Intel® LANDesk® Client Manager OEM Integration Guide

Mini Software Development Kit

Product: Intel® LANDesk® Client Manager 6.3
Documentation Version: Mini.1

Intel Corporation assumes no responsibility for errors or omissions in this document. Nor does Intel make any commitment to update the information contained herein. Implementation and integration information contained in this document may not apply to all versions of LDCM.

Table of Contents

Introduction

About this Mini SDK

The Intel® LANDesk® Client Manager 6.3 Mini Software Development Kit (SDK) provides information to integrate functionality and add value to LANDesk Client Manger that is offered with your OEM product based on Intel desktop boards.

Client Manager is based on the following design objectives:

- Provide robust management capabilities.
- Minimize use of disk space, virtual memory, and CPU.
- Support industry standards.

Important This Mini SDK supports only Windows 32-bit OS types (Windows* 98 Second Edition, Windows ME, Windows NT*, Windows 2000, and Windows XP) and must be used with Client Manager 6.3.

This SDK integration guide has been specifically tailored for use with Intel desktop and Server boards. For support or support contact information, please reference your OEM license agreement and/or OEM Customer Letter included on the OEM LDCM Master CD.

Mini SDK Advantages

Here are a few of the things you can do using the tools and information in this SDK:

- Lower the total cost of computer ownership for your customers.
- Promote and deliver standards via preferred, open desktop management technology.
- Automate custom installations of Client Manager more easily and effectively.
- Enhance the GUI of Client Manager to differentiate your product version. This includes customizing and displaying your own company information.
- Get local or remote access to the alert log file on a client.

What's New for Client Manager 6.3 Mini SDK version

The list below covers the significant updates to this version of the SDK:

- Added information on how to change the package installation key so that a user is notified when a package is installed.
- Added information about PDS2.

Mini SDK Documentation

The topics are divided into these parts (represented as top-level books in the left-pane Table of Contents):

- **Introduction** contains overview and detailed information about system requirements and supported hardware, Client Manager features and upgrades, and SDK installation and directory structures.
- **Client Manager Functional Areas** describes basic architecture, such as discovery, security, CIM and DMI, event management, instrumentation, and power management.
- **Installing Client Manager** describes how to customize and distribute Client Manager installations.
- **Customizing and Integrating with Client Manager** explains how to customize title bars and Help, and how to migrate existing Client Manager customizations. It also discusses integration points (with Intel® LANDesk® Management Suite, other management applications, AMS², SNMP, etc.) and Client Manager registry keys.
- **General Reference** provides details on topics such as the Template Parser Component, the Event Management System, computer health, and discovery technologies.
- **Instrumentation, ASIC, and BIOS Reference** provides details on topics such as support for SM BIOS and SMAL architecture, DMI attributes, ASIC configuration, Heceta Management ASICs, and Alert on LAN*.

Mini SDK Sample Code

The topics for code samples are described below.

Topic	Description
LDCM.MIB	Defines the Client Manager SNMP trap Protocol Data Unit
Creating OEMSTRINGS.INC	File for defining strings that modify Client Manager GUI

Client Manager-related Resources

The following URLs provide information that might be helpful in supporting this SDK:

- http://www.dmtf.org – DMI and SM BIOS specifications
- http://support.intel.com/support/landesk/clientmgr/index.htm – Client Manager support site for software and driver updates

Instrumentation-related Links

Below are links to technical material related to instrumentation:

- Intel Wired for Management:
  http://developer.intel.com/ial/wfm/
- DMI Specification, Conformance Requirements, and other DMI documentation:
  http://www.dmtf.org
- System Management (SM) BIOS Specification
  ftp://download.intel.com/ial/wfm/
- Component Object Model (COM) Overview
  http://www.microsoft.com/com

EULA Requirements

If you distribute Client Manager to end users, whether or not your version is customized from the original Client Manager 6.3, you must also provide an End-User License Agreement (EULA) for each copy you create. You can use the Client Manager EULA as a pattern for your own EULA.

Glossary of Terms

ACPI	Advanced Configuration and Power Management Interface - Specification developed by Intel, Microsoft, and Toshiba for describing and enabling the power management functionality provided by a computer system.
AMS²	Alert Management System 2.
CIM	Common Information Model - Data modeling standard of the DMTF.

