Solstice™ Enterprise Manager™

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

Please
Recycle

A Sun Microsystems, Inc. Business

SunSoft
2550 Garcia Avenue
Mountain View, CA 94043-1100
USA
1-800-SUN-SOFT
1-512-345-2412

United Kingdom	0800-96 27 61
France	05 90 86 09
Germany	0130 813 862

Printed in USA 1/95 94451-001

Cooperative Management

Overview

This paper articulates SunSoft's vision for management platforms.

Solstice™ Enterprise Manager (Solstice™ EM), a key component of this vision, is discussed in detail.

SunSoft's vision for management platforms is that in order to manage networks, you must network managers. This means that operators and applications must be able to manage anything from anywhere.

Operators and applications need to access a secure, multi-user, distributed management data repository which models network and computer system resources. Managers must be able to determine who can monitor and control which resources. Managers must also allow more than one operator or application to simultaneously analyze situations. And, managers must be able to deploy processing resources to gather, store, and present management information cost-effectively to operators and applications.

Solstice EM extends the SunSoft management platform product portfolio to address these requirements. Solstice EM provides basic management tools for monitoring and controlling network and system resources. It also functions as a runtime environment for third-party applications. Solstice EM also provides powerful interfaces which allow vendors to develop a new generation of secure, integrated management applications.

A Vision for Cooperative Management

Information Technology (IT) is key to managing business operations in today's fast-paced and dynamic environment. With IT's increased importance has come increased scrutiny by senior management. With the increased scrutiny has come increased pressure on IT professionals to ensure maximum return on investments in network and system resources.

Maximizing return has proven difficult with today's management tools. IT professionals have found themselves adjusting their needs to fit the tools' capabilities, when what they want is to be able to adjust the tools to meet their needs. As these tools become available, IT resources will align with an organization's structure and goals. IT operations will simplify. IT processes required to support major activities such as reorganizations, downsizing and mergers will be implemented consistently and in a cost effective manner. Aligning resources with business goals will maximize return on IT investments.

Comprehensive management of networks and systems is required to get this alignment, and the resulting optimization in IT return on investment. IT professionals have taken various paths to integrated, multivendor management that has resulted in a complex hybrid solution. This hybrid solution evolved as Element Managers and Manager of Manager solutions were deployed. Cooperative Management brings order to this hybrid environment.

The SunSoft vision of a Cooperative Management environment is one where technology and business relationships create links between multi-vendor tools to simplify IT management. The following sections describe the challenges created by this hybrid approach.

Element Managers

Vendor-specific ***Element Managers*** were deployed out of necessity. Until recently there were no industry-wide standards for management protocols or management data definitions. Each device vendor developed proprietary, and often device-specific, approaches. Element Managers for transmission equipment, switching equipment, hubs, routers, servers, and desktop devices proliferated in operations centers. While effectively managing discrete parts of an enterprise, Element Managers could not exchange information. This has created many inefficiencies in operations and planning functions.

Manager of Managers

Manager of Managers have been deployed to aggregate information obtained from lower-level Element Managers in the hope of improving operations and planning efficiency. This allowed coexistence with the extensive customer investment in Element Managers and also simplified development of the Manager of Managers to perform complicated, device-specific management tasks. However, this approach increased, rather than decreased, the number and complexity of the management systems in use. While this hierarchical strategy could reduce the number of consoles in an operations center, it did not reduce the number of management systems that an organization needed to buy, maintain, and learn to operate. Managers of Managers have proven to be too expensive, too complex, and too inflexible to be the foundation for integrated multi-vendor management.

Management Platform

The ***Management Platform*** has evolved to become the foundation of the solution for dealing with the scale and complexity of distributed systems and network management.

The Management Platform provides a standardized environment for writing network and system management applications. It separates the value-added management application software from the system-level mechanics of reading, writing, displaying, and storing management data. The best management platforms provide comprehensive basic tools for monitoring and controlling a wide range of systems and network devices. They also make it possible to combine different vendors' management software modules to create customized, multivendor management systems.

Management Platforms address the need in IT to manage as close as possible to each resource. This optimizes processing power and minimizes the bandwidth required to provide timely and accurate management information.

However, Management Platforms must evolve to deal with the increased complexity of the management functions required to support a networked systems environment. Management Platforms must be able to support multiple users, model a more complex environment and provide a richer set of basic management tools.

Hybrid Environment

Today's network and systems management operations use a hybrid approach to monitor and control resources:

- Management Platforms, and vendor- or task-specific applications
- Element managers for resources managed through proprietary protocols
- Telnet shells or console port configurations for older devices
- Various, often custom, tools for distributed system administration

This hybrid approach has created costly, complex, and inefficient islands of management tools.

Management Platforms are the key infrastructure for creating a "management backbone" to integrate islands of management tools. SunSoft is the leader in the Management Platform business.

The balance of this paper describes what SunSoft has learned from thousands of customers in terms of what is needed in the next generation of Management Platforms. SunSoft's portfolio of management platforms addresses customer requirements for secure multi-user support, a comprehensive management data repository, and better basic tools.

Evolving Management Platform Requirements

Multi-User and Secure

Because of scarce skill-sets, management responsibility has been redistributed, and is often shared between many operators and applications. Collaboration has become essential between generalists, generalist-to-expert, and expert-to-expert. Management platforms must enable simultaneous and coordinated execution of management tasks across these groups. This coordination requires tools that distribute information across hierarchical, peer-to-peer, and hybrid organizational structures.

In instances where the management of a resource crosses multiple management platforms it is imperative that the information pertinent to that resource remains consistent. A change made by an operator or application must be reflected across all relevant management domains.

Control and accountability are necessary when so many people and processes access management information simultaneously. Management platforms must support security policies flexible enough to meet IT management policies.

Richer, Distributed Repository

To deal with the complexity of today's IT environment, operators and applications must have easy access to configuration, state, and performance information. The management platform data model must be able to expand to include very complex resources, including other management platforms.

Management platforms must be easy to scale in order to handle the volume and diversity of IT resources. Functionality must be distributed to deal with the complexity of the managed resources and the cost of gathering management information. A new model for distributing information gathering, distributing information storage, and distributing application execution and presentation is needed.

Object-oriented technology provides a means to deal cost-effectively with the complexity and volume of management information needed by operators and applications.

For example, object oriented technology ensures that management application developers do not have to be concerned with the location or address of resources. To illustrate, if an administrator needs to know CPU utilization, the information might be available from a local management data repository, or the management platform may use a protocol to retrieve the information from a remote location. The application developer would not be required to learn SNMP, CMIP, or any other management protocol or protocol-intensive application programming interface (API) to gather the information.

In addition, object oriented technology, such as object inheritance, must be supported in order to simplify task automation. Object oriented technology makes it easier to address a group of objects with a simple command. For example, object *inheritance* could simplify the task of notifying an operator when "CPU utilization of all servers is below 50 percent." Object inheritance could enable a single command, rather than one command per server, to retrieve and send the information to the operator.

Object oriented technology also simplifies interoperating with complex and diverse managed resources. To simplify interoperability with applications, management platforms must be able to support standard object definition

languages, such as the Guidelines for the Definition of Managed Objects (GDMO) and the Interface Definition Language (IDL), to represent resources to be monitored and controlled.

Better Basic Tools

Several factors contribute to the fact that management platforms must provide more tools (i.e. core applications). Operators, and their managers, have grown accustomed to the productivity resulting from the functionality found in expensive Element Managers. Much of this functionality (e.g. color-coded alarm log displays) is common to Element Managers used in production environments. Today's management platform technology must enable this common functionality to be extended to all management domains and situations to maximize the effectiveness of training and documentation expenses.

Management platforms must be off-the-shelf, plug-and-play tools requiring minimum configuration and customization. Required functionality extends beyond real-time monitoring of simple devices. Required functionality includes trend analysis, event filtering, and report generation.

To manage today's complex IT environments effectively, operators and planners need a great deal of information in many different formats. Management platforms must be flexible enough to deal with the volume and diversity of the management information.

SunSoft Management Platform Product Line

SunSoft's portfolio of compatible management platforms provides powerful new ways for customers to build upon their investments in network and system management. SunSoft's range of management platform products include:

- Solstice™ SunNet Manager™
- Solstice™ Cooperative Consoles™
- Solstice Enterprise Manager

SunSoft focuses on providing the best management platforms in the industry rather than providing an entire management solution set for all environments. This focus provides a market opportunity for specialist software vendors to provide best-of-breed tools. Virtually all of the best management tools in the

industry are integrated with Solstice SunNet Manager. The quality of SunSoft's platforms means specialists can focus on their areas of expertise while SunSoft takes care of the platform and protocol standards for them.

