

The Architecture of Sun[™] Management Center

Technical White Paper



© 2000 Sun Microsystems, Inc. All rights reserved.

Printed in the United States of America.
901 San Antonio Road, Palo Alto, California 94303 U.S.A.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

TRADEMARKS

Sun, Sun Microsystems, the Sun logo, Sun Management Center, Solaris, Java, Sun Enterprise, JavaBeans, and JDBC are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.



Please
Recycle



Adobe PostScript

Contents

1. Introduction	1
The Systems Management Challenge	1
Simplifying Systems Management — Sun Management Center	2
A Comprehensive Systems Management Tool	2
Architectural Goals	4
2. Architectural Overview	7
Intelligent Agent-Based Architecture	7
The Three Tiers of Sun Management Center	9
Development Environment	11
3. Sun Management Center Console Layer	13
Console Functionality	14
Console Layer Components	15
Java GUI Components	16
Awx Components	16
Other Core Console Layer Components	22
Sun Management Center Client API	22

4. Sun Management Center Server Layer	23
Server Layer Components.	23
Sun MC Server	24
Trap Handler	27
Event Manager	28
Server Agents	28
5. Sun Management Center Agent Layer	29
Object-Oriented, Intelligent Agents	29
Agent Subsystems	30
Flow of Management Operations.	33
Probe Daemons.	34
6. Conclusion	35
References	37

The Systems Management Challenge

IT organizations face a difficult challenge in managing the availability of applications and computing resources within the enterprise. The growth of networks and distributed systems has led to an increasingly complex heterogeneous environment, encompassing a broad spectrum of hardware, software, and operating systems. Today, systems range from PCs and technical workstations on users' desktops, to small and mid-sized servers in departments, all the way up to large enterprise servers and mainframes in the corporate datacenter. Computing resources may be geographically dispersed across a business campus or around the world to support global business operations. The proliferation of LANs and WANs means that users can access corporate information assets almost anywhere, any time of day or night.

In a recent trend in distributed corporate computing, the use of mission-critical applications has blossomed, helping companies to become more competitive and conduct business more effectively. The mission-critical nature of these applications, however, is aggravating an already difficult system management task. Users are demanding systems and applications that are continuously accessible and available — with expectations for improved levels of service that are constantly on the rise.

As the demands for acceptable service levels and the complexity of the computing environment have increased, administrators have responded by standardizing procedures and adopting network-aware tools. While limited in functionality, many of these tools have helped address the need for remote network management. Still other tools allow administrators to monitor

individual systems and hardware components. To meet the rising demands for better levels of service, it is crucial both to manage and monitor the availability of applications and data, as well as the availability of individual systems and networks, and administrators still lack an integrated way of doing so.

While the job of managing systems, applications, and data is becoming increasingly complex, IT managers must still control costs. This means managing staffing levels and leveraging the existing skill base, while reining in education and training expenses.

Simplifying Systems Management — Sun™ Management Center

Clearly, systems management technology requires continued innovation. To this end, Sun Microsystems, a leader in designing products for increased applications and data availability, has created the Sun™ Management Center systems management framework.

Sun Management Center (Sun MC) provides *a single point for managing all Sun components in the enterprise* — Sun systems, Sun storage, and the Solaris™ operating environment. To facilitate systems management in heterogeneous environments, Sun MC provides *a flexible framework for integrating third party systems management products*, including leading enterprise management solutions like Tivoli's TEC, Hewlett Packard's Openview IT/OPS, and Computer Associates' Unicenter TNG. To extend the capabilities of the Sun MC product, Sun also offers the Sun MC Developer Environment, which allows new management modules to be easily created and applications to be easily integrated with the Sun MC console.

A Comprehensive Systems Management Tool

Sun Management Center is a comprehensive systems management tool, providing a broad range of functionality:

- *Proactive management and predictive failure capabilities*, allowing systems administrators to react (or automated remedial action to occur) before a debilitating failure, increasing overall systems and applications availability
- *Configuration management controls*, enabling administrators to remotely monitor or perform dynamic reconfiguration tasks, reducing reboots and the amount of costly downtime

- *Event and alarm management*, using complex rules and alarm limits to automate diagnosis, provide ongoing status information, and immediately notify administrators of significant events via the console, email, pagers, etc.
- *Automated management of common administrative tasks*, easing the burden of everyday administration
- *Domain-aware agents*, allowing dynamic system domains on high-end Sun servers to be monitored independently
- *Topology views*, showing a hierarchical topological map of the objects being managed, quickly familiarizing administrators with the Sun elements in the environment. A “discovery” process automatically builds the topology view, which can be divided into several administrative domains for distributed management.
- *Physical views*, displaying photo-realistic images of hardware components and relevant information such as network interface types, disk types, processor module speeds, etc. Components with associated events are highlighted, allowing administrators to easily identify areas of interest.
- *Logical views*, showing a hierarchical tree of managed hosts, including all related hardware, dynamic system domains, storage, and operating system components, and highlighting any components with associated events
- *Graph views*, displaying CPU, memory, disk, and network performance metrics
- *Per-process display* that extracts specific information on process resource usage and behavior, allowing an administrator to monitor the load an application imposes on the system
- *Application monitoring*, allowing administrators to check the health and performance of application processes, to examine and parse log files for recurring problems and particular status messages, and to monitor application files and directory statistics
- *The Sun Management Center Developer Environment*, enabling organizations to design, develop, and integrate third-party applications, tools, and customized solutions based on the Sun MC framework

In the future, through Sun and third party products, Sun Management Center could be extended to provide these additional capabilities:

- *Trend analysis and historical data storage*, facilitating more effective performance analysis, system sizing, and capacity planning efforts
- *Database and applications health monitoring* through the integration of Sun Management Center with products like BMC Patrol, allowing administrators to track status and provide better levels of data and applications availability
- *Interface to on-line diagnostics* with the addition of Enterprise Diag Tool, enabling hardware problems to be easily detected, isolated and resolved
- *Firmware and patch management*, facilitating easy firmware upgrades and patch administration with the addition of Enterprise Configuration and Service Tracker

A further discussion of Sun MC's capabilities is included in the Sun technical brief, *Sun Management Center — Managing the Integrated Enterprise*.

Architectural Goals

In designing Sun Management Center, Sun emphasized five primary goals:

- *Standards-based* — Sun Management Center is based on the industry-accepted standards of the Java™ programming language and the SNMP management protocol. The Sun Management Center management console is written entirely as a Java application. The SNMP management protocol, widely used by other network and systems management products, is also used by Sun Management Center agents for communicating status and alarm information. The use of standard interfaces and protocols simplifies the integration of Sun MC with third-party system management tools.
- *Extensible* — Sun Management Center is an extensible framework, allowing third party enterprise system management solutions like Tivoli's TEC, HP's OpenView IT/Ops, and Computer Associates' Unicenter TNG to be easily integrated. By using these tools in conjunction with the Sun MC framework, system managers can use a single interface to manage an entire heterogeneous enterprise, while very effectively managing the Sun elements. To further extend the capabilities of the Sun MC product, Sun offers the Sun Management Center Developer Environment. Using the tools provided in the Developer Environment, organizations can build new modules to monitor applications (for example, to monitor the status of a

database or a business-critical ERP application) and can integrate applications with the Sun MC console. The Sun MC Developer Environment also includes documentation for creating customized management rules.

- *Scalable* — Sun Management Center can manage a single system or it can scale up to hundreds of servers and desktops. In a large environment, multiple administrative domains that span geographies and timezones can be defined, allowing management teams to work concurrently and independently.
- *Flexible* — Easily customized, Sun Management Center can provide centralized or distributed management based on a management hierarchy. Systems can be grouped by location, server role, administrative responsibility, etc. Views are also customizable to present information in a tailored format.
- *Secure* — Sun Management Center includes a complete security model to authenticate system managers who use it, to protect system management information from unauthorized access, and to ensure data integrity.