CIMOM	Common Information Model Object Manager - Object manager for CIM meta-schema, schema, and instances. Typically, this term is used to specifically describe Microsoft's CIMOM.
CMDMI	Client Manager DMI instrumentation model.
CRM	Configuration Resolution Module - provides configuration data for management ASICs during Client Manager install.
CSS	Cascading Style Sheets - W3C standard for specifying HTML document formatting.
DMI	Desktop Management Interface - Data modeling standard of the DMTF.
DMTF	Desktop Management Task Force - Consortium comprised of companies that are interested in standards pertaining to computer system management.
EMAL	Event Management Abstraction Layer - accessed by Event Consumer Plug-Ins and the Event Management Data Access Component to query and update EMS configuration data
EMS	Event Management System - determines how Event Producers and Event Consumers are created and configured, and how they interact with one another.
EULA	End-User License Agreement.
Health Contributor	An instrumented and state-based DMI indication that helps determine the operational status or “health” of a managed computer as “OK”, “Warning”, or “Critical”.
HTTP	HyperText Transfer Protocol.
I^2C	Inter-Integrated Circuit bus - Multi-master serial bus used as the basis for the Intelligent Platform Management Bus (IPMB).
LDCM	Client Manager.
LDMS	Management Suite.
MMC	Microsoft Management Console.
MND tag	Managed Node Data tag - enables sub-processing of data in the TPC.
NDE	Node Discovery Engine - discovers nodes and node information on a network.
NLFS	Node Log File System - manages local and remote log files in Client Manager.
NPMS	Node Power Management System - manages shutdown and reboot of managed nodes.
PDS	Ping Discovery Service - uses protocols to discover and communicate with remote applications on the network.
PnP	Plug 'n' Play
SDK	Software Developer's Kit - Technical instructions that help third-party software developers incorporate Intel's software into their own product.
SMAL	Systems Management Abstraction Layer - hides ASIC hardware differences from Client Manager instrumentation.
SNMP	Simple Network Management Protocol
SSM	System Space Manager - core module of Client Manager that interacts with System Space Plug-ins (SSPs).

TLP	Tool Launcher Plug-in - launches OEM extension tools from a Client Manager page.
TPC	Template Parser Component - parses HTML display templates in Client Manager.
WbEM	Web-based Enterprise Management - an industry initiative to develop a standardized technology for accessing management information in an enterprise environment.
WfM	Wired for Management - Intel standard for baseline system management functionality.
WoL	Wake on LAN* technology.

Client Manager Requirements

System Requirements

Client Manager supports computers in networked or standalone environments. These are the minimum requirements to use Client Manager:

Administrator computers

- Windows* 2000, Server and Advanced Server, Windows NT* 4.x (Service Pack 6a or later), or Windows XP for the administrator console
- 64 MB of RAM for Windows 2000, 128 MB of RAM for Windows XP

Client computers

- Windows 98 Second Edition, Windows ME, Windows NT* 4.x (Service Pack 6a or later), Windows 2000, Server and Advanced Server, or Windows XP for the client console
- 24 MB of RAM for Windows 98 Second Edition, 32 MB of RAM for Windows ME or Windows NT, 64 MB of RAM for Windows 2000, 128 MB of RAM for Windows XP, Windows 2000 Server and Advanced Server

Administrator and client computers

- Intel® Pentium® microprocessor or higher
- 100 MB of available hard disk space to install
- 40-100 MB of available hard disk space to run (depending on cluster size)
- TCP/IP
- A network adapter or modem connection
- Internet Explorer 5.5 or later
- A monitor resolution of 1024x768, 256 colors or greater is recommended

For network computers, a network card is needed for communication on the network.

Upgrading to Client Manager 6.3

When Client Manager 6.3 is installed on a computer running a previous version of Client Manager, version 6.3 automatically uninstalls the previous version. As with all Client Manager products, you must build and validate the new version 6.3 CD image for all the computers you support.

When you upgrade existing installations of Client Manager, custom files from previous versions are not preserved. You must re-create the alert configuration, general system configuration, and instrumentation files.