Solstice™ SunNet Manager ™

Solstice SunNet Manager is the ideal solution for departmental environments. More than 15,000 copies have shipped. They are used to manage resources in workgroups, buildings, campuses, and other well-defined domains.

The Solstice SunNet Manager management protocol engines, called Proxies, are distributed. This relieves the console system and intermediate routers of polling load. It also makes management of multiple protocols, including CMIP and proprietary protocols, a basic part of the Solstice SunNet Manager architecture.

Solstice SunNet Manager 2.2 is available on SPARC® systems. It is also available for Intel® x86 machines. This allows customers to leverage the two most popular machine architectures in the world.

Solstice ™ Cooperative Consoles™

As an IT management organization grows, and multiple Solstice SunNet Manager consoles are deployed, it becomes useful to interconnect the topology information and network event information. This enables operators to work from the same network image and help each other solve problems.

Solstice Cooperative Consoles links multiple Solstice SunNet Manager consoles into a common distributed repository, giving a multi-user IT organization substantial flexibility.

A common repository allows straightforward transfer of control between operators. For example, in off-hours, a central site could substitute for multiple local day-shift operators.

Solstice ™ Enterprise Manager™

Solstice Enterprise Manager is SunSoft's next generation management platform. It embodies all of the features highlighted above and more in order to optimize Information Technology (IT) alignment with business goals. Solstice EM deployment can mirror the requirements of the IT organization.

The Solstice EM management framework can be deployed in a single-user, peer, hierarchical, or hybrid structure. This offers organizational flexibility to enable simple transfer of control, off-hours substitutions, or to implement a distributed or centralized arrangement. Solstice EM also offers geographical independence: IT operators can manage from anywhere on the network.

Solstice EM offers future-oriented integration. It integrates any protocol, any resource/object, any vendor tool, any standard application interface, and even other management systems. It integrates data to provide operators with more meaningful action requests.

Solstice EM is API independent. It can support any of the emerging network or system management API standards. It includes compatibility with Solstice SunNet Manager 2.2, and the Portable Management Interface (PMI), supported by Solstice EM and NetLabs' DiMONS 3G™. It can also support any interface from XMP to something more proprietary. Solstice EM maximizes integration with the best-of-breed management tools in the industry, and allows IT managers to focus on any interface, even in a time of rapid change.

Migration Paths

SunSoft management platforms protect investments in agents, applications and training.

Solstice EM supports the Solstice SunNet Manager 2.2 dynamically-linked service libraries. Applications developed to the Solstice SunNet Manager 2.2 dynamic libraries are compatible with Solstice EM. Since application will run, out of the box, on Solstice EM, investments in training and documentation are protected.

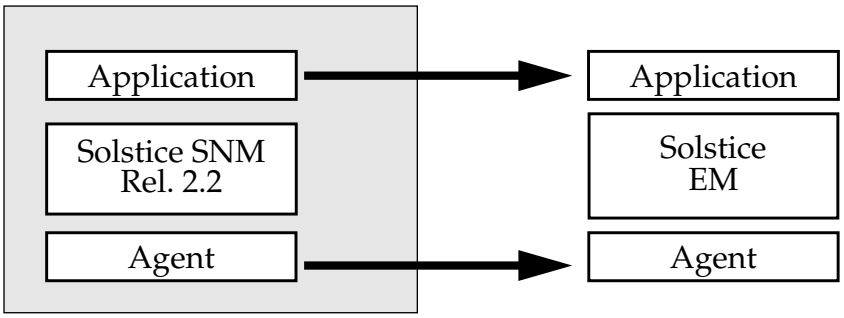

Figure 1 Single Platform Upgrade with Solstice SunNet Manager Compatibility

In addition, since Solstice EM supports the Solstice SunNet Manager RPC-based agents, no additional effort is required to install and test remote agents.

SunSoft Management Platform Product Line

The following table summarizes the similarities and differences between the various SunSoft management platforms.

	Solstice SunNet Manager	**Solstice SunNet Manager Solstice Cooperative Consoles**	**Solstice EM**
Multiprotocol	✓	✓	✓
Multi-User		✓	✓
Inter-user Security			✓
Object Oriented			✓
Multi-Vendor Server Interconnection			✓

Table 1 SunSoft Management Platform Summary

The rest of this paper describes Solstice EM and the concurrent initiatives underway by SunSoft and its partners in support of the vision of Solstice Cooperative Management.

Solstice EM is described in the following terms:

- Architectural Philosophy
- Architectural Overview
- Basic Tools
- Development Environment

The concurrent initiatives focus on advancements in system management.

Solstice™ Enterprise Manager™

Solstice Enterprise Manager is a distributed, secure, multi-user management platform. The platform is made up of cooperating components, such as applications, libraries, and databases, each with specific functional tasks. Solstice EM addresses the following high-level requirements:

- Enable the flexible distribution and configuration of management authority and responsibility
- Separate management information from management applications and agents
- Provide a single interface to management information and management activities
- Provide programming facilities for integrating and developing network and system management applications and agents
- Support multiple management protocols simultaneously by the same management environment
- Support both international and de facto management standards

Architectural Philosophy of Solstice Enterprise Manager

The course of activities conducted by a management application and an agent of a managed resource are symmetrical. The activity of a management application is to specify the desired state of the resource to the agent of the resource.

In comparison, the agent of a managed resource specifies the actual state of the managed resource to the management application.

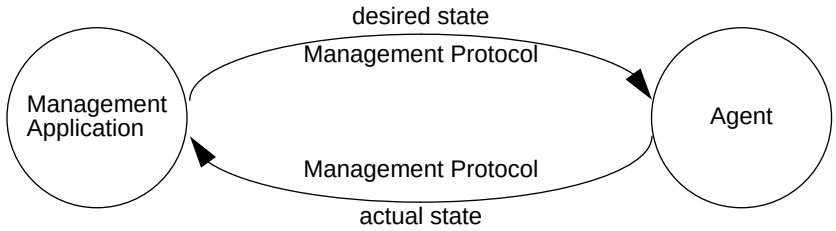

Figure 2 Communication Roles Between Management Applications and Agents

Solstice EM mediates the interaction between a management application and an agent of a managed resource. The Solstice EM architecture provides the following framework:

- Interposes software models of managed resources between management applications and agents of managed resources
- Provides a single interface to these resource instances
- Provides a set of management functions, such as monitoring and examining or changing attributes, used to coordinate management on the managed resources

Each software model of a managed resource contains information about the attributes of the real resource, and the operations to examine or manipulate those attributes. Management applications use the attributes and operations to direct the management activities that can be imposed upon a managed resource.

The single interface presented to management applications insulates the applications from the details of the underlying management protocol used to manipulate the real resource. For example, the real resource may be manipulated through SNMP or CMIP while the communication details are not exposed to the application.

The set of management functions in Solstice EM are facilities such as a monitoring function, a logging function, or a combination of functions, that all applications may invoke to help them to conduct their management tasks.

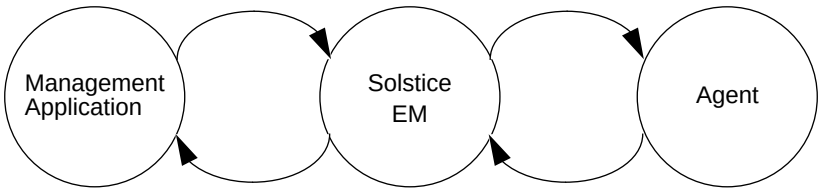

Figure 3 Relationship of Solstice EM and Management Applications and Agents

When assuming the manager role, management applications issue directives to Solstice EM through a well-defined interface. In turn, components within Solstice EM assume the manager role and use the software models of the managed resources to direct the changes in the real resource.

Similarly, an agent of a managed resource monitors the real resource for changes. When changes occur, the agent issues protocol-specific directives (i.e. traps, notifications, or exceptions) to Solstice EM. Solstice EM then modifies the software model of the managed resource to reflect the state of the real resource.

Architectural Overview of Solstice Enterprise Manager

The Solstice EM management environment consists of two architectural components, each with a particular function and separated from the others by well-defined interfaces through which all interaction is mediated. The two architectural components are the *Management Information Server*, and *Management Protocol Adapters*.

Management applications (e.g. Solstice EM core applications and third-party applications) direct management activities via the Portable Management Interface (PMI) to a Management Information Server (MIS).

The MIS is a repository of management data and management functions, both static and dynamic. The MIS contains *resource instances*, software representations of the managed resources, and the operations that can be performed upon them.

A *Management Protocol Adapter* (MPA) communicates to and from specific management protocols outside of Solstice EM.

For example, in an environment where both the CMIP and the SNMP management protocols are used, there would be at least one MPA for the CMIP managed agents and at least one MPA for the SNMP managed agents.