This paper describes the architecture of Sun Management Center in detail and explains how it is designed to effectively address enterprise-wide system management. Chapter 2 provides an architectural overview, while subsequent chapters explore the Sun MC components in more detail.

Architectural Overview



Intelligent Agent-Based Architecture

Like many other system and network management tools, Sun Management Center is based on a manager/agent architecture. In this paradigm, a “manager” monitors and controls managed entities by sending requests to “agents” residing on the managed node (Figure 2-1). Agents are key software components that traditionally collect management data on behalf of the manager.

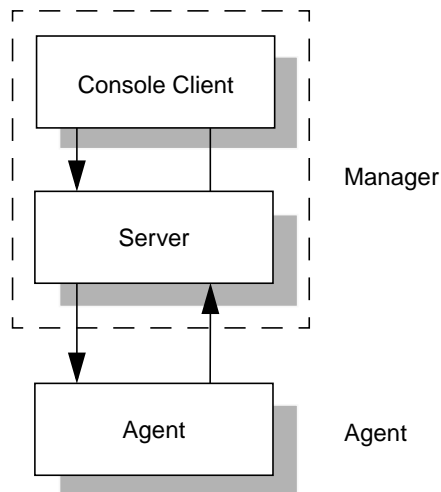


Figure 2-1 Manager/agent architecture

Unlike most other systems management frameworks, Sun Management Center uses *an intelligent agent-based architecture*. Instead of concentrating the responsibility for managing objects at the management station, much of the management responsibility is pushed down to the agent level, closest to the managed object. System management tools that concentrate decision-making and state in the management station typically do not scale well as the number of managed nodes increases. Distributing this responsibility to the agents in Sun Management Center greatly improves its ability to scale.

An intelligent agent-based architecture also enhances Sun Management Center's reliability and availability. Sun Management Center agents work autonomously — they are not dependent on other software components since they collect and process data locally. Based on SNMP protocol, agents can independently run processes or signal events via an SNMP trap, even if the connection to the manager is severed.

Intelligent agents also offer a savings in network bandwidth since agents collect data on the managed nodes, only reporting status and significant events as necessary to the server and management console.

The Three Tiers of Sun Management Center

With Sun Management Center, the manager itself is based on a traditional client/server architecture. Thus, the manager/agent architecture for Sun Management Center translates to a three-tiered architecture, as shown in Figure 2-2.

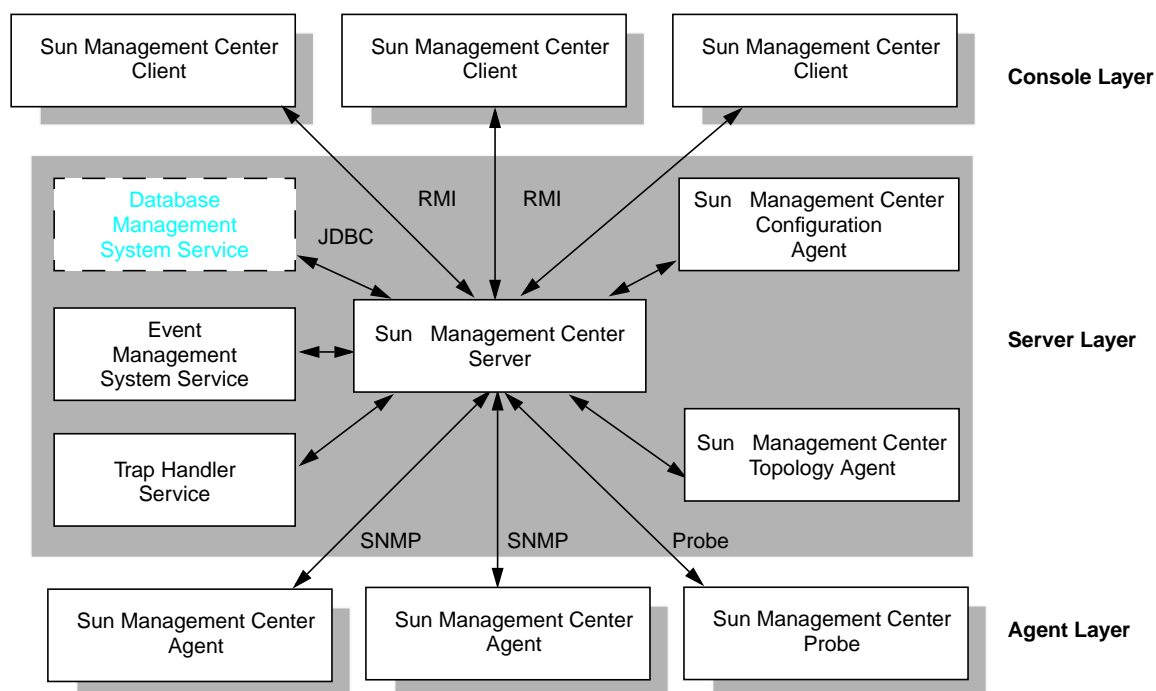


Figure 2-2 Detailed architectural view of Sun Management Center

The three tiers in the Sun Management Center architecture consist of:

- The console layer
- The server layer
- The agent layer

At the console layer, management clients are Java-based, providing platform-independent, network access to system management information. The clients display the managed objects and allow system administrators to manipulate

object attributes and properties. As shown in Figure 2-2, multiple client views can exist for different users — an important capability when system management tasks are distributed among a team of different system managers.

At the core of the Sun MC server layer is the server itself. Mostly written in the Java programming language and multi-threaded, the server securely and intelligently processes and routes requests between console clients and agents. One of the server's advantages is its ability to recognize redundancy — it recognizes when identical information requests are made independently by different clients and merges them. In multi-user environments, this efficiency can yield a significant savings in system and network resources.

The Sun MC server also dispatches events and alarms to the subscribing consoles. The server relates incoming alarms to client jobs in a process called *trap correlation*. By identifying jobs that are affected by system events, the server can immediately re-issue these requests and update Sun MC clients immediately after a system event. This has the important effect of making the polling-based mechanisms of SNMP appear to be subscription-based to the Sun MC clients.

In addition, the server provides a centralized point of access for distributed management information. This enables a robust security model to be easily implemented and enforced within the server.

The Sun MC agent layer consists of the Sun MC agents and probe components deployed on the managed nodes. Sun MC agents manage objects by intelligently and autonomously collecting and monitoring data. The agents use rule-based technology to determine the status of the managed objects. Sun MC agents store the data and status of managed objects in a Management Information Base (MIB) — this information can then be accessed by the Sun MC server via standard SNMP protocol. Agents may automatically generate alarms or perform actions based on detected conditions, enabling predictive failure capabilities and proactive system management.

The console, server, and agent layers are further described in chapters 3, 4, and 5 respectively.

Development Environment

No two computing environments are the same, or have the same requirements for system monitoring and management. As a result, organizations need to be able to create solutions that meet their diverse needs, as well as ensure enterprise-wide integration of critical management tools. The Sun MC Developer Environment provides an application programming interface (API) and documentation to help organizations plan, design, develop and integrate third-party applications, tools and customized solutions with the Sun MC framework. Available as a separate product, the Sun MC Developer Environment enables organizations to:

- Build modules

The Sun MC Developer Environment includes all the tools needed to implement dynamically loaded modules — encapsulated sets of monitoring functions that focus on a particular aspect of system or application health and performance. In particular, organizations can generate module configuration files, create and realize a data model, and write data acquisition code with TCL procedures, shell scripts, or shared object libraries.