Supported Standards and Technologies

Supported Standards

Client Manager 6.3 supports the latest versions of industry standards and technologies, such as WfM, CIM, DMI, HTML and JavaScript*, ACPI, management ASICs, Wake On LAN*, Heceta, ICH, ICH2, and Metolious.

ACPI

Client Manager 6.3 supports ACPI. The administrator console displays the ACPI state of a client, even when the client is "asleep." The 6.3 administrator conforms to the requirements of all ACPI power states, does not interfere with transitions between states, and continues to operate after a state transition. All aspects of Client Manager 6.3 are power management aware; for example, management ASICs and Alert on LAN* do not give alerts when entering or exiting various power states.

Alert on LAN I and II Technology

Client Manager supports the Alert on LAN I and Alert on LAN II technology specifications.

CIM v2.0.

CIM property values may be acquired, modified, and displayed. This functionality depends on the presence of Microsoft's CIMOM on the client.

DMI v2.0s.

DMI attribute values may be acquired, modified, and displayed. This depends on the DMI v2.0s Service Provider and SDK.

DMI Instrumentation for Desktop and Mobile Platforms

Client Manager 6.3 provides HTML display templates that support DMI instrumentation unique to desktop and mobile platforms.

HTML

Client Manager 6.3 supports HTML 4.0 and 3.2, Dynamic HTML, and JavaScript (ECMA script).

Secure File Transfer

The administrator version uses the file transfer functionality of the Win32 Explorer, which depends on the OS security. Using the Win32 Explorer, the administrator version can securely transfer files between an administrator and client, and perform file transfer, rename, and delete on a remote client.

SMBIOS 2.3.1

Client Manager 6.3 supports SMBIOS versions 2.3.1. Client Manager gets nearly half of its management data values from the SMBIOS.

SNMP

Alerting via SNMP traps is supported. Client Manager sends events from DMI, Alert on LAN, and other sources to AMS², which in turn forwards them to SNMP traps. The traps are transmitted to a configured SNMP management application on an administrator computer. Client Manager can also directly forward SNMP traps.

Wake on LAN Support

Power management system "wake up" functionality is supported on clients via Wake on LAN technology.

WfM (Wired for Management) v2.0

Support is provided for certain management functionality and data described by the WfM v2.0 Baseline Specification.

Supported Systems

Client Manager 6.3 has been tested on the following new systems and motheroards:

- S845WD1-E
- SE7500CW2

Client Manager Features

Overall Features

Because Client Manager supports the Win32 operating systems listed in System Requirements, Client Manager can run on those platforms, including servers, without modification. (Client Manager 6.3 is not supported on Windows NT* 4.0 Server.)

A Client Manager 6.3 managed client includes an HTTP server for point-to-point management. To enhance manageability of mobile platforms, the following features are implemented:

- Improved client-to-administrator discovery
- Cross-subnet discovery

- Bandwidth / throughput monitoring
- Alert queuing (store and forward)

Below is a summary of Client Manager features.

Client Manager Feature	Description
CIM and DMI Data Support	• HTML display templates for acquiring and displaying CIM or DMI data
Discovery	• Extended client discovery capabilities through CBA technology.
AMS² Support (AMS² is not included in Client Manager)	• Ability to forward DMI indications as AMS² alerts.
Windows Registry Support	• Windows registry is used instead of .INI files for storing application options and state information.
Communication Support for Remote Management	• Support for RPC via DMI 2.0. • Support for TCP/IP protocol.
Other Features	• Displayable Client Manager text is localized. • Computer Summary Export in CSV and HTML formats, to feed enterprise management applications.
Standards and LANDesk Product Compatibility	• Supports DMI 2.0 Service Provider. • Client Manager can be launched by Intel® LANDesk® Management Suite through a plug-in. • Supports CIM 2.0, WfM 2.0, HTML 4.0 and other industry standards.
Mini SDK Support	• SDK documentation • Broad ASIC support. • OLE functionality. • Customization support for OEM icons, logo display, links, etc.

Product Features

Inventory Features

The administrator and client versions of Client Manager can view hardware and software inventory data on a client.