Although there are potentially many interfaces to Solstice EM, there is only one interface required by the architecture. The fundamental interface is the ***Portable Management Interface*** (PMI). The PMI functions defines a management protocol, management services, and the transport mechanism for all components of the Solstice EM platform. The PMI can be characterized as CMIS messages transported over TCP/IP or OSI.

The diagram below is an architectural schematic of Solstice EM showing several applications using a single Management Information Server to communicate management information to and from a variety of management protocols.

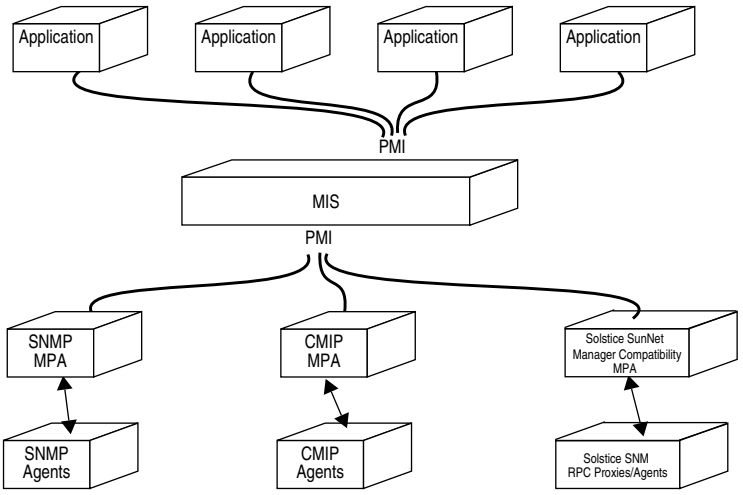

Figure 4 Schematic diagram of the architectural components of Solstice EM

The Solstice EM environment uses a common communication infrastructure and interface to communicate among all of the components. Distribution of the system can occur at any point where the PMI interface is exposed. The natural points for distribution are between applications and the MIS, the MIS and an MPA, and an MPA and agents of managed resources.

Basic Tools

Previous sections described why management platforms must be off-the-shelf, plug-and-play tools requiring minimum configuration and customization. Solstice EM provides many tools.

All Solstice EM basic tools (i.e. core applications) present operators with a common user-interface theme. While every tool may have functionality which causes it to vary the interface, the general look and feel of each tool is the same. Furthermore, an application Style Guide is available, specifying how applications should look, and how they should be integrated into Solstice EM. A Programmers Guide is also available to assist people in utilizing the application programming interfaces (APIs) provided to write new applications. These guides encourage third party applications to have the same look and feel as the tools provided with Solstice EM.

The Solstice EM basic tools have both graphical user interfaces (GUIs) and command line interfaces. GUIs are well suited for novice operators. Command line interfaces maximize expert operator productivity, and also provide additional programming flexibility for value-added applications.

The following sections describe the basic tools and their user interfaces.

Standards Conformance: Motif® and COSE

All Solstice EM basic tools are Motif 1.2 compliant.

The COSE initiative (Common Operating System Environment) specified the CDE (Common Desktop Environment) to provide a standard look, feel, and interface for UNIX® applications. SunSoft intends that Solstice EM will be able to run in CDE environments. Future releases of Solstice EM Basic Tools will be written to the CDE specifications, and use CDE services such as PRINT and HELP, as the specifications become standards.

User Model

In the Solstice EM user model, icons represent managed resources on the network, or software objects (e.g predefined request templates, views). Operators manage network and system resources by manipulating these icons. The primary tool for manipulation is the mouse. The basic categories of

manipulation are activating (clicking), dragging (for drag and drop activities), and selecting (for subsequent operations).The icons can be supplemented by text for presenting the user with more information.

Solstice EM applications also have a common understanding of objects so that one can drag an icon from one application to another. The application can then initiate some activity based on the object being dragged and dropped.

Control Panel

The **Control Panel** is the first application users see when starting Solstice EM. It is an application launcher for Solstice EM core applications and third-party applications. Users start applications by clicking on a icon. The Control Panel allows administrators to configure which applications are available to a specific user. The following table describes the Control Panel.

Control Panel Components	Description
Central Window	The Central Window consists of three areas: a Menu Bar, an Application Icon Bar, and a Status Area. The Menu Bar is the standard Motif offering used to bring up the various dialogs/windows of the Control Panel. The Application Icon Bar provides a graphic representation, icon, of each application. Clicking on an icon starts the application. The Status Area displays overall information about the MIS the Control Panel is currently connected to. It also provides an area for error messages to be displayed when starting applications.
User Configuration	User Configuration Window provides users with GUI controls for setting the user configurable options for overall MIS interaction. Each application with configuration options will present the user with a similar window. For example, the User Configuration Window enables administrators to determine which MIS to connect to by default at start up.

Table 2 Control Panel Components

Control Panel Components	Description
Platform Connection	The Platform Connection Window presents the user with a list of MISs known to the Control Panel. It also highlights the current MIS. When applications are launched from the Control Panel, they try to connect to the Control Panel's notion of the current MIS. By changing the current MIS in this window, the user changes the MIS that subsequent applications will try to connect to. The applications already running, remain connected to the previous MIS. Only applications started after the change will connect to the new MIS.
Import File	The Import File Window will be used to select the type of file to load into the system: GDMO Description, SNMP MIB, Solstice SunNet Manager Schema file.
Tool Bar Configuration	The Tool Bar Configuration Window allows users, with appropriate permissions, to configure which tools appear in the tool bar. It is the mechanism for adding, deleting, and determining the order, of icons in the Tool Bar.

Table 2 Control Panel Components

Discover

Discover automatically populates Solstice EM with objects representing managed resources. It also monitors additions/subtractions of managed resources, and updates the Solstice EM MIS accordingly. Discover has the following functionality:

- Discovery of IP and OSI Networks
- Discover both SNMP and CMIP devices
- Graphical user interface

For IP networks, Discover works by building a hierarchical linked list structure of the network topology. There is one entry in the list for each Class A,B, or C network that is discovered. Each of these network entries contains a list of subnets, a list of connected networks, and a list of hosts directly connected to this network if there is no subnetting. Each subnet list contains pointers to a list of connected subnets, a list of routers connecting those subnets, and a list of hosts directly connected to that subnet.

The ***Monitor*** portion of Discover reads from the Solstice EM database and constructs a list of views, and hosts contained in those views. It then proceeds to ping the address space for each view/subnet checking for new hosts. Monitor will also flag hosts that are in the databases but are not responding.

Similar functionality can be applied to the OSI network structure. The Discover modules get system and routing information from OSI nodes.

MOVE: Modelled Object Viewer/Editor

MOVE (Modelled Object Viewer Editor) is the central tool for monitoring and controlling network and system resources. It is responsible for providing an interface to:

- View resources
- Monitor resources
- Launch Requests against managed resources
- Receive reports/events/traps
- Provide user-defined filtering action on reports/events/traps

MOVE consists of the following components:

- Viewer
- Configuration Window
- View Navigator
- Object Inspector
- Overlay Window
- Object Pallet
- Search & Retrieval
- Graphic Editor
- Background Image File Chooser

Viewer

The ***Viewer*** presents the contents of a single view to the user. Each Viewer window (there can be numerous) can only look at a one view at a time. The contents of the view are represented by icons. Textual descriptions (names, IP address, etc.) can optionally be displayed in addition to the icons. The Viewer is capable of displaying connectivity information (links, busses, etc.). The connectivity information is represented by objects.

Configuration Window

The ***Configuration Window*** lets users set the user configurable options of MOVE. It is started from the Viewer window.

View Navigator

The ***View Navigator*** serves three functions:

i. Allows the user to easily see all of the views available via an icon with optional text descriptions underneath

ii. Allows the user to start up Viewers to inspect the contents of a view (by clicking on the icon representation of a view)

iii. Allows the user to connect views to form a hierarchical layout by a simple connect mechanism and/or drag & drop mechanism.

The ***View Navigator*** consists of a central canvas (used for displaying view icons) surrounded by vertical and horizontal scroll bars. It has GUI controls for constraining which views (by permission, type, etc.) get displayed.

Object Inspector

The ***Object Inspector*** provides the user with detailed information about an object. Information available in the Object Inspector includes: any attributes (and their values if they reside in the MIS), any MIB variables, any requests currently running against the object, the view names in which this object appears, etc. Since this information will be greater than what could be displayed clearly in a single window, the Object Inspector presents many "panes", each containing information related to a specific area (e.g. request data, or agent-data, etc.).

The Object Inspector also presents GET/SET buttons which allow the user to GET attribute values that not currently stored in the Management Information Server (MIS). The SET feature allows users to set values of attributes represented by object instances within, and outside of, the MIS. The Object Inspector shields operators from the protocol specific syntax and semantics when retrieving information and initiating actions on managed resources.