- Build console applications

No management application can provide all the unique features that every organization requires. With the Sun MC Developer Environment, organizations can customize or build new console applications, modify configuration files, and restart the console to include the new or modified capabilities.

- Write rules

Detecting alarm conditions and subsequently triggering alarm actions is the core of the Sun MC framework. With the Sun MC Developer Environment, organizations can customize alarm checking and rule evaluation techniques to capture comprehensive information during an event.

- Use the Client API

The Sun Management Center Client API contains a set of public Java classes that can be used to retrieve data from Sun MC servers and agents. Using this API, console applications can fetch live or historic data, support synchronous and asynchronous requests, and use a variety of Java language object classes.

- Conform to internationalization and GUI guidelines

The ability to display localized text is paramount in any monitoring and management environment. With the Sun MC Developer Environment, organizations can create programs that use native language conventions for labels, error messages, headings, titles, etc.

Sun MC Console Layer



The console layer consists of all clients that interact with the Sun MC server, including the primary user interface, the graphical console. Since the graphical console is written in the Java programming language, it is *platform-independent*. This frees the administrator to work remotely, from any Sun system or PC on the network, or even off-site via a PPP connection.

The Sun MC graphical console provides a visual representation of managed objects. It uses icons to represent individual hosts and networks, allowing the administrator to manipulate attributes and properties associated with managed objects and to set thresholds for object characteristics. Sun Management Center subsequently *signals alarm conditions* when those limits are exceeded, alerting the administrator through the graphical console, and in some cases, taking an appropriate pre-defined action. The web-based Sun Management Center console *enables the administrator to actively perform many system configuration and management tasks* — for example, the administrator can use the Sun MC console to remotely perform dynamic reconfiguration tasks on Sun Enterprise™ 3500-6500 servers.

Sun Management Center supports *multiple graphical consoles*, allowing several administrators to monitor and manage systems throughout the enterprise. Multiple management domains can be defined, enabling management tasks to be distributed in a secure fashion.

Console Functionality

The graphical console for Sun Management Center provides the following functionality:

- *General user interaction.* The console provides “forward”, “backward”, and “home” navigation buttons, allowing administrators to easily navigate between historical console views. In addition, the console supports “drill-down” views, allowing an administrator to click on an alarm icon, for example, to easily obtain more detailed information about the alarm. Certain views (such as network topology diagrams) are created dynamically and are customizable, allowing console views to be tailored according to specific needs.
- *Data management — acquisition and logging.* All data acquisition and data logging occurs at the agent layer. However, using the graphical console, the administrator modifies the data acquisition and logging properties, such as the refresh interval for acquiring data, where it should be logged, how frequently it is stored, etc.
- *Alarm management.* The administrator uses console screens to define alarm thresholds and corresponding actions to be taken if an alarm condition occurs. The graphical console also displays alarm summaries, allowing the administrator to quickly detect a change in status.
- *Performance management.* Performance data can be graphed, allowing the administrator to visualize common performance metrics, such as CPU usage, disk performance, etc.
- *Configuration management.* Using the Sun Management Center console, administrators can query existing system configurations, easily obtaining a full breakdown of system options and installed boards. Administrators can issue general administrative commands, such as commands to monitor or initiate dynamic reconfiguration (DR). On high-end Sun Enterprise 10000 systems, administrators can track hardware configurations independently within dynamic system domains.
- *Tool integration.* Using the Sun MC Developer Environment, other tools (based on JavaBeans™) can be integrated into the graphical console.

Console Layer Components

The graphical console is built on a core set of underlying components. These components are designed generically, allowing application programmers who are developing other client applications to take advantage of them. The console infrastructure is based entirely on JavaBeans component technology — JavaBeans is the reusable software component architecture for the Java application environment. The console layer's component-based design allows developers to easily create and integrate new management console applications using the Sun MC Developer Environment and the Client API.

As shown in Figure 3-1, the infrastructure of the console layer includes:

- Java GUI components, such as AWT (Abstract Windowing Toolkit) and JFC (Java Foundation Classes) components
- Awx (abstract windowing extensions) components¹
- The bean connector framework
- Service managers
- The configuration file system
- The data router
- Client utilities
- Sun MC Client API

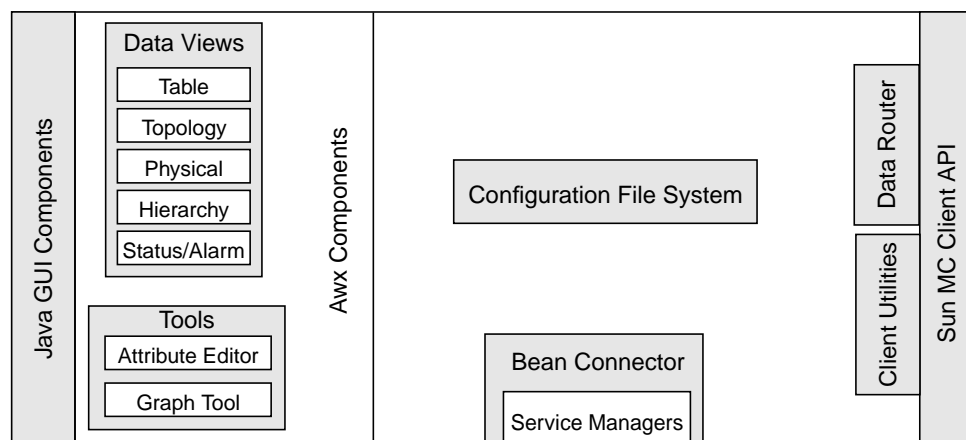


Figure 3-1 Infrastructure of the console layer

¹ Awx components provide functionality beyond AWT and JFC classes, enabling consistent presentation styles in Sun Management Center.

Java GUI Components

The graphical console of Sun Management Center is based on existing Java GUI components. The console uses Java Abstract Windowing Toolkit (AWT) classes as well as Java Foundation Classes, and in some cases, components developed by third parties (in particular, the plotting component was developed externally). The console adheres to the Java Management API (JMAPI) User Interface Style Guide where possible.

Awx Components

The Awx components include a number of “drop-in” bean objects that are assembled to provide the graphical console display. These components define consistent data presentation styles and provide useful tools for developers who are building other console clients. The Awx components provide a standardized interface to the bean connector model, allowing events and properties to be handled in a consistent fashion.

The Sun Management Center console uses these five views to present data:

- Tabular view
- Hierarchical topology view
- Physical view
- Logical view
- Status/alarm view

Tabular View

In some cases, data is presented to the console user in tabular form, such as the window entitled “Discovery Requests” in Figure 3-2. The layout of the tables is typically derived from information about the managed object and its properties that resides at the agent layer, which permits tables to be formatted consistently between different console instances.