- Inventory information provided by previous versions of Client Manager is maintained and supported.
- The administrator and client versions and client instrumentation support dynamic/swappable devices.
- Battery charge and wear information is provided. ACPI supports this feature. The Smart Battery specification provides wear by counting charge cycles or by comparing the LastFullChargeCapacity to the DesignCapacity control method.
- Bus card information is provided (bus type, number of bus, detailed information for the computer bus).
- Memory array information is provided (memory type, size, bank location, number of socket).
- Node up-time information is provided (as reported by the OS).

- System contact information is supported.
- System slot information (slot type, width, current usage, description, category, virtual slot) is provided.
- Video BIOS date, version, and model number are provided.

Alerting Features

The alerting functionality is used to notify a user of significant conditions on a client that may require attention. Alerting provides value "out of the box," with little user configuration needed.

Alert on LAN* and DMI indications are supported as alert sources. Alert handling and alerting are supported via message boxes. Alerts are logged on the node where they originated.

SNMP Traps

The traps are transmitted to a configured SNMP management application on an administrator. Alerting via SNMP traps is supported. Client Manager sends events from DMI, Alert on LAN, and other sources to AMS², which in turn forwards them to SNMP traps. Client Manager can also directly forward SNMP traps.

Queuing

Alerts that are generated while a client is disconnected from the network are delivered (to a configured managing node) when the client is reconnected. Alert handlers are pre-configured as much as possible.

Local or Remote Alert Configuration

Alerting on the client is configurable from the Client Manager 6.3 administrator console.

Power - Sleep State Features

The basic power/sleep-state features of Client Manager 6.3 are described below.

- The administrator or client console can modify the power/sleep state of a client.
- Soft shutdown is supported; the OS is allowed to exit gracefully.
- System reboot is supported. A soft shutdown is performed, and then the client reboots.
- "Wake up" is supported. Wake-on-LAN*-enabled clients can be awakened from a sleep state.

Design Features

Console Design

Administrator Console

The Client Manager 6.3 administrator console provides management access to Client Manager 6.0 to 6.3 clients. The administrator console supports management of remote nodes via dial-up networking, such as RAS.

The administrator console consists of HTML pages processed by the Template Parser Component (a CGI script) on the administrator's computer. These HTML pages contain discovery information and can perform a wide range of functionality.

The URL for a computer's Select Computer page is:

http://<computername>:<port>/index.tpc?_admin=1

The Template Parser Component:

1. Communicates with the NDEDAC (Node Discovery Engine Data Access Component) to get discovery information.
2. Fills in the admin HTML templates.
3. Serves these to the administrator.

The discovery HTML page enables the administrator to provide parameters for discovery, such as including certain subnets. These settings are saved and populated within the discovery HTML page the next time the administrator selects it.

Client Console

The Client Manager 6.3 client console provides local management access to a client. It supports viewing management data via customizable HTML display templates. It also provides for simpler migration and integration of user data.

About the Client Manager HTTP Server

The Client Manager HTTP server gives a local or remote web browser primary access to management data and functionality on a client, via the industry standard HTTP protocol. The Client Manager HTTP server is

installed on both the administrator and client computers.

Console Toolbar

The toolbar within the Client Manager console provides easy access to the following features described below.

Icon	Description
	Discover Computers
	Add Computer(s) to List
	Refresh the Computer List
	Remove Computer(s) from List
	Run Inventory Reports
	Export Inventory Information
	Reboot Computer
	Shut Down Computer
	Wake Up Computer
	Configure Alert on LAN* Options
	Perform Alert on LAN Actions
	Download Packages
	Install Packages
	Launch Windows Explorer, Use OS File Transfer
	Show All Computers
	Show Available Computers
	Show Only Unhealthy Computers

Web Features

Client Manager 6.3 provides enablement for browser-based management. Clients and administrators can use a Web browser to access management information via HTTP.

- Each client is equipped with an HTTP server that supports direct point-to-point management access from a Web browser.
- The HTTP server on each client permits access by authenticated and authorized users only.

