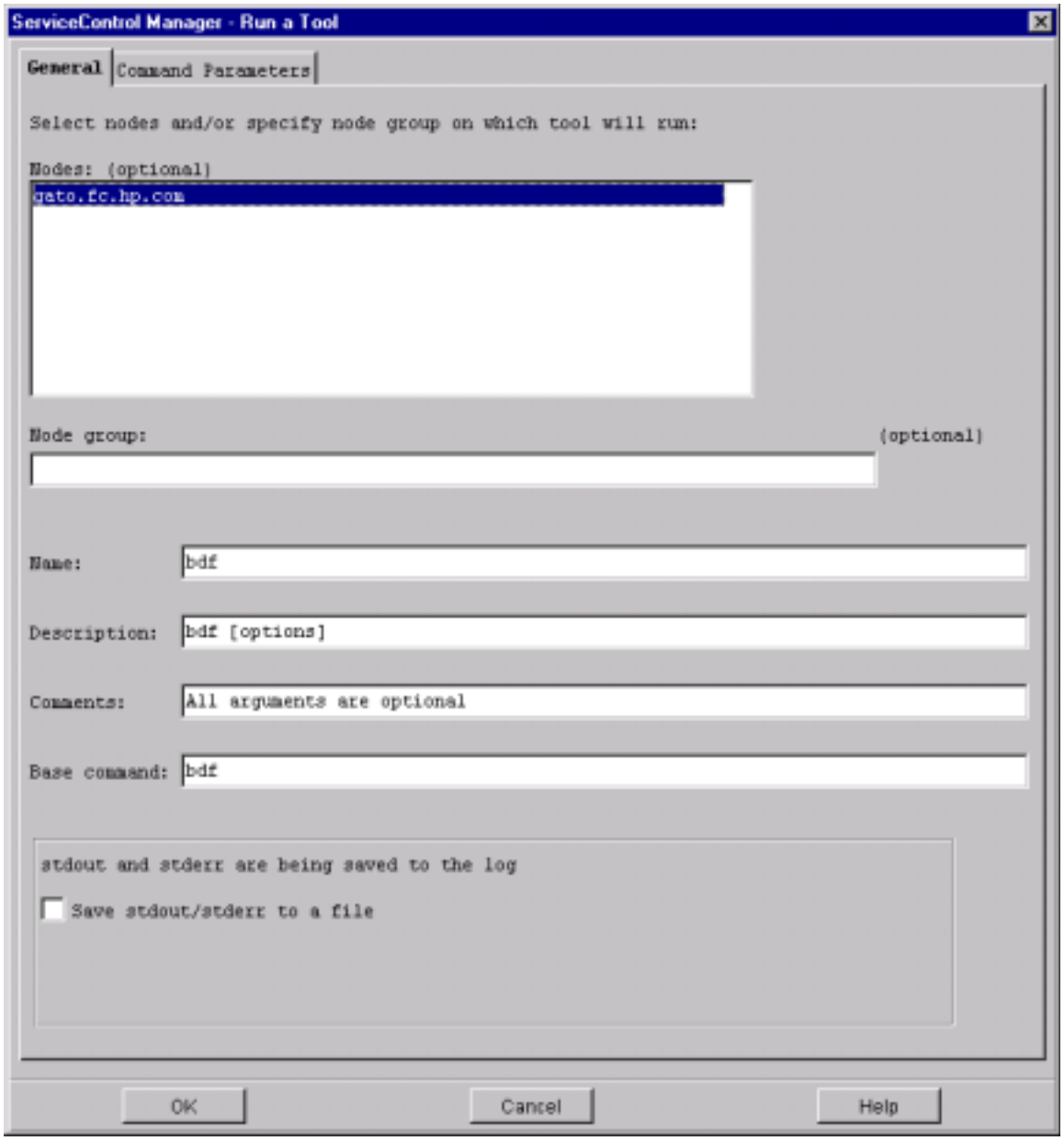
The SCM provides mechanisms to execute tools from the GUI or the CLI. To execute a tool, the SCM user must provide the tool name. If the tool does not specify a default target in the tool definition, the SCM user must specify the managed nodes or node groups on which the tool may run. Additionally, if the tool definition requires additional parameter values to complete the command line, the SCM user must provide those parameter values. The SCM will try to construct a command line from the tool definition and the user-provided data. The SCM takes the tool's command attribute and iterates through each of the parameters in the tool's parameter list. While iterating it concatenates any defined parameters prefix and any provided parameter value.