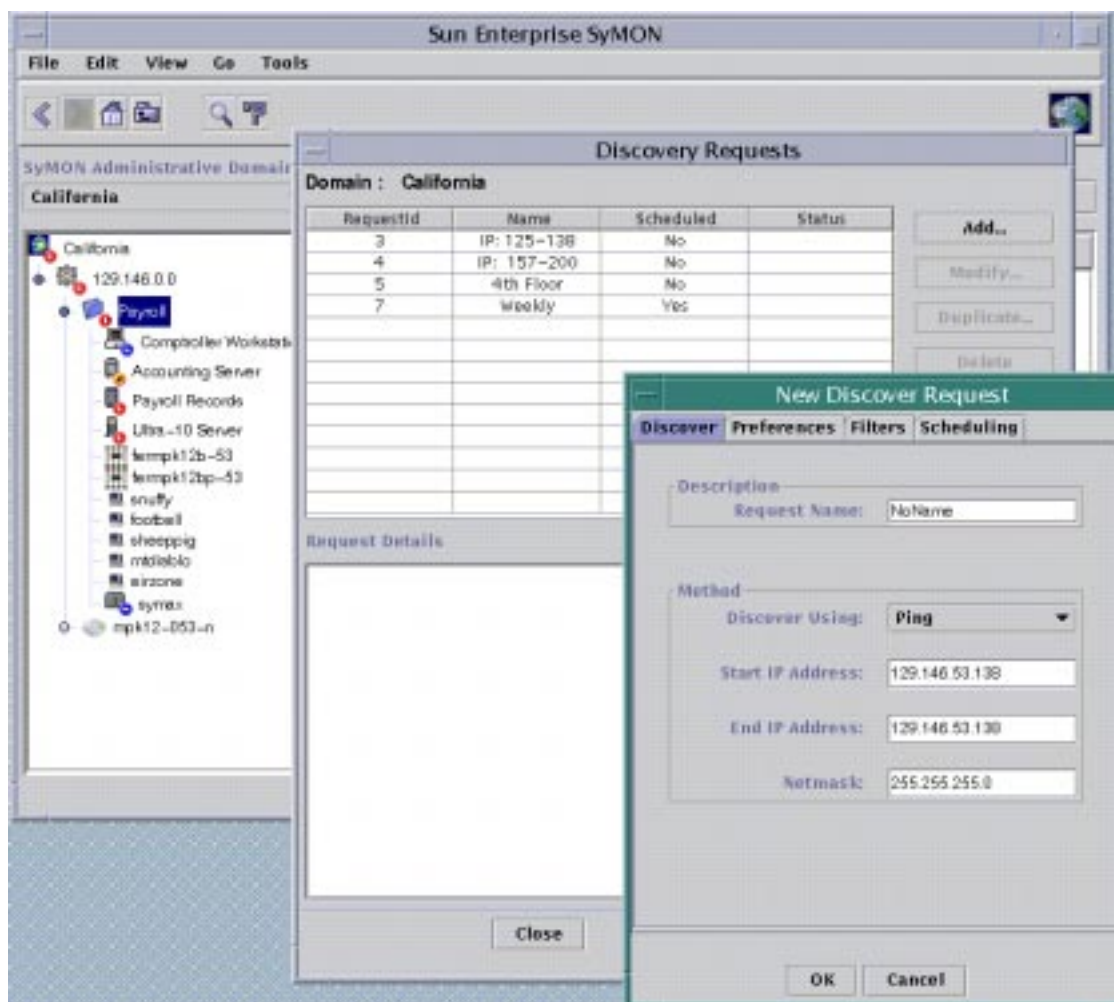


Figure 3-2 Example of tabular view

Hierarchical Topology View

The hierarchical view depicts the “tree” of managed objects and shows their relationships. Figure 3-3 provides an example. Navigating through this view, an administrator can easily traverse the managed object hierarchy.

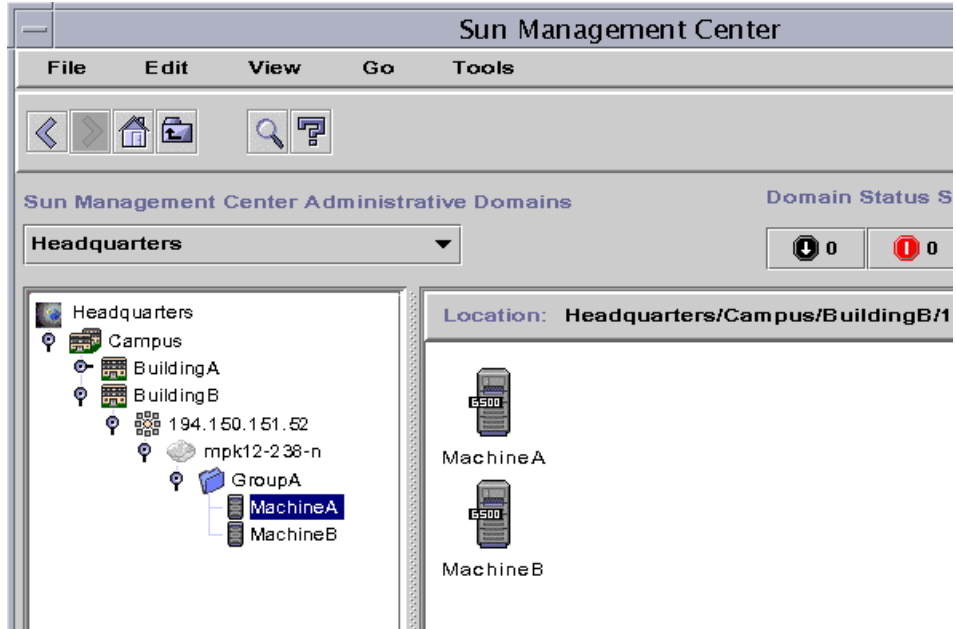


Figure 3-3 Example of a hierarchical topology view

Detail Views

For objects in the hierarchy shown in Figure 3-3, there may be detail views, such as a logical view, a physical view, or a status/alarm view.

Logical View

This detail view provides a logical representation of the managed node and all of its children. For example, Figure 3-4 shows the logical view of a server and all of its components. A logical view could include additional text annotations or images, allowing it to be easily customized.

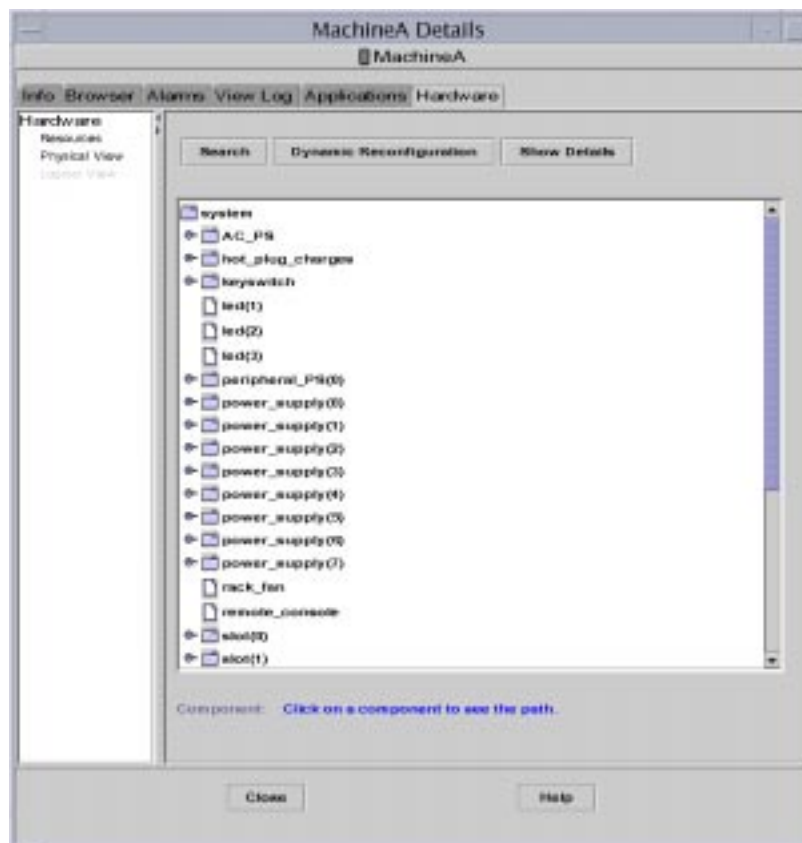


Figure 3-4 Example of a logical view

Physical View

The physical view provides a photo-realistic image of a managed object, such as that shown in Figure 3-5.