- Baseline Web content standards are supported. The baseline Web content on a client makes use of browser-independent standards-based conventions such as HTML 3.2, CSS level 1 (CSS1), and JavaScript* 1.2.
- The baseline Web content provided on a client is validated against Internet Explorer 4.x and 5.x.
- Some baseline Web content is provided in the client software stack.
- Hardware/software system summary information is provided.
- A snapshot of node health information is provided.
- Access to the alert log on the client is provided.

About Template Parser Components

The Template Parser Component (TPC) is at the core of all Client Manager management activities. It can parse a Display Template containing combinations of XML- and HTML-encoded entities. These entities refer to Data Access Components (DACs), which are dynamically linked modules typically used for obtaining management data.

The TPC is a CGI executable that gets and sets data with one or more management data sources. These sources may include DMI, CIM, SNMP, or any management data source that has a corresponding Data Access Component (DAC).

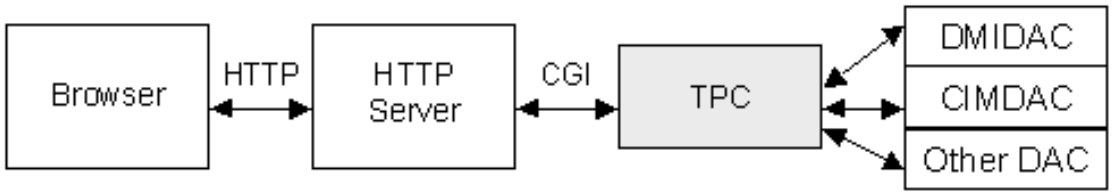


Figure - General Client Manager data flow

The TPC follows this general procedure:

1. Accepts a Display Template as input.
2. Ignores HTML directives.
3. Processes the XML-encoded entities.
4. Passes each entity to the DAC specified within the entity.
5. Replaces each entity with the output from the DAC that processes it.

What remains is an HTML page that is returned to the HTTP Server and browser.

Discovery

Discovery Features

Central to good computer management are the concepts of managed node and managing node. In the case of Client Manager, the administrator is the managing node that performs discovery operations. Discovery allows the administrator node to create a list of known computers that it can then manage.

Discovery is used by administrators and clients to locate and connect with one another for management operations. Discovery features include:

- Extended client discovery capabilities through PDS (Ping Discovery Service) and PDS2 technology.
- Client-to-administrator communication. When a managed node comes online (is booted or connects to a network), it validates the presence of an administrator and requests (initiates) management activities and services from the administrator.
- Enhanced discovery across sub-nets. A managed node can discover an administrator on a different sub-net.
- Multiple administrators. A managed node can validate and be managed by as many as five administrators that reside on the same or different sub-nets as the administrator.
- Minimal configuration needed. A managed node requires no (or minimal) manual configuration to validate one or more remote administrators.
- Constraint by health. Discovery of managed nodes can be constrained to nodes based on health characteristics.

- Constraint by sub-net. Managed node discovery may be constrained to nodes on particular sub-nets.

Discovery Diagram

Below is a diagram of the overall discovery process in Client Manager. PDS is included for backwards compatibility.