SCM users can define SCM tool parameters with a prefix, a prompt, or both. If the SCM user defines only a prefix for the parameter, the SCM user must also define the parameter as required. For parameters that are required, the SCM will generate an error if the tool definition contains a prompt, but the SCM user does not provide a value. For parameters that are optional, the SCM only concatenates the parameter with its optional prefix if the SCM user provides a value. For this reason, the SCM will flag as an error any optional parameters that have no prompt defined.

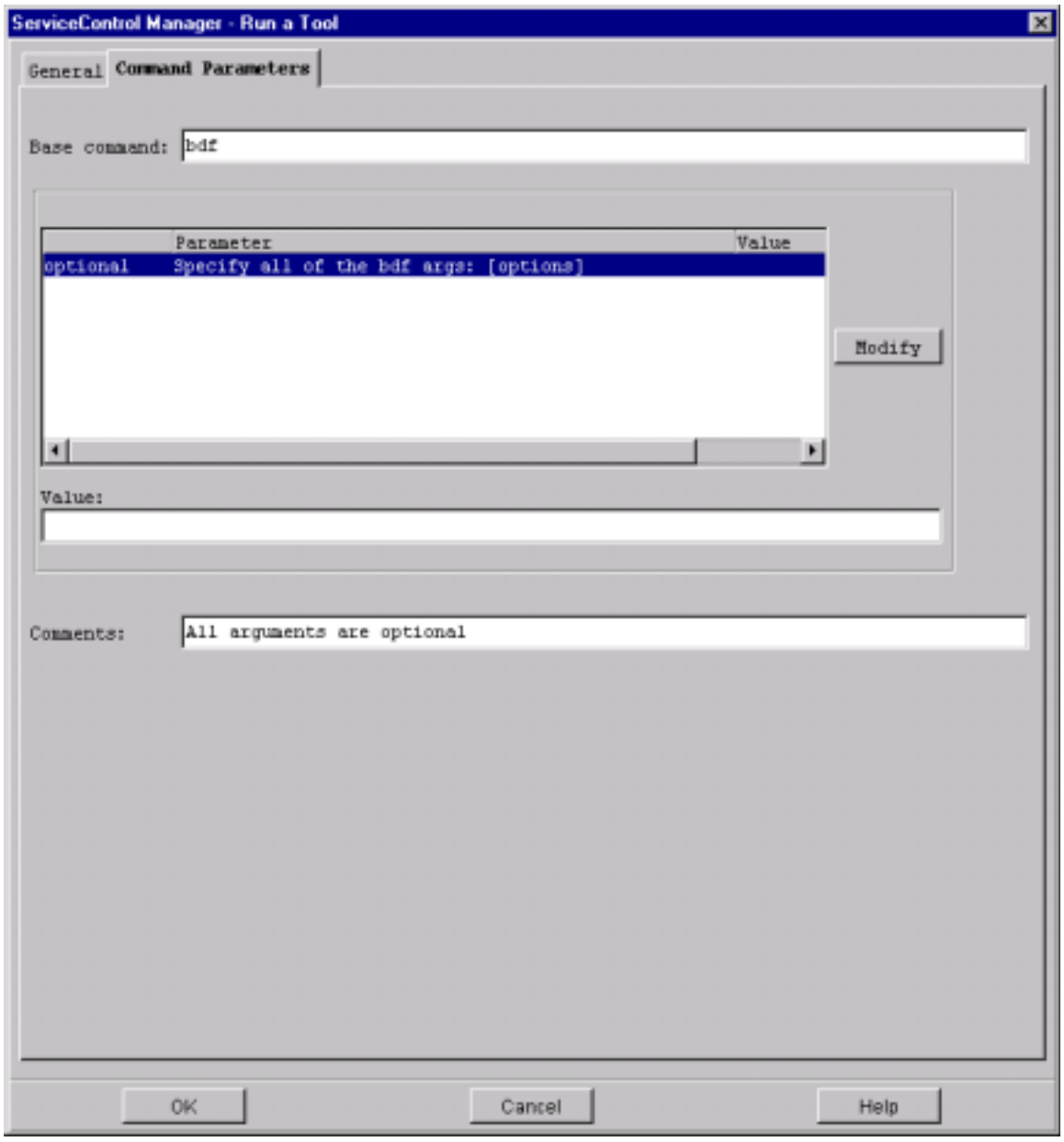
Once the SCM has concatenated the command and parameters into the complete command line, the SCM will determine the managed nodes on which the tool will run. First, the SCM will generate a list of unique managed nodes based on the nodes and node groups specified by the SCM user. If the SCM user is not authorized for all of the specified managed nodes, the SCM will not run the tool.