Object Palette

The ***Object Palette*** contains icons (or text or both icons and text) representing the types of objects a user can add to a view. Dragging a icon from the Object Palette to a Viewer window adds the a new object to the

view. If new object types (classes that can be represented as objects in views) are added to the MIS, they automatically appear in the Object Palette. The MOVE configuration window allows the user to control which objects appear in the Object Palette.

Search & Retrieval

The ***Search & Retrieval*** window presents the user with an easy way to interrogate the MIS to find groups of objects (based on type, class information, specific attribute values, etc.). A search generates a list of matching objects represented by a collection of icons. These icons can then be acted on like any other icons (i.e. launch Requests).

Graphic Editor

The ***Graphic Editor*** provides the user with basic tools for drawing in the canvas. This includes making textual annotations next to icons, doing basic layout of objects, and for drawing connections and links between objects. It consists of a small palette of icon buttons, each representing a separate function.

Background Image File Chooser

The ***Background Image File Chooser*** allows the user to select files to use as the background image for a particular view. Solstice EM supports the following graphic formats for background images:

- GIF (Graphic Interchange Format)
- Sun® raster file
- xpm
- JPEG

Log Editor

The **Log Editor** enables the user to work with logging descriptors. A logging descriptor is an object which defines the conditions under which information should be written to a log and also how it should be written. The Log Editor provides the following functionality:

- Allows the user to enable logging descriptors from predefined logging descriptor templates

- Allows the user to create custom logging descriptor templates which can subsequently be used to enable logging descriptors
- Allows the user to browse and to stop/resume the logging descriptors currently enabled
- Allows the user to browse and modify the logging templates

Grapher and Browser

The ***Grapher and Browser*** applications help operators analyze log data. This is done by presenting the log data in various ways including data reports and graphs.

Alarm Summary

The ***Alarm Summary*** presents operators with a concise view of traps and notifications generated by managed resources. Like all Solstice EM core applications, multiple instances of the Alarm Summary can run simultaneously. These instances can display alarms based on different sorting criteria (e.g. severity, time, resource) to support various management policies.

Request Editor

The ***Request Editor*** enables operators to develop applications to gather values for selected attributes over time, and to monitor changes in attribute values based on predefined thresholds. These applications are called *Requests*. The results of the Requests can be displayed, logged, or cause an action such as a visual indicator or executing a script.

The Request Editor is responsible for manipulating (i.e. creating, modifying, and deleting) *Request Templates* which form the basis for Requests. The Request Editor consists of a primary window listing Predefined Requests. Operators can select a Request to modify (or use as the basis for a new Request), or operators can enter the Template Editor to develop a new Request. The Template Editor consists of a *state machine* diagram which defines the conditions and transitions that make up the Request.

The Request Editor enables customers to define *conditions*, combinations of polled information, events, or traps, to determine when state changes, transitions, occur. These conditions can be simple (i.e. when a counter equals N, the state changes from *up* to *down*).

Or the conditions can be more complex. For example, assume that Solstice EM momentarily loses communications with a workstation. With robust communications, the link is reestablished and no one notices the momentary outage. Even though Solstice EM will know that the brief interruptions occurred (through an event or trap), the customer may not want to conclude that the workstation is down. The customer could define a condition where a specific number of occurrences of the event/trap describing the outage put the workstation into a *missed*, rather than *down* state. In addition, the customer could define another condition where once the workstation is in the *missed* state, and a specific number of occurrences of the event/trap have happen, then the workstation could in fact then be placed in a *dow*n state.

Complex conditions can also involve comparing event streams from two or more resources to derive a single conclusion. For example, events describing an increasing bit error rate from serial link could be compared with an event stream describing an increasing retransmission rate to lead an operator to determine that diminishing response time can be resolved with a telephone call to the transmission service provider.

The Request Editor enables customers to use arithmetic, relational, equality, logical, and address operators to define conditions.

Object Information Logger (OIL)

The ***Object Information Logger (OIL)*** logs all of configuration and state change information, and stores them into an optional relational database. By transferring information to the optional relational database, OIL enables customers to use standard tools such as SQL to access and retrieve management information.

The MIS uses log objects to record notifications which are messages reporting changes in the state of any managed object. OIL creates a set of log objects in the platform in order to receive notifications whenever there is an event of the following types:

- Creation of any object instance
- Deletion of any object instance
- Any attribute value change of a particular object instance
- Any state change notification received from an agent of a particular managed object instance

This management information can be used for trend analysis, and as an audit trail to determine changes to the network.

Remote Application Execution

The core applications described above, and any tools that use the Solstice EM *Portable Management Interface* (PMI) can run remotely from the *Management Information Server* (MIS). This provides both operational and financial benefits.

From an operational perspective, the ability to run applications remotely means that tools can be provided in locations where they are needed. Solstice EM enables tools and applications to be located to match the management task and the skill-set of the operators. For example, when a technician is paged to address a problem, the technician would not have to go to the operations center to access the required tools. Similarly, in off hours, senior analysts would not have to come into the operations center to begin resolving an escalation. While running on a remote processor, the tools would have access to the same management information available to the operations center staff.

The financial benefits from enabling applications to operate remotely come from bandwidth savings, training, and hardware. Other management platforms often use X Windows technology to redirect displays to remote devices. While addressing the need to extend the availability of the tools, management platforms and applications that rely on X Windows for remote distribution have proven to be bandwidth intensive. This is costly when using wide area facilities such as leased lines. By sending just management information rather than graphical user interface overhead, Solstice EM is bandwidth efficient.

By enabling applications to run remotely from the management information, Solstice EM can also help reduce training requirements. Operators in the remote sites can concentrate on tools, and not be concerned with the configuration of the management platform (i.e. administering data stores and agent connectivity). In addition, hardware requirements decrease since less processor capacity and disk space are required when running just an application, and not both the application and platform.

Multi-user Capabilities and Security Services

The ability to run applications remotely is just one aspect of the ability of Solstice EM to support multiple users.

Unlike most of today's platforms which are single-user (or single operator and multiple read-only users), Solstice EM is designed to be multi-user.

The Solstice EM multi-user capabilities revolve around the ability to give all components of the management solution (e.g. operators, tools, applications, other management systems) access to consistent management data. This enables management tasks to be divided across organizations and geography with confidence that all users will see a same view of the management information and status. This is particularly useful in fault management scenarios where cooperation among staff members leads to quicker problem resolution.

However, providing multi-user access creates the need to be able control access to management information. Security policies must be developed and enforced to control who can make changes to specific resources.

The Solstice EM security model uses the notion of an Access Control List (ACL) to govern access to specific management information elements and the Management Information Server (MIS) itself. These ACLs enable the Solstice EM administrators to create access control policies that suit their security and restriction needs.

The security model has two levels, each with a generic set of permissions:

i. Platform: permits or denies connections to an MIS
ii. Object Instance: permits or denies creation or deletion of object instances

Each MIS has available to it a list of allowed users. (*Users* are defined by UNIX log-in name/ID.) When a connection to the MIS is established, the user's application communicates information about the user to the MIS. That information is used by the MIS to determine whether or not the connection is permitted or denied.

Once a connection has been established, subsequent accesses to object instances are compared against an ACL defined for each object. The object's ACL will specify whether the requested operation is permitted or denied. Each ACL specifies positive permission. If an ACL exists, the user and the requested operation must explicitly be declared to be allowed.

An ACL may be used by more than one object instance. An administrator may create a single ACL with a set of users and permissions which represent a management domain or group.

ACLs specify user name and permissions consisting of Create, Delete, Get, Set, Action, and Event, and an optional default permission.

By default, an object instance has no ACL. If an ACL is required, the creator must assign an ACL to that object instance. An object instance specifies the ACL used to govern access. If an object instance does not specify an ACL, access is open.

Management Information Server (MIS)

The ***Management Information Server (MIS)*** is the key component of Solstice EM which enables operators and applications deal with the scale and complexity of today's networked systems environment. The Solstice EM MIS provides an extensible set of management functions and an environment for implementing and manipulating managed objects—software models of managed resources. The MIS is a repository of management data and management functions, both static and dynamic. The MIS contains *resource instances*, software representations of the managed resources, and the operations that can be performed upon them.

Management applications direct activities upon managed resources through requests transmitted to the MIS. Subsequently, the MIS returns the results of the commands to the application via the Portable Management Interface (PMI).

The information associated with a request consists of an operation, a target object, and parameters. These requests provide the following services to applications:

- Creating and deleting managed objects
- Searching the Management Information Base for a managed object that fits a certain criteria
- Fetching the value of a managed object's attribute

- Comparing the values of a managed object's attributes
- Changing a value of an attribute, invoking a special function implemented by a managed object
- Sending or receiving notifications of asynchronous events

Through these fundamental requests additional operations such as polling may be directed to managed resources by manipulating parameters governing these additional operations. A management application may use the MIS functions to specify certain conditions under which an operation is to be processed, or to record the result of an operation.