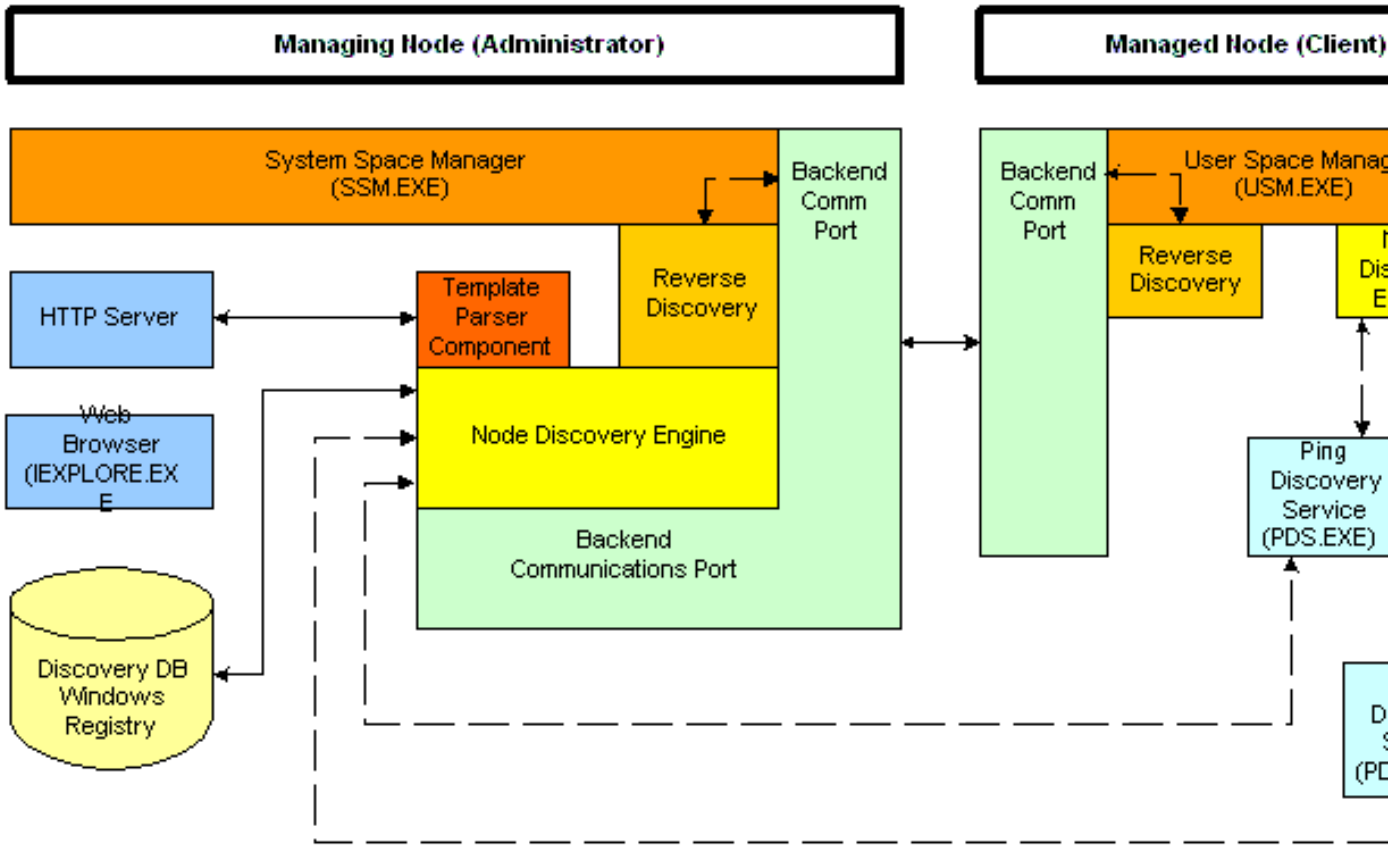


Figure: Client Manager discovery process

CIM

About CIMOM

Important: CIM Object Manager (CIMOM) is not installed with Client Manager; you must install or include it yourself. Client Manager CIM components can be installed only if CIMOM is already present on the computer. If you install CIMOM after Client Manager, you must re-install Client Manager for it to use CIMOM and CIM components.

CIMOM is the Microsoft implementation of the CIM specification, which facilitates gathering and manipulating information about system resources that are represented as managed objects. Instead of organizing data into tables and rows as in the relational database model, CIMOM organizes data into classes and retrieves them as instances.

CIMOM cannot access management information directly. Instead, it relies on providers (Win32 provider, NT Event Log provider, SNMP provider, the Registry provider, etc.) that gather information for management applications. Providers, whose equivalent in the DMI world is instrumentation, do the following:

- Supply CIMOM with data from managed objects
- Handle requests on behalf of management applications
- Generate notifications of events.

The CIMOM returns managed objects when requested. These objects have properties that are accessed to

inform the management application about the client. CIMOM is similar to the DMI Service Provider.

WbEM and CIM

Web-based Enterprise Management (WbEM) is an industry initiative to develop a standardized technology for accessing management information in an enterprise environment. WbEM is also an information manager designed to implement the Common Information Model (CIM), a schema of managed objects developed by the Desktop Management Task Force (DMTF).

WbEM includes a central component called the CIM Object Manager (CIMOM), which helps gather and manipulate information about system resources represented as managed objects.