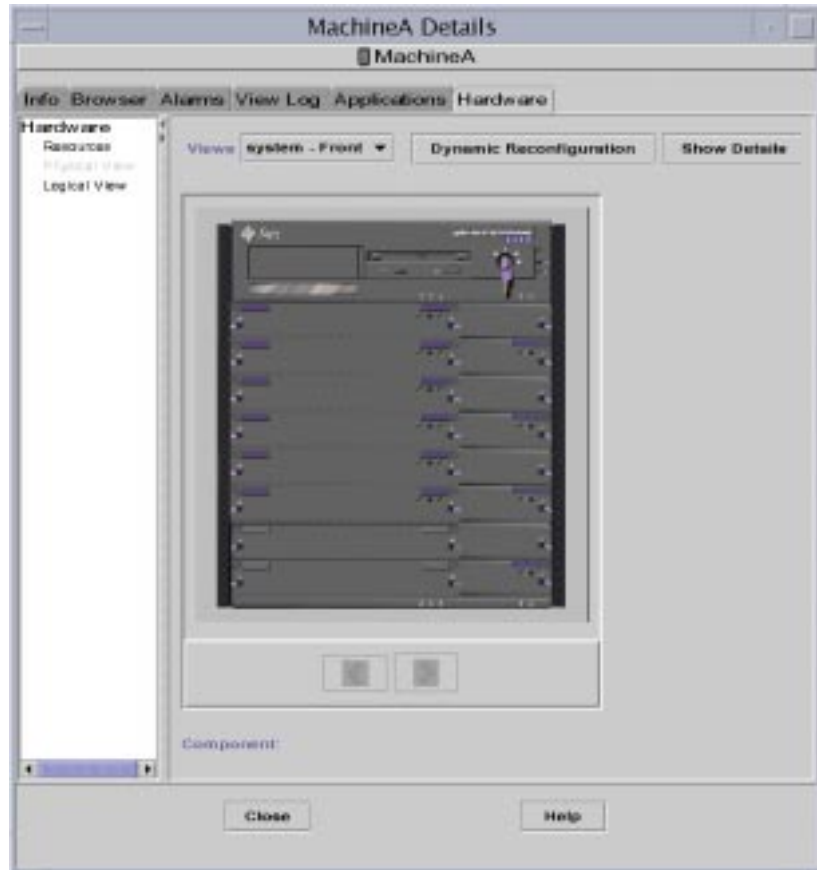


Figure 3-5 Example of a physical view

Status/Alarm View

As shown in Figure 3-6, this view provides status and alarm information about a managed object.

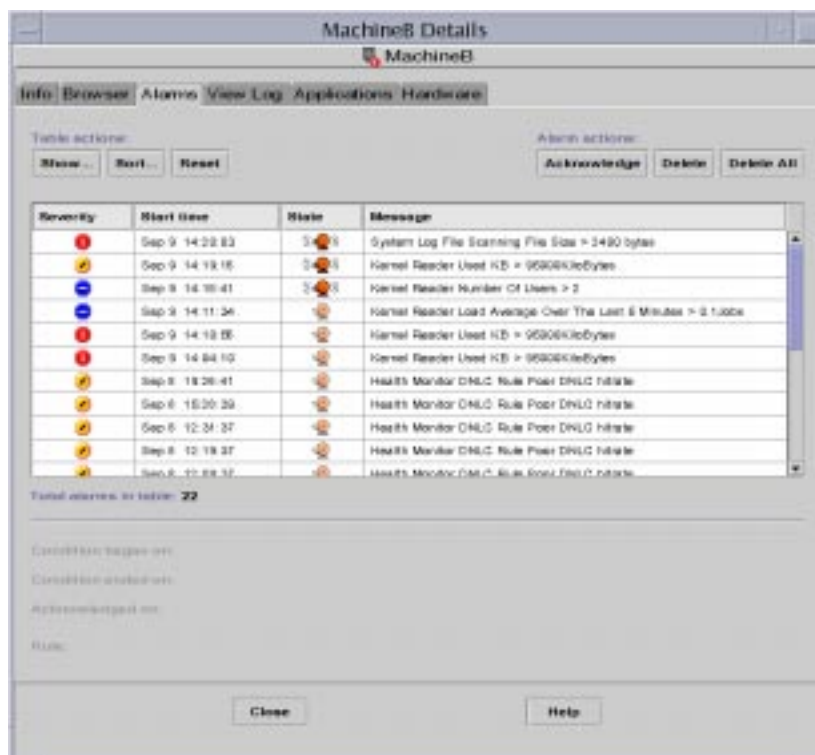


Figure 3-6 Example of a status/alarm view

Tool Components

Like the data view components, the tool components (Figure 3-1) are JavaBean components used by the Sun MC graphical console. These tools provide generic functionality and can be implemented in any Sun Management Center console. The tool components include:

- *The attribute editor.* With this tool, the administrator can modify properties of managed objects stored at the agent layer.
- *The graph tool.* This tool provides simple plotting capabilities to graph managed object data.

Other Core Console Layer Components

Figure 3-1 shows these other console layer components:

- *Bean connector.* At the heart of the console layer, the bean connector framework provides a generic mechanism to plug JavaBeans components into the console. Other custom beans — including those providing graphical functionality for a customized console — can hook into the Sun Management Center framework via the bean connector.
- *Service managers.* Service manager components provide non-graphical functionality for the console screen. For example, a service manager bean coordinates navigation within the management domains defined by the administrator.
- *Configuration file system.* When the administrator uses the Sun Management Center graphical console and saves console preferences, these changes are stored in a configuration file. The configuration file system provides a mechanism for tailoring the basic console design details (such as menubars, window decorations, etc.).
- *Data router.* All of the console components access the data router to retrieve data. The data router acts as a centralized point for data flow from the client APIs or other data sources.
- *Client utilities.* The client utilities provide non-console specific functionality such as data logging and internationalization.

Sun Management Center Client API

The console layer communicates with the server layer using Sun MC Client API (Figure 3-1), which is also provided as part of the Sun MC Developer Environment. The Client API is a standardized set of Java classes based on the Java RMI (Remote Method Invocation) API, and provides a consistent way for the graphical console — as well as any other console applications — to access the server and retrieve data and properties from managed objects located at the agent layer. With the Sun MC Client API, third party tools can retrieve data collected by all of the Sun MC agents.

Sun MC Server Layer



The Sun Management Center server layer provides a secure centralized point of access for all system management operations. All requests from the console layer — intended to modify object properties or retrieve data from the agent layer — are funneled through the Sun MC server. The server receives all alarms from the agent layer, quickly dispatching them and status information to all interested console clients.

Since the Sun MC server acts as a focal point, it provides a number of core centralized services. First, it receives and routes requests from multiple user consoles. The server *recognizes duplicate requests*, intelligently consolidating them for higher network and system efficiency. Secondly, the server *enforces the security model*, authenticating users and handling all session management.

Written mostly in the Java programming language and based on JavaBeans components, the server is *highly configurable and extensible*. The component-based design of the server makes it easy for Sun to add functionality, such as implementing support for new protocols or security models. A configuration file specifies the bean components that the server should instantiate and assigns properties for each bean.