The following sections describe the MIS Modelling Environment, the management functions of the MIS, and the ability to distribute MISs.

MIS Modelling Environment

The MIS ***Modelling Environment*** provides a mechanism for implementing and manipulating software models of managed resources. The Modelling Environment is controlled by the Object Manager component of the MIS.

The MIS Modelling Environment contains information about a variety of resources. Each of these resources is characterized by an *object class*. An object class is a template describing the conceptual format of the information stored in an object instance created from the object class. Each of these resources is represented by an object instance which is a specific example of an object class. For example, there may exist an object class called `Router` which describes the manageable characteristics of all routers, and an object instance called `MyRouter` which is a specific representation of a router and characterized by the object class `Router`.

The definition language used to describe the objects internally is the "Guidelines for the Definition of Managed Objects" (GDMO) outlined in the ISO/IEC 10165-4 standard.

An object class defines the structure of the management information of a managed resource. Each object class is specially constructed to represent a particular kind of managed resource. Some object classes are more broad-based in their applicability, and define information that may be found in a variety of managed resources. For example, an object class may declare an attribute that contains the amount of time that the resource has been in operation. Such an object class would be used in other object class definitions for workstations or PBXs, applications or MIS management operations. In contrast, an object class

may declare the number of concurrently executing topology applications. Such an object may not have broad-based application and would be used in only a few cases.

An object class *attribute* is a specific property with a name, a value, and a type for that value. For example, `sysUptime` and `sysContact` are attributes of the SNMP MIB-II object class. Each has a name, a type, and potentially a value.

A *method* is a function that can be applied to resource instances of an object class. For example, consider the `View` object class. This class implements a method used to add a new resource instance to the existing list of resource instances that belong to the particular `View`. This provides a valuable mechanism for automation.

Definitions of object classes are created via the Modelling Environment. The organization of these object classes into a hierarchical arrangement is the Solstice EM MIS ***Object Model.***

Object Model

The ***MIS Object Model*** maintains a tree-structured naming scheme for all of the managed resources. Every managed resource is uniquely identified by its Fully Distinguished Name (FDN). The FDN is the path through the Object Model to the object instance. The following is an example of a Fully Distinguished Name:

```
/systemId=name:"flatline"/subsystemId="G2r1Kernel"/applicationID=2
```

The MIS Object Model conforms to rules outlined in the Network Management Forum, OMNIPoint™ 1.0. The Object Model conforms to two ISO/IEC standards called out in OMNIPoint 1.0:

- ISO/IEC 10165-1: Management Information Model
- ISO/IEC 10165-2: Definition of Management Information

The Object Model is divided into two parts: standards-based objects and SunSoft Solstice EM objects. The standards-based object definitions are specified in the Network Management Forum OMNIPoint 1.0, and by various standards bodies (i.e. ISO, IETF). The SunSoft Solstice EM objects define and control behavior of the Management Information Server (MIS).

The Object Model is extensible. New object classes can be added as descendents of existing object class definitions, or as entirely new object class definitions. The Object Model can be extended using Solstice EM administration tools. These tools enable GDMO, Internet™ Concise MIB and ASN.1 object definitions, and Solstice SunNet Manager Schema to be loaded into the Management Information Server.

One of the key aspects of the Object Model is how it provides protocol translation to support both the OSI and Internet management models. Where a managed object is reported by an SNMP agent, the Object Model creates a *proxy*. Although the proxy accepts PMI (Portable Management Interface) requests and returns PMI responses, the proxy uses SNMP to communicate with the remote SNMP agent. Because the PMI provides facilities that are not directly supported in SNMP, the proxy is considerably more than a translator. For example, to support OMNIPoint scoping, the proxy may have to generate multiple requests and coordinate multiple responses. To handle some OMNIPoint notifications, the proxy may have to poll the SNMP agent.

Persistent Storage & Information Management

The Object Model, and the object instances representing the managed resources, are stored in a persistent storage mechanism. Persistent objects are stored independently of the workstation's dynamic memory, in a medium that makes the object available without having to rely on keeping it continuously in memory. This simplifies service restoration in the event of an upgrade or service affecting catastrophe. The storage mechanism is private, fast database called `TDBM`. `TDBM` is a transaction processing database based on the `dbm` database library widely used in UNIX environments. Access to the Object Model and the object instances is provided through the Solstice EM *Portable Management Interface (PMI)*.

The Object Model provides information management for two types of data: configuration data, and log data. Configuration data includes all the network elements and devices, and their relationships which comprise the management domain. Log data are the events or reports about managed resources received from management agents.

An optional relational database management system (RDBMS) is available to store log data. Log data can then be queried and retrieved using RDBMS access tools such as SQL. The Object Information Logger (OIL) tool automatically formats and forward the log data into the optional RDBMS.

The Solstice EM Portable Management Interface (PMI) API will always be available to query and retrieve configuration and log data. The PMI will coexist with an SQL interface when the optional RDBMS is installed.

MIS Management Functions

A Management Information Server (MIS) constitutes an extensible set of management functions. The functions are invoked via a single interface provided by the MIS Service Manager and communicated through the Portable Management Interface (PMI).

Each management function presents an interface which is used by the Service Manager, and by other management functions, to make use of each other's capabilities. An MIS presents the common functions for management and insulates management applications from locating and administering the management function.

Many of the management functions provided by the Solstice EM MIS correspond to the OSI management functions. The MIS may also provide functions that are outside of the scope of the OSI management function model, or not explicitly described by the OSI model. The following diagrams described the MIS management functions and their relationships to the OSI management model.

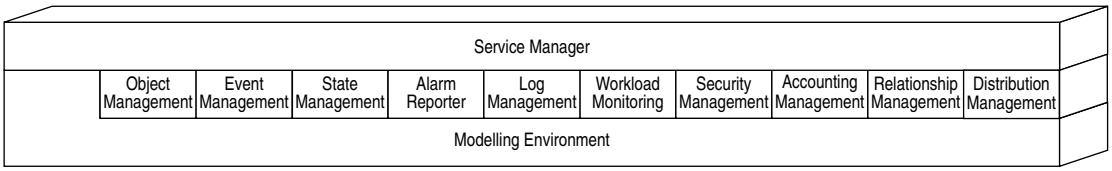

Figure 5 A schematic diagram of an MIS

Service Manager

The ***Service Manager*** function provides a single connection point for all external components using the MIS to issue management directives via the PMI.

The Service Manager establishes and maintains associations with management applications through the PMI. Associations are initiated by management applications and destroyed when no longer needed or no longer usable.

It also permits or denies session creation based upon user name and password. Both the user name and password are ascii strings.

The Service Manager dispatches management directives received via the PMI to the appropriate management function within the MIS. In addition, management functions within the MIS may transmit messages to other management functions either directly in the local MIS, or in a different MIS through its Service Manager.

Object Management

The ***Object Management*** function provides services for examining and manipulating object instances within the Modelling Environment.

Use falls into two categories: 1) Using the `create`, `delete`, `get`, `set`, and `action` directives to manipulate an object instance; 2) To report when the following operations occur: object instance creation; object instance deletion; attribute value changes.

Event Management

The ***Event Management*** function enables applications to receive notifications from managed resources. It monitors of all Event Forwarding Discriminators. It also forwards notifications received by the MIS to other Solstice EM components.

State Management

The ***State Management*** function provides generic definitions for querying and changing the management state of an object instance. It also reports changes in the management state of an object instance.

State Management is not implemented as a separate MIS management function. The State Management function is accomplished by either inferring the state of a object from the values of its attributes, or by specifying an attribute in the object instance to represent the state of the object.

Alarm Reporting

The ***Alarm Reporting*** function enables management applications to notify other applications of potentially abnormal conditions.

Alarm Reporting is not implemented as a separate MIS management function. The Alarm Management function is accomplished by inferring an alarm from the values of a object instance's attributes, or by specifying an attribute in the object instance to represent the alarm state of the managed object.

Log Management

The ***Log Management*** function provides a mechanism to control the logging of information such as event, data, and trap reports. The Log Management function models the operation of the log, but the actual logging activity is accomplished by applications.

Workload Monitoring

The ***Workload Monitoring*** function is responsible for monitoring the values of object instance attribute in reference to threshold values. An attribute value is continually or regularly used to determine a relation between the value and its reference. If the relation becomes true, a notification is issued.

Security and Access Management

The ***Security & Access Management*** function permits adding and deleting users. The information kept for each user is name and password and is obtained from an external directory service.

Distribution Management

The ***Distribution Management*** function supplies the mechanism for an MIS to assume the manager role in relation to another component such as another MIS or an Management Protocol Adapter (MPA).