Important: Client Manager supports CIMOM but does not install it. To access CIM data via Client Manager, you must install CIMOM before installing Client Manager.

CIM MI Diagram

Below is a diagram of the CIM Management Interface in Client Manager.

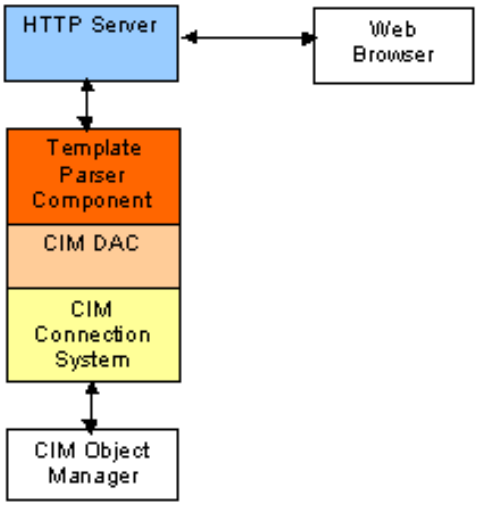


Figure: CIM Management Interface in Client Manager

DMI

DMI Elements

About DMI

The DMI specification defines standards to remotely manage hardware and software on a platform. DMI provides information about a platform that can help in analysis and repair situations.

The DMI software stack is divided into three main layers:

- **Management Application (MA)**A program that initiates management requests. The MA uses the DMI SP to access data from Component Instrumentation.
- **DMI Service Provider (DMI SP)**An application that runs on the target system and provides services between the MA and Component Instrumentation. The DMI SP arbitrates access to Component Instrumentation and manages the Management Information Format (MIF) database.
- **Component Instrumentation (CI)**Works directly with the hardware, software, and OS to make a platform manageable to DMI.

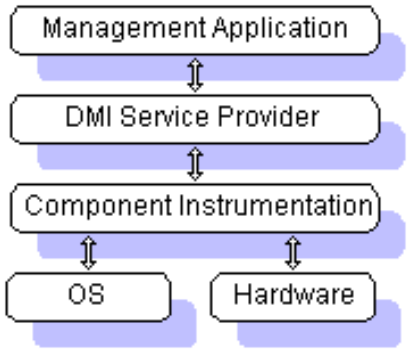


Figure: The DMI software stack

For more information on DMI terminology and technology, see the Desktop Management Task Force (DMTF) Web site at http://www.dmtf.org.

DMI Indication Support

DMI indications are processed and used by Client Manager consoles and agents.

- All DMI 2.0 indications generated by Client Manager instrumentation are mapped to AMS² alerts (if AMS² and WUSER agents are installed).
- All DMI 2.0 indications generated by Client Manager instrumentation are mapped to SNMP traps (if SNMP is installed).
- DMI indications can be forwarded to pagers, beepers, SNMP, etc. via AMS² (this requires Intel® LANDesk® Management Suite).
- Local agents constantly monitor key elements to produce DMI indications if problem areas exceed pre-defined, configurable thresholds.
- Indications are published externally (if connected) and stored in a local log file.

DMI Service Provider

The DMI Service Provider (SP) used by Client Manager coordinates communication between component instrumentation and management applications. It also manages information in the MIF database and passes information to the Management Application.

All functions that manage the MI and CI interfaces are coordinated with the SP. These functions consist of MIF-enabled component installation, registration of the CI and the management app, request serialization and synchronization, and general flow control.

Wherever applicable, Client Manager modules that call the DMI 2.0 MI use class strings (not hard-coded IDs) as the index to MIF attribute groups. This allows instrumentation that is registered with the SP to work transparently with the GUI.

The SP invokes instrumentation to interact with manageable components and attributes on a computer. To query or set the value of an instrumented attribute, the designated instrumentation is invoked to call the operating system or APIs to perform the task.

MIF Files

Getting Management Information

The Client Manager GUI displays management information about local and remote computers. This management information:

- Comes from the MIF database on each computer that has Client Manager loaded.
- Is static or dynamic.
- Is linked into the user interface via Component IDs, Group IDs, and Attribute IDs.