4.3.1.1 Run a Tool GUI option

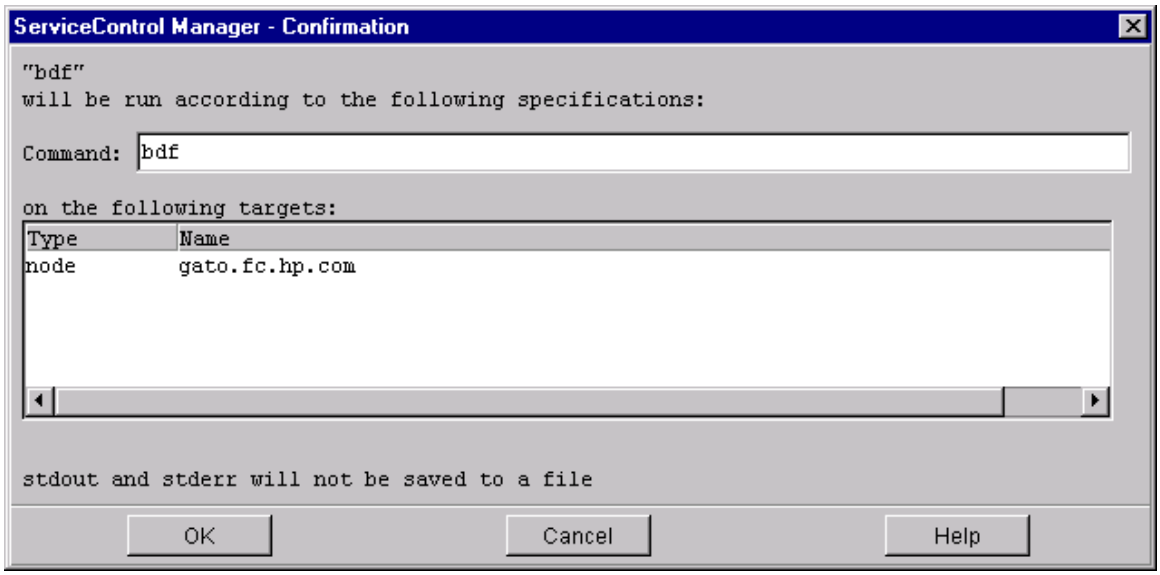
To execute a tool from the GUI, the SCM user must navigate to the tool category window that displays the icon of the tool to execute. The SCM user may then either click the tool icon and select the “Run a tool” item from the “Actions” menu or double-click the tool icon. Either action displays the following window:



The SCM user can then select the node(s) or specify the node group(s) from which to generate the list of target nodes and select whether to save the output to a file. If the SCM user wants to provide arguments to the command, the SCM user selects the “Command Parameters” page to display the following window:



When the SCM user completes specifying the command, the user clicks the “OK” button. SCM then displays a window to confirm the tool execution:



To run the tool, the SCM user clicks the “OK” button.

If the tool definition specifies a default target and the tool does not require the SCM user to specify any parameters, the GUI will skip the preceding steps, execute the tool immediately, and display the task results.