MIS Distribution

To manage a large number of resources, or to increase system availability, a Solstice EM environment may have more than one Management Information Server (MIS). Distribution of management information, control, and authority is accomplished by distributing multiple MISs. The MIS-to-MIS communication mechanism enables operators and applications to act on consistent management information, without regard to the location of the application. Each MIS then acts as the router of management information that is served by the MIS.

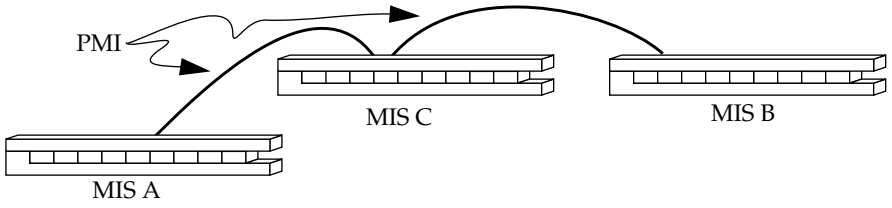

Figure 6 Three cooperating, distributed Management Information Servers

There is no independent MIS-to-MIS interface or communication mechanism. In all cases management information is transmitted through the Portable Management Interface (PMI). In an environment with two or more MISs needing to transfer information or act on behalf of another, one MIS assumes the role of a manager when requesting information from another MIS.

MIS-to-MIS communication is accomplished by adding the name of a MIS to the Fully Distinguished Name (FDN) table of the current MIS. This is analogous to an Network File System (NFS®) *mount* where a resource, in this case another MIS, is attached to a file system hierarchy, the original MIS.

As mentioned above, components of Solstice EM can be distributed to help manage a large number of resources and to increase system availability. Additional components of Solstice EM can be distributed to reduce the cost of gathering information from managed resources. Distributing the information gathering process reduces costs by minimizing the bandwidth required to poll for information, and filtering extraneous events closer to the source.

Distributed Data Collection and Filtering Capabilities

SunSoft pioneered the concept of distributed management protocol engines, called ***proxies***. Proxies leveraged Sun's ONC™ RPC technology to relieve the management station and intermediate routers of polling load. Proxies also make management of multiple protocols a basic part of the system's architecture.

Solstice EM extends the concept of the Solstice SunNet Manager proxy by leveraging the transport capabilities of the Portable Management Interface (PMI). The PMI uses CMIS messages, rather than RPC, to allow a Management Protocol Adapter (MPA) to communicate with an MIS.

Management Protocol Adapter (MPA)

The ***Management Protocol Adapter*** (MPA) component of Solstice EM provides access to management protocols for the purpose of communicating with agents of managed objects. There is at least one MPA for each management protocol. Via an MPA, the MIS can access a managed object without having the need to know about the details of the protocol. Thus, the MIS can concentrate on the management of object information and the provision of services to the user applications.

When an MPA receives a message from the MIS, the MPA is responsible for translating an PMI message into the proper format dictated by the protocol with which it interfaces, and delivering the protocol data unit (PDU) to the proper agent. The translation includes:

- Encoding the PMI message
- Formatting the PMI message into a management protocol PDU.

When the MPA receives a PDU from an agent, it is responsible for converting the PDU into an PMI message and delivering the message to the proper MIS. The PDU conversion includes:

- Decoding the PDU
- Formatting the decoded PDU into an PMI message

In addition to providing access to the management protocol, each MPA also supports data reporting and simple threshold monitoring on managed objects. This service reduces the traffic on the network.

An MPA can be instructed to store the collected data and forward the data only when the MIS asks for them. Because of this feature, there is a policy governed by the MPA regarding how many samples the MPA can store and what action for the MPA to take when the storage is full.

Some customers may choose to install multiple copies of the Solstice EM MIS to distribute the Event Management function. This enables complex thresholding functionality to be distributed throughout the network.

Solstice SunNet Manager Agent and Daemon Support

To protect customers' investments in management agents, a Solstice EM ***Solstice SunNet Manager Compatibility*** MPA supports both the Solstice SunNet Manager proxy agents, and the Solstice SunNet Manager ***SNMP Trap*** daemon. The Solstice SunNet Manager SNMP Trap daemon provides the ability to filter events and traps close to the source in an error corrected fashion.

Application Development Environment

The following sections describe how Solstice EM functions as a runtime environment for third-party applications. The Solstice EM Development Environment provides powerful interfaces and tools that allow a new generation of secure, integrated management applications to be developed.

Solstice EM uses object classes and methods to enable applications to retrieve and store data concerning managed resources. These same mechanisms can also be used to receive notification of events that occur related to managed resources. The Portable Management Interface (PMI) provides these mechanisms.

The PMI is one of two interfaces available to applications developers. The Solstice EM application programming interfaces (APIs) are:

- Platform Management Interface (PMI)
- Solstice SunNet Manager Compatible API

Portable Management Interface (PMI)

The Solstice EM ***Portable Management Interface (PMI)*** presents applications with object classes and methods to provide access to a wide range of management information in a very flexible environment. The PMI is implemented in C++, and later with C bindings. The PMI provides the following services:

- Initialization, including establishment of a distributed message passing interface to the MIS
- Event subscription and propagation
- Remote caching and cache control
- Local object cache management for applications
- Encoding/decoding of parameters into ASN.1
- Encoding and encapsulation of data into a format (message class) passed to the MIS
- Access to security services

PMI Object Classes

The following table describes object classes implemented in the PMI to shield developers from the details of the MIS.

PMI Object Name	Description
Platform	An instance of the `Platform` class represents a potential or actual connection to a particular MIS, along with all the implied semantics of the framework implemented by the MIS.
Image	An instance of the `Image` class is the local representation of an actual or potential object in an MIS's framework. Typically, an `Image` represents a managed object: a host, server, router, subnet (that is, the representation of a physical device) or a conceptual entity (a line, a queue, or some other aspect of network operation that can be represented as a managed object). An `Image` usually represents data as text, but also allows "raw" data to be passed in the form of `Morf`s. `Image`s also provide attribute-like access to object and attribute schema information. When representing an actual object, an `Image` can be synchronized with the object it represents either manually or automatically.
Album	An `Album` is a set of `Image`s representing a set of objects that are somehow related. An `Album` thus permits a collection of images to be treated as a whole, in much the same way as the MOVE or Request Editor services treat a view or a collection of managed objects. Like a mathematical set, an `Album` may be constructed either by rule or by enumeration. Certain operations can be performed on an `Album`, and thereby to each of the `Image`s in the `Album`. Like `Image`s, `Album`s may be synchronized either manually or automatically.
Morf and Syntax	A unit of data is represented by an instance of the `Morf` class (Mysterious Object Related to Framework). Each `Morf` contains an opaque, encoded value, along with the information the PMI needs in order to decode it. For each framework, there's a derived class that's able to manipulate that type of data in the context of the framework. An instance of the `Syntax` class represents a type. All framework-encoded data, whether part of an object or not, has a type. This type specifies, among other things, how to produce and understand human-readable representations of the data. The `Syntax` class is part of the PMI's implementation; application developers should not need to deal with `Syntax`es directly except when building `Morf`s.
AlbumImage	An `AlbumImage` is the representation of the state of an iterator that is progressing through either all of the `Image`s in an `Album` or all of the `Album`s containing an `Image`. This is analogous to a pointer into a linked list.

Figure 7 Portable Management Interface Object Classes

PMI Object Name	Description
CurrentEvent	A `CurrentEvent` is the representation of an event. When a callback is requested from the PMI. The PMI will call the callbacks back with a parameter, `CurrentEvent`. The application may then interrogate the `CurrentEvent` by various methods to find out what kind of event occurred, which `Image` or `Album` it relates to, etc.
Waiter	A `Waiter` is the representation of an ongoing asynchronous operation. The `Waiter` provides methods for cancelling the operation or awaiting its completion. The `Waiter` may also serve as the base class for asynchronous operations.
Coder	A `Coder` is the representation of a pair of methods for encoding and decoding values. The `Coder` class is a reference-counting wrapper around an inner class, called `CoderData`. Application developers may derive from `CoderData` to provide custom translation routines to the PMI. A `Coder` may be bound either to a `Syntax` or to an attribute name.

Figure 7 Portable Management Interface Object Classes

Manipulation Objects

The PMI simplifies the effort required to communicate with managed objects. Low-level routines would require atomic actions to obtain information such as location and encoding schemes. For example, CMIS messages exist to create and delete objects, to get and set attributes, and to perform various actions or to signal events. The PMI replaces these notions with the `Image` object class. Applications use the PMI to manipulate an `Image` object instance. An `Image` object instance acts as a surrogate for a local or remote object, and tracks where it is and what it's doing.

Naming Objects

Objects are named by starting from a known starting point in a tree, and traversing a map of containment relationships. The object's name is formed by concatenating the ***key*** for each of the steps in the traversal, with slashes between the components.