Server Layer Components

As shown in Figure 4-1, the server layer includes:

- the Sun MC server process itself, which provides services for job and event management as well as authentication and session management services
- the trap handler

- the event manager
- the topology agent
- the configuration agent

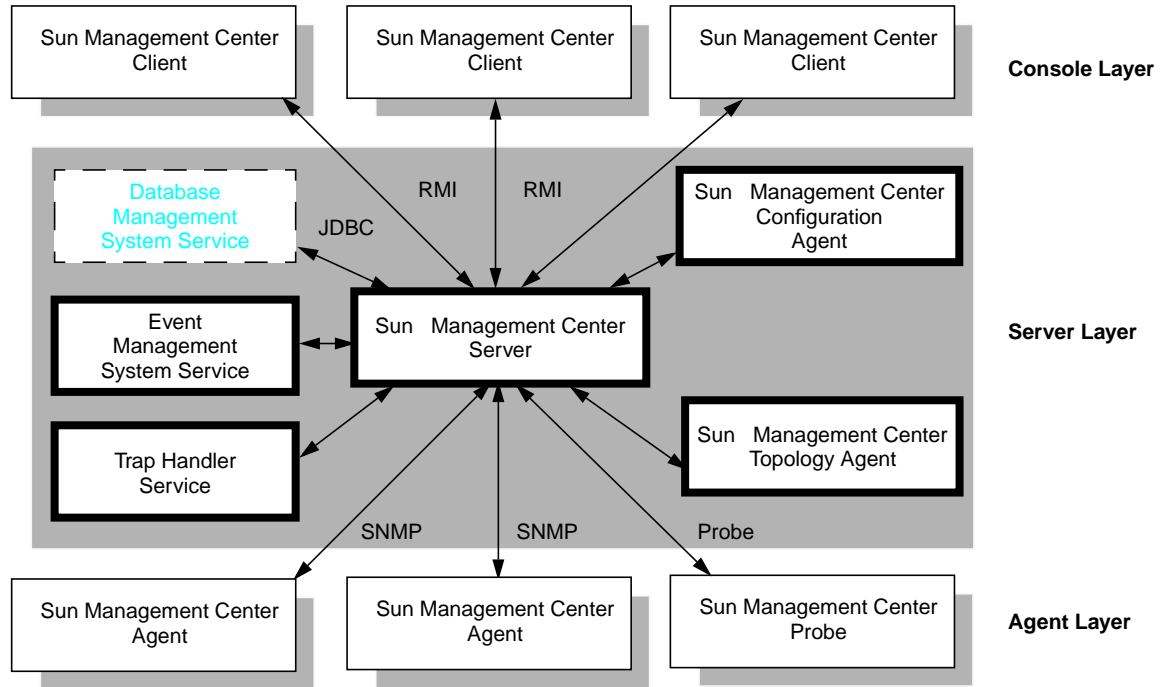


Figure 4-1 Functional components of the Sun Management Center server layer

In the future, the server layer could interface with a database management system (via the JDBC™ Database Access API) for centralized long-term data logging and retrieval.

Sun MC Server

Like the graphical console, the Sun Management Center server is built from JavaBeans components (Figure 4-2):

- *Receptor beans* receive external RMI or SNMP data requests
- *Emitter beans* initiate external data requests
- *Remitter beans* format each data request as a URL (Universal Resource Locator) and optimize data requests

- *The timer bean* provides a time clock for the other server beans
- *The security manager bean* provides centralized security and session management services to the other server beans
- *Security model beans* implement the security model, including secure protocols for the receptor and emitter beans

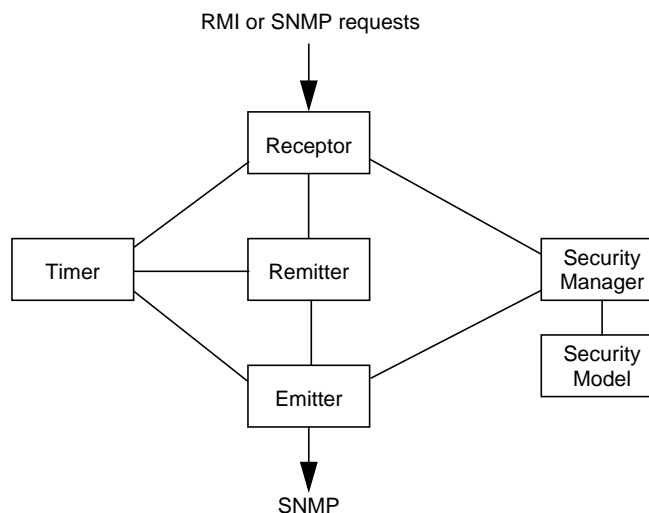


Figure 4-2 Bean components in the Sun MC server

Receptor, Remitter, and Emitter Beans

When an SNMP or RMI data request comes to the server, an SNMP or RMI receptor bean accepts the request, and routes it appropriately. For example, if the request is an SNMP trap, the request is translated and issued to any listening emitter beans.

The job remitter bean, at the heart of the Sun MC server, caches data requests in a *periodic job cache*. (Since the server is stateless, the periodic job cache actually resides in the agent layer — see the discussion of the SNMP subsystem in the next chapter.) The job remitter bean performs optimization, recognizing repeated data requests and merging them into a single request. The job remitter bean also correlates trap data requests with cached periodic requests, enabling any matched jobs to be immediately resent. This feature, known as *trap correlation*, allows status updates to occur asynchronously, without having to wait until the next scheduled polling time.

For outgoing SNMP requests, an SNMP emitter bean implements the actual SNMP protocol stack. Outgoing SNMP requests use a URL to identify the object data being requested. The SNMP emitter bean matches URLs to cached OIDs (Object Identifiers), retrieving the cached OID data if possible or issuing an SNMP request to the agent on the target host.

Security Management and Security Model Beans

The security manager and security model beans (Figure 4-2), along with a server agent, implement the bulk of the Sun MC security framework.

The security manager bean facilitates session authentication and management. When an administrator first logs into a Sun MC console, authentication occurs via a username and password. (By default the authentication mechanism is the standard UNIX authentication scheme.) Once authenticated, the user establishes a session with the server based on the user's security credentials. The security manager bean requests these credentials from a server agent (the security configuration agent, discussed below). If a session remains idle, the server manager bean will also automatically log out the user after a period of inactivity.

All requests from the Sun MC console are then arbitrated according to the user's credentials. The server sends and receives SNMP requests on the user's behalf, all of which include the user's identity. Agents restrict access to managed objects based on an Access Control List (ACL) mechanism. Access can be granted to individual users, groups of users, or to all users. Access is also hierarchical — for example, an administrator typically has access to an administrative domain that represents a network of systems, and thus has access to all of the managed objects within that domain.

Sun Management Center defines two special users — the Interagent User and the Superuser. The Interagent User is a special user that agents use to interact with other agents. The Superuser is a special user for handling the authentication tables which protect the overall security framework.

Server API

The server API encompasses the JavaBean components described here. To implement a new protocol or security model in the server, Sun could write new components — receptors, emitters, security models, etc. — using this API and integrate them easily into the existing Sun MC framework. (The server API is for Sun's use only and will not be published.)

Trap Handler

The trap handler (Figure 4-1) is the server layer component that receives and processes all SNMP traps. The trap handler listens for traps on a well-known UDP port and processes them accordingly. There are five trap types:

- *Value traps.* A managed object generates a value trap when it refreshes its associated management information.
- *Status traps.* When the status of a managed object changes such that it exceeds a particular threshold or it meets the specified rule conditions, then a status trap occurs.
- *Message traps.* Message traps convey informational messages and are not usually associated with a particular managed object.
- *Configuration traps.* When a configuration change occurs, a configuration trap communicates the change to the server layer. For example, loading or unloading an agent module, or modifying object properties (such as altering thresholds or refresh intervals) will result in a configuration trap.
- *Subscription traps.* These are used to tell the trap handler that a job is interested in receiving subsequent traps. (See the discussion of SNMP trap correlation in the next chapter.)