4.3.1.2 mxexec CLI command

To execute a tool from the command line, the SCM provides the mxexec command. When entering the mxexec command, the SCM user provide a tool name for example:

```
mxexec –t “My Tool Name”
```

If the tool’s parameters require argument values or the SCM user wishes to provide optional argument values, the SCM user must provide the desired values, for example:

```
mxexec –t “My Tool Name” –A value1 “value 2” “value 3”
```

If the SCM user wants to skip some optional arguments but specify others, the SCM user must insert empty strings as placeholders for the skipped arguments, for example:

```
mxexec –t “My Tool Name” –A value1 “” “value 3”
```

If the tool does not specify default targets or the SCM user wishes to specify the target nodes, the SCM user must enter the nodes and node groups on which the tool will run. The following example shows an mxexec command for which the user only specifies nodes:

```
mxexec –t “My Tool Name” –n target1 target2 target3
```

The next example shows an mxexec command for which the user specifies nodes and node groups:

```
mxexec –t “My Tool Name” –n target1 target2 g:group1
```

The SCM user may specify that the output from the tool execution goes to a single file, for example:

```
mxexec –t “My Tool Name” –O outputfilepath
```

The SCM user may specify that the output from the tool execution go to a directory with a separate output file for each target node, for example:

```
mxexec –t “My Tool Name” –o outputDirectoryPath
```

4.3.2 Determining the authorized nodes

Before executing a tool, the SCM determines the SCM user’s authorizations on the managed nodes. The SCM user may specify the nodes and node groups on which the SCM will run the tool or may defer to the nodes defined by the tool’s default targets attribute. When the SCM user provides the list of nodes, that list always overrides the tool’s default targets.

When the SCM user does not specify a list of target nodes, the SCM will determine the target nodes from the tool’s default target attribute. If the tool’s default target is “CMS”, the SCM will run the tool on the managed node that the SCM designates as the CMS server. If the default target is set to a managed node, the SCM will run the tool on the specified managed node. If the tool’s default target attribute is set to “ALL” and the SCM user does not provide a list of target nodes, the SCM will automatically generate the list of authorized nodes by intersecting the tool’s roles list and the SCM user’s authorizations. If the tool definition does not specify a default target, the SCM user must provide the target nodes on which the tool must run.

Once the SCM determines the target nodes, the SCM will check if the tool’s roles and the SCM user’s SCM authorizations intersect for the provided target nodes. The SCM user’s authorization consists of the user’s name, a role, and a managed node or node group. The SCM user may have many authorizations to get the proper set of roles associated with the appropriate nodes. The SCM will only allow the SCM user to run a tool for which the SCM user has one of the tool’s roles authorized for the target node.

4.3.3 Running the executable command

When an SCM user runs a tool, the result is an SCM task. The SCM Distributed Task Facility (DTF) manages SCM tasks. The DTF consists of two processes, a DTF daemon, which runs on the CMS, and DTF agents, which run on the managed nodes. A task represents the invocation of a specific tool on a specific set of target nodes at a specific time. Therefore, a task has a lifecycle and a lifetime. The lifecycle of a task is the sequence of states that a task steps through from definition to completion. The lifetime of a task is the time duration from when the DTF starts a task to when it completes. After defining a task and before handing off the task to the DTF agents on the managed nodes, the DTF daemon assigns a task identifier for each task. The SCM user can use this identifier to track the task and to look up information later about the task in the SCM log. The DTF manages 20 tasks at a time on a circular queue.

For every target node, a task goes through a set of states that track the progress of the task on each node. The states are:

- Pending,
- Contacting target,
- Copying files,
- Running tool, and
- Complete.

The following table describes each of the task states:

State	Description
Pending	Nothing regarding the specific target is happening. This state is used when there are a very large number of target-managed nodes and the DTF daemon is only able to run a task in parallel on a smaller number of nodes. The DTF daemon can run 20 managed nodes in parallel. Those target managed nodes that are waiting to start are in the Pending state.
Contacting target	During this state, the DTF daemon makes contact with the DTF agent on the target node, establishes a connection, and passes the task information to the DTF agent.
Copying files	If there are any files to copy, the task enters this state, and the DTF daemon transmits the contents of the files to the target managed node. Before attempting to copy the files, the DTF will try to open all of the files in the task. If any file cannot be open, i.e. cannot be read, the DTF will fail the task immediately. The target's DTF agent writes the files and sets the files' ownership and permissions.
Running tool	If there is a command line to execute (the command line is optional for a tool copying files), the DTF daemon assigns the Running tool state to the task. During this state, the target's DTF agent forks a process to run the command, establishes a clean process environment, and executes the POSIX shell with the command line as the argument. The DTF agent sets the stdin for the process to /dev/null. If the tool is a launch-only tool, as soon as the POSIX shell has successfully executed the command line, the target's DTF agent ignores the child process and moves to the next state. If not, the target's DTF agent waits while the command executes and after it exits, it gathers up the stdout, stderr, and exit code of the process, returns the results to the DTF daemon on the CMS, and closes the connection.
Complete	The task has completed, and the DTF daemon on the CMS makes the resultant information available to the user interface and logs the information to the SCM log. The information about a running task is stored dynamically by the DTF daemon. The DTF daemon remains in contact with the nodes on which tools are running and provides status information back to the user interface. As long as a task is running and for a short time after it completes, the DTF daemon retains all the state information about the task, including all of the output and exit codes from each target node. The information about a task is kept in the SCM after the task completes as long as there is a user interface process that has the task open (via the View Task Details screen, for example). Because the logging of tool output is optional (controlled by the tool definition), the log may or may not contain the tool output (stdout and stderr). But the exit code from each target node, if the command was successfully executed, and the failure information, if it was not, is always logged. The user can view the information stored in the log at any time in the future until the log information expires, i.e. for the last 20 tasks executed.

When a task is executing on a node, the agent will set the following environment variables in the environment in which the tool command runs:

- MX_USER - This contains the UID of the SCM user who ran the tool.
- MX_TASKID - This is the SCM assigned task id.
- MX_TOOL - This is the SCM tool name (may not be the same as the tool script name).
- MX_TARGETS - This contains a space-separated list of target nodes for multi-system aware applications. This environment variable is empty for applications that are single system-system aware. The SCM lexicographical sorts the target nodes.
- MX_CMS - This is the SCM CMS hostname.
- MX_REPOSITORY - This is the hostname of the system containing the NDS repository.

In addition to these special environment variables, the agent also sets some other standard variables:

- DISPLAY - This is set depending upon the way in which the SCM user invoked the SCM. If the mxexec CLI invokes the tool, the value of DISPLAY in the current environment is used. If the GUI was invoked in a browser and used to run the tool, the SCM creates an X server that provides an X window onto which X applications, including Java windowing applications, can map their windows.
- HOME - This is set depending upon the UID used to invoke the command. The DTF looks up the home directory value from the user registry (/etc/passwd or NIS) and is used to set HOME.

4.3.4 Canceling and killing tool execution

The SCM provides two mechanisms to interrupt task execution, canceling or killing the task. While the task is in a pending or copying files state, the SCM user may cancel the task. Canceling the task interrupts file copying and prevents the DTF daemon from transitioning the task to the running state. Once the task enters the running state, the SCM user is limited to killing the task. When the SCM user requests to kill a task, the DTF daemon sends messages to all daemons involved in the task to kill the processes related to the task. Naturally, if a DTF agent receives the kill command near the completion of the task's process, the process may complete before the agent can kill the task.

To cancel or kill a task, use the CLI mxexec command with the cancel or cancel with kill option. For example, to kill task number 392, use the following command:

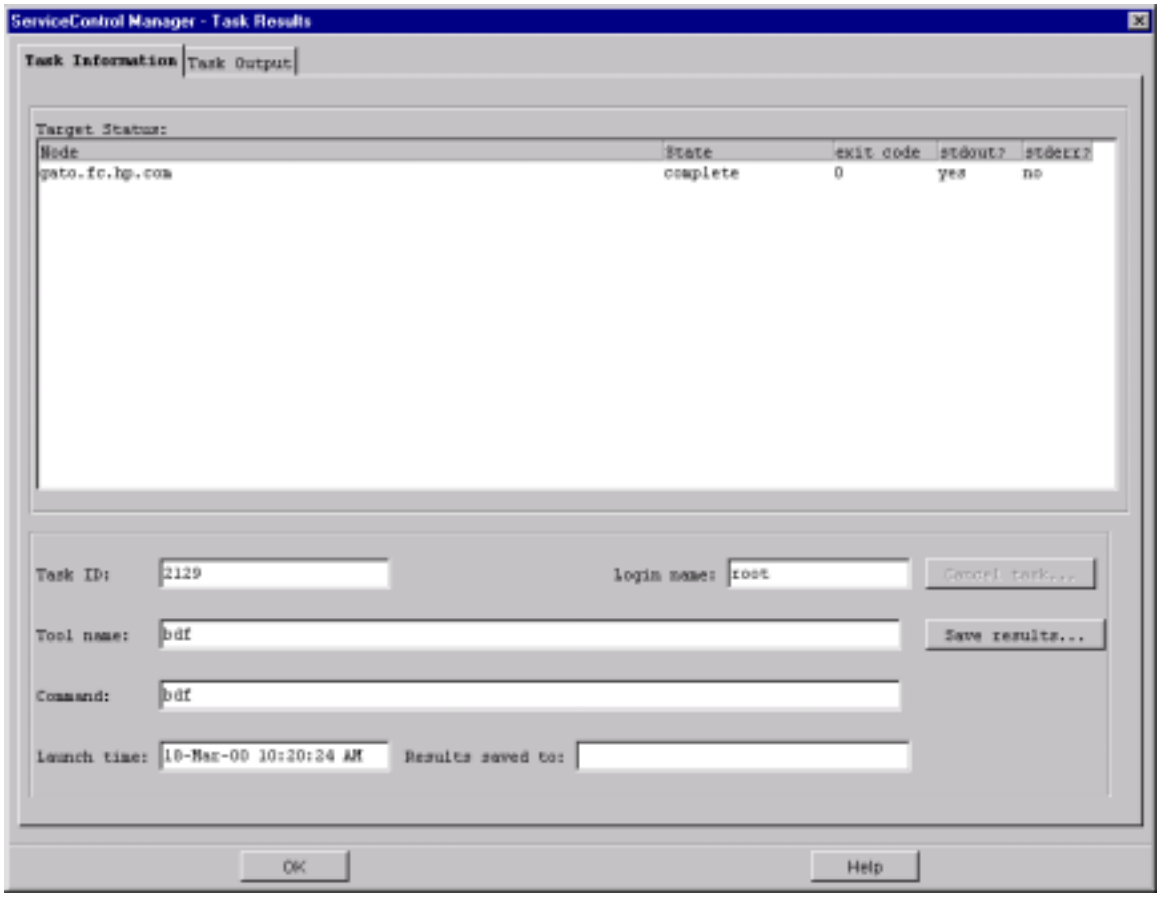
```
mxexec –ck –i 392
```

4.3.5 Examining the results

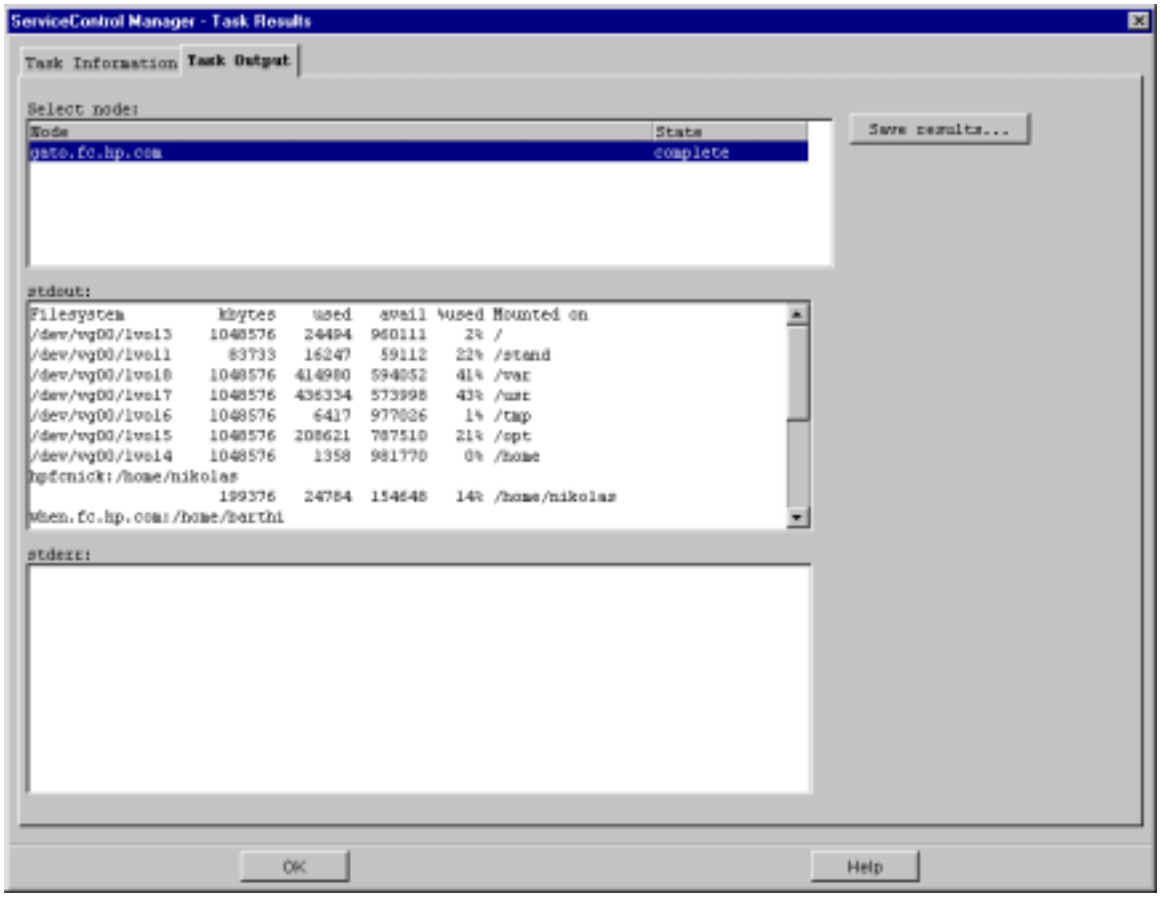
There are three possible mechanisms to view the results of tool execution.

If the tool was not a launch-only tool, the DTF daemon will send the tool's task output directly to the SCM user. If the GUI executes the tool the output goes to an output window.

The following figure shows the “Task Information” page of the “Tasks Results” window:



The next figure shows the “Task Output” page of the “Task Results” window:



If the tool is run from the CLI, the output goes to standard output and error and looks like the following example:

```
Task ID      : 392
Tool Name    : bdf
Task State   : Complete
User Name    : root
Start Time   : Friday, March 17, 2000 3:21:54 PM MST
End Time     : Friday, March 17, 2000 3:21:54 PM MST
Elapsed Time : 432 milliseconds
Node         : gato.fc.hp.com
Status       : Complete
Exit Code    : 0
STDOUT       :
Filesystem          kbytes    used   avail %used Mounted on
/dev/vg00/lvol3    1048576   24494  960111    2% /
/dev/vg00/lvol1      83733   16247   59112   22% /stand
/dev/vg00/lvol8    1048576  408521  600116   41% /var
/dev/vg00/lvol7    1048576  436334  573998   43% /usr
/dev/vg00/lvol6    1048576    2315  980873    0% /tmp
/dev/vg00/lvol5    1048576  208621  787510   21% /opt
/dev/vg00/lvol4    1048576    1358  981770    0% /home
```

If the DTF daemon is still managing the tool's task, i.e. the task was one of the last 20 executed on the CMS; the task output is available by providing the task's id to the CLI's mxexec command, for example, to display the output for task 392:

```
mxexec –ld –i 392
```

If the tool definition set the tool's log flag, the DTF daemon will write the tool's task output to the SCM log, and the SCM user may view the SCM log data until the SCM log expires, i.e. is overwritten.

5 Summary

The ServiceControl Manager tool feature provides a rich set of built-in tools to support the management of multiple systems. Additionally, the SCM provides an extensible framework to incorporate legacy tools from the customer's domain.

Associating SCM roles with tools and using the SCM, role-based authorization mechanism provides a secure approach to executing tools across an SCM managed cluster.

Because the user executes the tool under the user's own, login account, as opposed to root, the SCM can keep an accurate account of administrative activities on the SCM managed cluster.