An object name that begins without a slash is interpreted relative to the container object within the `Platform` that represents the MIS. An object name beginning with a slash (`/`) is an absolute name, and refers to an object that is global. Names used in the PMI may be a superset of the OSI naming tree. Names that are not part of the OSI naming tree must not conflict with the OSI naming tree.

Within the application, any object may also have a nickname. Nicknames offer a convenient way to program in a more human readable format.

Relationships between Objects

The relationships between objects are represented using `Album` objects.

The `Album` contains a set of `Image`s. Membership of an `Image` in an `Album` can model the inclusion of objects in a mathematical set. The application can build up an `Album` by enumerating the `Image`s that it wants to include, much as a mathematician might build a set by enumeration.

Other relationships are of a directed nature. The PMI models these relationships as `Albums` built by a derivational rule rather than by enumeration. A simple derivation might form the set of objects that are *children* of another object. A more complex derivation might examine all the objects in another `Album`, select a subset of them, and for each object of that subset specify a relational attribute that defines a new set of objects.

Managing Notifications

An `Image` (or an `Album` of `Image`s) can alert an application when a change of state occurs. The PMI transmits the event to the application by invoking the callback function that the application registered earlier.

Managing Data Types

Each `Image` knows the attributes that a given object class supports. It also knows the type of each attribute. This enables an application to deal with the object on a purely textual basis.

The language in which textual data is expressed looks much like ASN.1 textual data. For example, scoping is indicated by curly braces, and choice names are delimited by a colon.

Applications occasionally need to deal with data apart from the definition of any particular object. While this can be done using text, it is sometimes more convenient to pass encoded data. The PMI supports the notion of typed encoded data. Such an object is called a `Morf`. A `Morf` is an attribute without any associated object. It knows its type and the associated syntax. A `Morf`'s value can be converted to and from textual representation.

Object Schema Management

An application program may discover various facts about the object class and its various attributes by using a `get_prop` or `get_attr_prop` function. These routines work much like the UNIX `getenv` call, but acquire their information from the MIS.

Solstice SunNet Manager Compatibility

To protect customers' investments in applications and agents, Solstice EM provides interfaces to support applications and agents written to the Solstice SunNet Manager 2.2 interfaces and libraries. Applications written to the Solstice SunNet Manager 2.2 interfaces and libraries will be compatible with Solstice EM.

The Solstice SunNet Manager 2.2 application library consists of manager services routines and database access routines. This library has been rewritten to use PMI (Portable Management Interface) function calls. For example, functions such as `netmgt_set_instance()` and `netmgt_set_argument()` are replaced by functions which store the set information into an appropriate structure, and send the final request as part of the `SetReq` class of the PMI.

In addition, the Solstice SunNet Manager Compatible MPA (Management Protocol Adapter) enables Solstice SunNet Manager 2.2 agents to communicate with the Solstice EM MIS. This MPA is a translator from Solstice SunNet Manager ONC RPC to Solstice EM Portable Management Interface (PMI).

The Solstice SunNet Manager application library has also been enhanced to manage events and traps in the Solstice EM model. The application library has been modified to construct an event discriminator. This event discriminator definition is used by the PMI to query Solstice SunNet Manager agents and determine when an event occurs. In this case, all requested events are handled by the Solstice EM MIS (Management Information Server). The Solstice SunNet Manager model assumes that agents handle both events and traps. In the Solstice EM model, Solstice SunNet Manager agents only handle traps.

Development Tools

The Solstice EM development environment provides a set of features and utilities to aid the developer in building Solstice EM applications. These include classes and macros useful for debugging and for exception handling, as well as several utilities useful for extending or inspecting the contents of the Object Model. The following features and utilities are described below:

- Debugging and Tracing facilities
- Exception Handling facilities
- Object Editor/Browser
- GDMO Parser
- ASN.1 Parser
- Concise MIB Parser
- Solstice SunNet Manager Schema-to-MIB Translator

Debugging and Tracing

The Solstice EM development environment provides a debugging and tracing facility that can be used when an application is in debug mode. This facility provides the following functionality:

- Print debugging information within an application
- Show flow-control information about functions within an application, including their return status and exceptions raised

Exception Handling

Solstice EM exception handling macros are designed to emulate the exception-handling mechanism defined in the C++ Annotated Reference Manual. The Solstice EM Development Environment provides exception handling macros similar C++ try and catch blocks that implement the throw function in various forms and manage exception propagation.

Object Editor/Browser

The Object Editor/Browser is a tool for inspecting and changing objects in the Solstice EM MIS Object Model. This tool can be used to determine the state of objects in an application. The Object Editor/Browser provides the following functionality:

- Browse the MIS Object Model

- Display existing collections of objects
- Display and edit CMIP and SNMP object attribute information
- Add objects to, and delete objects from, the MIS Object Model
- Create pointer objects that refer to remote managed resources

GDMO, ASN.1, & MIB Parsers and Schema-to-MIB Translator

Applications may need to extend the Solstice EM MIS with additional classes of managed objects. The Solstice EM Development Environment provides the following utilities for adding these descriptions to the MIS Object Model:

- The **GDMO Parser** processes a description written in GDMO format.
- The **ASN.1 Parser** processes a description written in ASN.1 type format. Such a description may be produced as output from the Concise MIB parser, or be supplied directly.
- The **Concise MIB Parser** translates a description written in the Internet™ MIB format into both GDMO format and ASN.1 format, and thus serves as a preprocessor for both the GDMO parser and the ASN.1 parse.
- The **Schema-to-MIB Translator** translates Solstice SunNet Manager Schema files to MIB format. The resulting MIB file must then be converted to GDMO format by the Concise MIB Parser, just as with any other MIB.

Concurrent Initiatives in System Management

In keeping with COSE and X/Open™, SunSoft plans to leverage the OMG framework for system and network management. It is important to note that there are few standards in system management today. SunSoft is working with systems companies, to do map-level and menu-level tool integration, preceding and somewhat in parallel with the framework-level integration described below.

Step 1: Solstice EM: GDMO Object Model, Routing

Solstice EM is Network Management Forum kernel compliant. As mentioned earlier, the Solstice EM Management Information Server (MIS) models managed resources in GDMO formats. Object calls to the MIS interface allow requests, responses and events to be routed between MISs following CMIS conventions. The following diagram describes how the MIS acts as a GDMO object router, similar to the way the CORBA ORB routes requests from IDL objects.

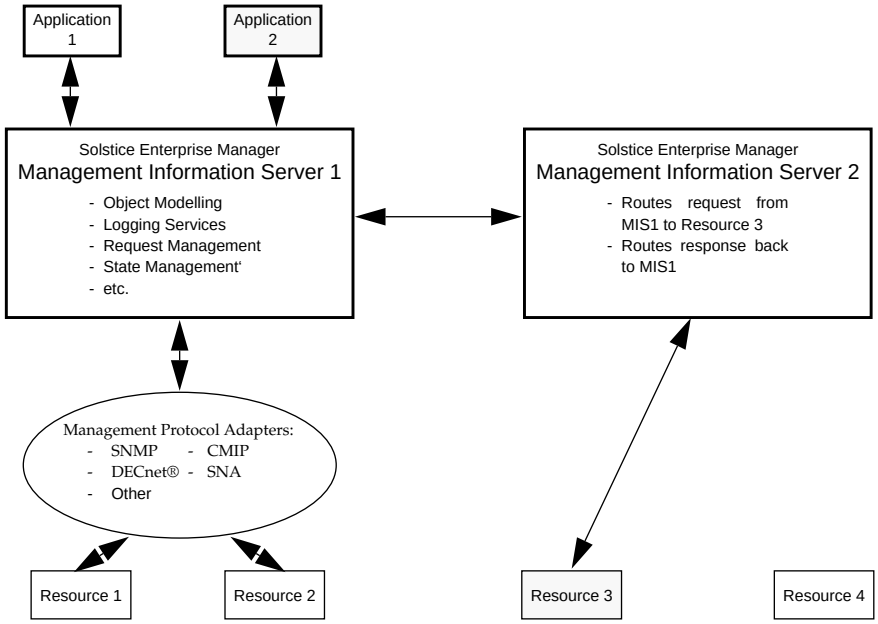

Figure 7 Step 1: Solstice Enterprise Manager as a GDMO Object Router

Step 2: Desktop Manage Taskforce (DMTF) Desktop Management Interface (DMI)

SunSoft was a charter member of the DMTF and secretariat of the technical committee. The functionality of the DMTF interfaces is critical to systems management.