Based on the type and content of the trap received, the trap handler will log the trap, or forward it to the interested agents (for example, to jobs that have previously sent subscription traps). Agents receiving the forwarded trap can then take the appropriate action — they can send immediate notification to the administrator or perform some other user-defined action. Trap actions are specified in a configuration file, which defines the actions to be taken for each trap type.

One advantage of the Sun Management Center design is that the trap handler is stateless. If the trap handler becomes inaccessible (e.g., the Sun Management Center server process goes down), the only effect is that alarm delivery is temporarily delayed. (In this case, traps are delivered to the trap handler as soon as the server restarts.)

Event Manager

The event manager (Figure 4-1) uses a rules-based mechanism to determine when a significant system management event has occurred. Rules can involve multiple variables and expressions, enabling more complex conditions to be represented. (Administrators can customize rules for site-specific requirements

using the Sun MC Developer Environment.) When an event condition is met, the server logs the event, and agents autonomously alert administrators via the graphical console, email, pagers, etc. Agents can also be configured to react proactively, initiating high-level management operations on their own.

Server Agents

Server agents are really the same as other Sun MC agents, differing only in terms of the specific management modules loaded into the agent (see the discussion of agents and management modules in the next chapter). Because server agents are based on loadable management modules, the server layer can be easily extended by modifying or adding new modules.

There are two key server agents (refer back to Figure 4-1):

- *The topology agent.* By managing user domains and the topologies for each domain, the topology agent enables the console to display and customize logical summaries of managed objects in a topological view. Using a network discovery module, the topology agent can automatically discover nodes on the network, helping administrators build topologies within their user domains. Other topology agent modules — such as the logical summary module — allow an administrator to use the Sun MC console to customize topology views using drag and drop.
- *The security configuration agent.* The security configuration agent provides security and configuration services to the server and Sun MC agents. This agent stores the security credentials of all Sun Management Center users. For example, whenever an administrator logs into a Sun MC console, the server consults the security configuration agent to validate the username and password and to retrieve the user's credentials (see *Security Manager and Security Model Beans*, earlier in this chapter).

The Sun MC agent layer monitors the managed nodes. Two types of components exist at the agent layer — *agents* and *probe daemons*. Both run as background processes on the managed nodes.

Object-Oriented, Intelligent Agents

Sun Management Center is based on an object-oriented paradigm in which objects are used to model various aspects of a system for the purpose of monitoring and managing that system. At the agent layer, agents are implemented using TCL Object Extension (TOE) technology, an object-oriented extension to TCL. The hierarchical nature of entities to be managed — systems, hardware components, and operating system-related components — lends itself well to a hierarchical, object-based model.

Sun Management Center uses *sophisticated, intelligent agent technology* on the managed nodes. Unlike many other system management frameworks, Sun Management Center focuses management capabilities at the agent layer, closest to the managed nodes themselves. Agents perform their management tasks autonomously, gathering and processing data and alarms locally, without relying on the server or the console layers. This promotes reliability and availability, since the agents continue to log data and process alarms even if the connection to the server and console layers is temporarily severed.

Sun MC agents perform management tasks through the use of *management modules* — a design which allows Sun Management Center to be easily extended and customized. A default set of agents and modules are loaded

through the console, and *additional modules can be loaded or unloaded dynamically* (management modules are discussed in more detail below). The Sun MC Developer Environment includes necessary tools and documentation to enable customers and software vendors to develop their own management modules.

Sun Management Center agents are *based on SNMP standards*. They support the current SNMPv2 and SNMPv2u (secure SNMP) standards, and the earlier SNMPv1 protocol for interoperability.

Agent Subsystems

Figure 5-1 shows the logical subsystems of an agent.

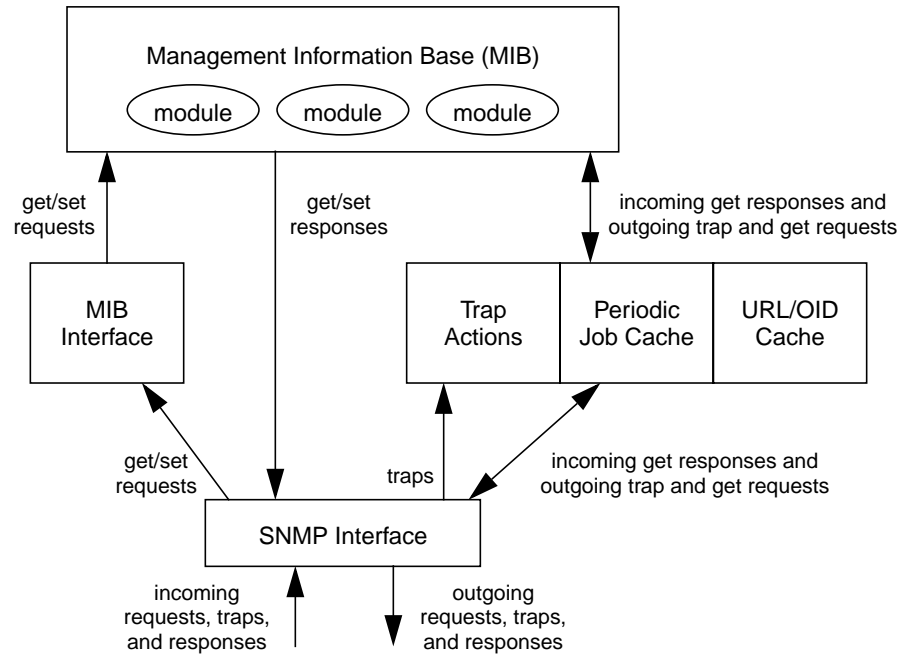


Figure 5-1 Subsystems of a Sun Management Center agent

Management Information Base (MIB) and Management Modules

As shown in Figure 5-1, at the heart of each agent is a *Management Information Base (MIB)* that acts as a repository for the managed objects. The managed objects — a collection of TOE (TCL Object Extension) objects — together represent a model of the system and its components being managed. The managed objects are arranged in a tree, showing the hierarchical relationship of the components.

Within the MIB, managed objects are logically grouped into *management modules* that collectively implement management functions. Each agent loads a set of management modules that focus on a specific aspect of system or application health or performance. These modules enable Sun Management Center to manage five types of entities:

- hardware components of Sun systems (such as processors, system boards, memory, disks, power supplies, etc.)
- operating system-related components in the Solaris operating environment (such as CPU usage, swap space, processes, files systems, etc.)
- dynamic system domains in high-end Sun Enterprise 10000 servers
- software applications (through the use of customized modules designed to monitor application health)
- other Sun MC agents (such as legacy SNMP agents running on remote hosts)

Other modules provide basic agent functionality such as agent clustering, data logging, and event management.

Using the Sun MC console, an administrator can alter existing functionality by loading and unloading modules. For example, it may be desirable to monitor only a subset of entities on a desktop system, in which case an administrator may choose to unload particular modules. When modules in a running agent are loaded or unloaded, these changes are recorded in a configuration file. Thus, if an agent is restarted, the previous module configuration is automatically reloaded.

Each agent includes a core module loader, which bootstraps the agent by pulling in all of the required modules. The core module loader can load a module that contains object loaders, which enable modules to be self-configuring. Some examples of self-configuring modules are:

- The board module, which discovers all slots on a system board and subsequently loads a slot for each board
- The slot module, which discovers what card is present in a particular slot and loads the corresponding card module
- A database module, which could determine what processes, logfiles, and tables comprise a database instance and in turn load the appropriate modules to manage the database components. (Database modules are not part of Sun Management Center, but the flexibility of the Sun MC framework and the Developer Environment would make it easy to develop and integrate them.)

Through the use of discovery algorithms and self-configuring modules, powerful Sun MC agents can model complex systems and applications with very little interaction from the system manager.

SNMP Interface

Sun MC agents store management data in the MIB where the server can access it via SNMP. To facilitate secure SNMP communications, the SNMP interface (Figure 5-1) uses SNMPv2u, a Version 2 SNMP protocol that implements a user-based security model. Legacy SNMP agents are supported via SNMPv1 protocol, allowing Sun Management Center to integrate easily with existing network management applications based on the earlier SNMP protocol.

When incoming SNMP packets are received on the SNMP port, they are decoded to determine the SNMP packet type. SNMP “get” and “set” requests (e.g., operations to modify or retrieve MIB object attributes) are routed to the MIB interface, while SNMP traps are forwarded to the trap actions object.

The *MIB interface* (Figure 5-1) authenticates incoming SNMP requests. If the request is valid, it interacts with the MIB to modify or retrieve object properties. For SNMP “get” requests, the MIB returns the result to the requestor through the SNMP interface.

The agent’s SNMP subsystem maintains a *periodic job cache* for the server — a table of periodic SNMP requests that are sent out for the purpose of data acquisition. The job cache enables the server to optimize periodic SNMP requests by merging identical requests and facilitating *trap correlation*. For trap correlation, the agent sends a subscription trap to the trap handler in the server

layer, notifying the trap handler that this job is interested in asynchronous alarms. (See Chapter 4 and the discussion of the job remitter bean in the server layer.)

The trap handler then forwards all status traps relevant to jobs in the cache. If the trap OID (object identifier) matches a job OID in the job cache, then the status trap is immediately sent to the requesting job. This allows the agent to perform asynchronous updates when a status change occurs, rather than waiting for the next regularly schedule sample time.

Outgoing SNMP requests to other agents use a URL to identify the MIB object data being requested. This URL must be converted to a MIB OID. The *URL/OID cache* (Figure 5-1) in the SNMP subsystem caches the most recently sent URL/OID mappings.

Flow of Management Operations

Sun Management Center performs monitoring or management operations at regular intervals or on demand. Typically this is the sequence that occurs:

1. *Data acquisition.* To refresh status information in the MIB, an agent performs data acquisition, generally on a configurable, periodic basis. A *refresh service* object implements the actual data acquisition, and can include command line programs, TCL procedures, and TCL extensions (shared object libraries). The refresh service object acts autonomously, without any directives from the Sun MC server or console layers, allowing status information in the MIB to be updated reliably.
2. *Data cascading.* Large amounts of data may be acquired in a single acquisition and cascaded into several managed properties or objects. Data may also be digitally filtered.
3. *Alarm rule checking.* When data is collected, it is checked to determine the status of the managed object. Alarm conditions can be signalled through simple comparison checks (using threshold limits or regular expression matching) or more complex rule evaluation. Rules may embody complex computations and relationships, enabling more complex conditions to be represented. Rules can depend on other derived status information — for example, a rule might generate an alarm when a disk is over 20% busy with a service time exceeding 50 milliseconds. (The Sun MC Developer

Environment includes documentation that describes how to develop and customize rules.) In decreasing order of severity, the standard alarm levels are down, critical, alert, caution, and indeterminate.

4. *Alarm actions.* When an alarm status condition occurs, the agent sends an SNMP trap to the trap handler in the server layer, which notifies other agents about the trap. Based on the type of trap, agents will then take the appropriate user-defined alarm actions (such as sending emails, dialing pagers, etc.). (See *Trap Handler* and *Event Manager* in the previous chapter.)
5. *Data logging.* Alarms are automatically logged to maintain a persistent record of alarm events. Sun MC agents also support data logging, enabling an agent to store collected management data for later use.

Probe Daemons

In addition to intelligent agents, probe daemons reside at the Sun MC agent layer. Probe daemons provide a secure mechanism for executing ad-hoc commands on the managed node. For example, probe daemons facilitate:

- Running interactive commands (such as viewing the contents of a log file)
- Retrieving bulk data for console display (such as a process listing)

Probe daemons use the same security model as agents, and thus are restricted to authenticated users.

Conclusion



Computing infrastructures have become increasingly complex, with global operations, wide-ranging systems and software, and intricate networks. Users in these environments are demanding higher and higher levels of service — they seek better applications and systems availability in addition to improved levels of system performance. More frequently, data and applications must be available around the clock to meet mission-critical requirements. Effective systems management enables system administrators to proactively respond to imminent problems, and increases systems and applications availability.

Sun's strategy is to offer outstanding manageability and availability in addition to performance and scalability. Sun Management Center provides a comprehensive systems management solution for all Sun components in an enterprise. In addition, by providing a development environment and a flexible framework for integrating third party system management tools, Sun Management Center can be the cornerstone of a robust systems and applications management environment.

References



Sun Microsystems posts product information in the form of data sheets, specifications, and white papers on its Internet World Wide Web Home page at: <http://www.sun.com>.

Look for abstracts on these and other Sun technology white papers:

Sun Management Center — Managing the Integrated Enterprise, Technical Brief, Sun Microsystems, Inc., 1999.

Availability Features in the Sun Enterprise 3500 to 6500 Server Family, Technical White Paper, Sun Microsystems, Inc., 1998.



Sun Microsystems, Incorporated
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300
FAX 650 969-9131
<http://www.sun.com>

Sales Offices

Argentina: +54-1-317-5600
Australia: +61-2-9844-5000
Austria: +43-1-60563-0
Belgium: +32-2-716-7911
Brazil: +55-11-5181-8988
Canada: +905-477-6745
Chile: +56-2-638-6364
Colombia: +571-622-1717
Commonwealth of Independent States:
+7-502-935-8411
Czech/Slovak Republics:
+42-2-205-102-33
Denmark: +45-44-89-49-89
Estonia: +372-6-308-900
Finland: +358-9-525-561
France: +33-01-30-67-50-00
Germany: +49-89-46008-0
Greece: +30-1-680-6676
Hong Kong: +852-2802-4188
Hungary: +36-1-202-4415
Iceland: +354-563-3010
India: +91-80-559-9595
Ireland: +353-1-8055-666
Israel: +972-9-956-9250
Italy: +39-39-60551
Japan: +81-3-5717-5000
Korea: +822-3469-0114
Latin America/Caribbean:
+1-650-688-9464
Latvia: +371-755-11-33
Lithuania: +370-729-8468
Luxembourg: +352-491-1331
Malaysia: +603-264-9988
Mexico: +52-5-258-6100
Netherlands: +31-33-450-1234
New Zealand: +64-4-499-2344
Norway: +47-2218-5800
People's Republic of China:
Beijing: +86-10-6849-2828
Chengdu: +86-28-678-0121
Guangzhou: +86-20-8777-9913
Shanghai: +86-21-6247-4068
Poland: +48-22-658-4535
Portugal: +351-1-412-7710
Russia: +7-502-935-8411
Singapore: +65-438-1888
South Africa: +2711-805-4305
Spain: +34-1-596-9900
Sweden: +46-8-623-90-00
Switzerland: +41-1-825-7111
Taiwan: +886-2-514-0567
Thailand: +662-636-1555
Turkey: +90-212-236 3300
United Arab Emirates:
+971-4-366-333
United Kingdom: +44-1-276-20444
United States: +1-800-821-4643
Venezuela: +58-2-286-1044
Worldwide Headquarters:
+1-650-960-1300