Before the DMTF Desktop Management Interface (DMI), no standard mechanism existed to provide a single point of information contact for asset management, monitoring, and control that was operating system independent. While SunSoft continues work in this area, SunSoft is principally acting as an agent to promote adoption of the interfaces in the industry’s leading operating environments, including protocol mapping (e.g. SNMP to DMI). Widespread adoption will greatly accelerate the ability of generalized management platforms like Solstice EM to enable system management solution. This set of services may gain other interface definitions (e.g. IDL for CORBA compliance),

as the DMI standard evolves. The following diagram describes how the DMTF DMI provides the foundation for a standard extensible agent for system resource and asset management

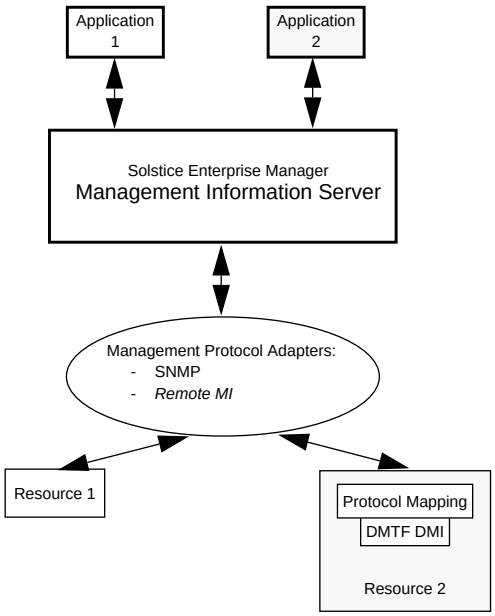

Figure 8 Step 2: Extensible Agents using DMTF Desktop Management Interface

Step 3: Solstice Enterprise Manager Manages IDL-defined Objects

Most future-oriented work in systems management is being defined in the OMG IDL language for access through CORBA-compliant object management brokers. Prior to completion of this work, SunSoft expects no standard access mechanisms between different systems companies.

If an object is defined by IDL, and Solstice EM is the toolset for monitoring/controlling it, two steps need completion:

- An object adaptor needs to be linked to the MIS in the form of a Management Protocol Adapter (MPA). The MPA architecture, and even the Adaptor part of its name, were designed with this in mind.
- The IDL description of the object needs to be compiled into the MIS, just as an SNMP MIB would be compiled.

At that point, any X/Open- or COSE-defined systems object can be managed from Solstice EM. The following diagram describes how the Solstice EM application will manage CORBA-compliant objects.

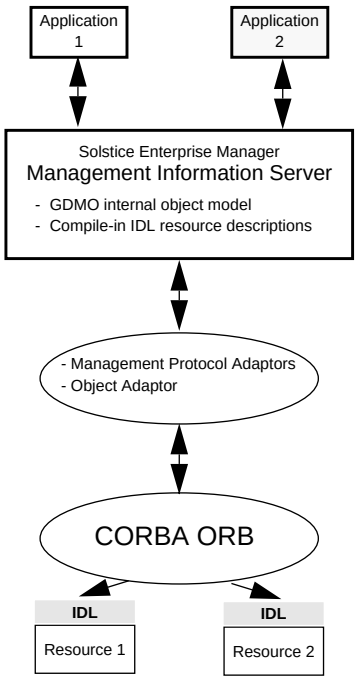

Figure 9 Step 3: Solstice Enterprise Manager Managing CORBA-compliant Object

Step 4: MISs Interface to ORBs

The principal interface to the Solstice EM Management Information Server (MIS), the PMI (Portable Management Interface), does not initially use CORBA ORBs. SunSoft is investigating redefining the PMI in IDL, in accordance with the X/Open work on management interfaces that will start with Tivoli's recent submission. The PMI-to-IDL mapping, coupled with the OMG COSS services and X/Open's work on mapping GDMO to IDL, the following scenario is expected:

- Applications act as object clients when requesting services from the MIS. The MIS acts as an object server for that transaction. Application developers can use the OMG approach and object calls to access management information.

- MISs still route object requests between themselves, but they may use CORBA transports for interoperability, in addition to the existing mechanisms (i.e. OMNIPoint).

SunSoft intends to support the X/Open interface work and not develop a proprietary encapsulation of PMI with IDL. The following diagram describes how an IDL interface could be mapped to the PMI.

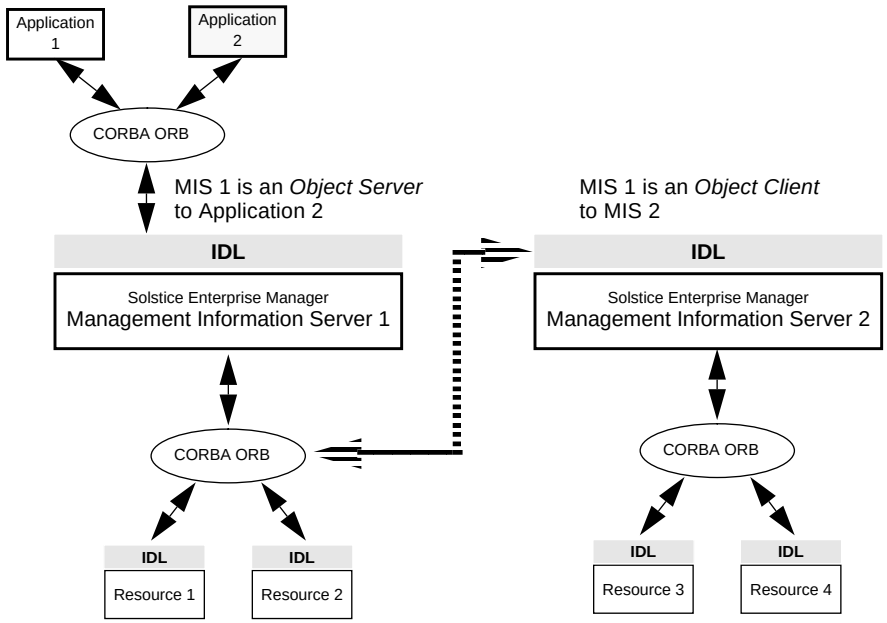

Figure 10 Step 4: IDL Interface for MIS Services

Step 5: MIS Services broken out with IDL interfaces

The MIS services are in many cases superclasses of work in progress in OMG (COSS) and X/Open (Sys Man). Over time, Solstice EM will use those service definitions or leverage them as subclasses. At that point, the MIS services that are now an enclosed bundle could also be available to developers as separate interfaces. The following diagram describes IDL interfaces for the MIS service components.

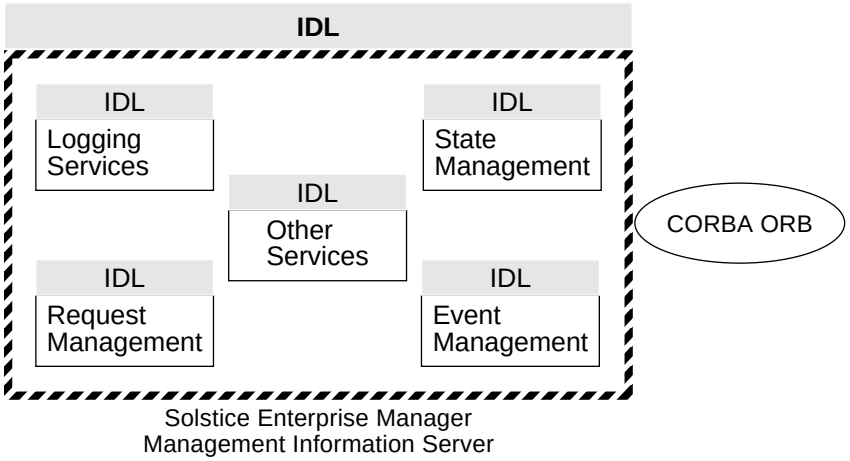

Figure 11 Step 5: IDL Interfaces for the MIS Service Components

Summary

The SunSoft vision of a Cooperative Management environment is one where technology and business relationships create links between multi vendor tools to simplify IT management.

Management platforms are the key infrastructure for creating a “management backbone” to integrate today’s islands of management tools. SunSoft is the leader in the management platform business.

Solstice EM advances management platform capabilities to address customer requirements for secure multi-user support, a comprehensive management data repository, and better basic tools.

The Solstice EM management framework can be deployed in a single-user, peer, hierarchical, or hybrid structure. This offers organizational flexibility to enable simple transfer of control, off-hours substitutions, or to implement a distributed or centralized arrangement.

SunSoft management platforms protect investments in agents, applications and training. Whether one is interested in upgrading from Solstice SunNet Manager to Solstice EM, or in using both and connecting them with Solstice Cooperative Consoles, SunSoft ensures that it is easy and compatible.

Finally, SunSoft is working concurrently with systems vendors to ensure that management platforms can supply the framework for distributed system administration and management. The Solstice EM framework is positioned well to support the initiatives in distributed object computing.