Because DMI standardizes at the group level rather than the component level, a product can include standard groups of attributes as well as proprietary ones. MIF files are usually provided by hardware and software vendors who support DMI manageability.

Static Information

Static information can be added to the DMI MIF database. For example, static data produced by Client

Manager is added to SOFTWARE.MIF, which is loaded when the Service Provider starts.

Dynamic Information

Dynamic DMI data comes from DMI instrumentation, or programs written by the component manufacturer. Dynamic DMI data provides real-time attribute values to the Service Provider as they are requested.

MIF Files

The MIF language describes a component's manageable attributes. It consists of specific grammar and syntax defined by the DMTF. A MIF file is a simple ASCII or Unicode file describing a product's manageable attributes. Each DMI component must provide its own MIF file.

All MIFs that ship with Client Manager are compliant with applicable DMTF definitions for standard groups. The COMPCHK2 compliance-checking utility is used to validate each Client Manager product release.

Where necessary, the Client Manager console has been modified to support accessing attributes within standard groups. For backward compatibility, a Client Manager 6.3 console can fetch information from Client Manager 6.x (and later) components.

MIF Data Formats

The MIF data model is organized into *components*, *groups*, and *attributes*. Components can have *attributes* that provide information for management applications. The attributes are collected into *groups*. For information on how the DMI data format is implemented, see the latest DMI Specification at http://www.dmtf.org.

MIF Files Provided with Client Manager

The MIF files shipped with Client Manager describe the interface between the instrumentation and the Client Manager GUI. However, not all attributes in the MIF files are used by Client Manager. To learn more about MIF files, see the DMI 2.0 Specification.

Client Manager includes both DMI-specific and Client Manager-specific MIF files. The following describes the MIF files included with Client Manager.

- **IOAOL.MIF**Alert on LAN* information, such as cover or LAN leash tampering, temperatures, voltages, and processor missing.
- **IOAPPS.MIF**Inventory information, such as product name, version, filename and file size, path, and installation date and time.
- **IOCPU.MIF**CPU information, such as processor type, system cache, and Slot 2.
- **IODISKS.MIF**Disk information, such as physical attributes, partitions, logical drives, disk space, and disk failure prediction.
- **IOHID.MIF**Human input device information, such as pointing devices and keyboards.
- **IOHWRES.MIF**Information for hardware system resources, such as extensions, DMA, IRQ, I/O, and memory.
- **IOMEDIA.MIF**Multimedia information, for devices and drivers.
- **IOMEMORY.MIF**Application information, such as physical memory array, memory-mapped addresses, memory devices, system memory settings, virtual memory, and parity error detection.
- **IOMOUSE.MIF**Sample MIF information (mouse position).
- **IONET.MIF**Network information, such as network adapter, network adapter driver, and TCP/IP.
- **IOOS.MIF**Operating system information, such as general information, environment variables, Win32 tasks, and Windows NT* drivers.
- **IOPORTS.MIF**Information for serial, parallel, and infrared ports.
- **IOPOWER.MIF**Application information, such as battery, power management, and dynamic states.
- **IOSMAL.MIF**Information for fans, intrusion, temperatures, and voltages.
- **IOSYSTEM.MIF**System information, such as system BIOS, system management BIOS, motherboard, system slot, and POST error detection.
- **IOVIDEO.MIF**Video information, such as video BIOS, video and monitor resolutions, and video drivers.

MIF Database

The MIF files are used to seed a database maintained exclusively by the Service Provider. The MIF database describes (and optionally stores) information about a computer's hardware and software via

manageable component attributes. The database format is determined by a MIF definition file that you can create with a text editor. When a MIF file is initially installed, the information in the MIF file is added to the MIF database and made available to the Service Provider and thus to management applications.

The values associated with a MIF attribute are found in the MIF database or provided by the instrumentation that runs when the value is requested. An attribute can be accessed individually, or as an entry in a table. Values can be designated as read-only, read-write, or write-only. As components are removed from the system, the Service Provider handles the removal of the information from the MIF database.

Hardware and software developers are encouraged to supply MIF files for components so customers do not have to manually create a MIF file to represent the manageable component being added to the system.

MIF database processing consists of installing and uninstalling component schema and querying and setting MIF information.

CI and Instrumentation

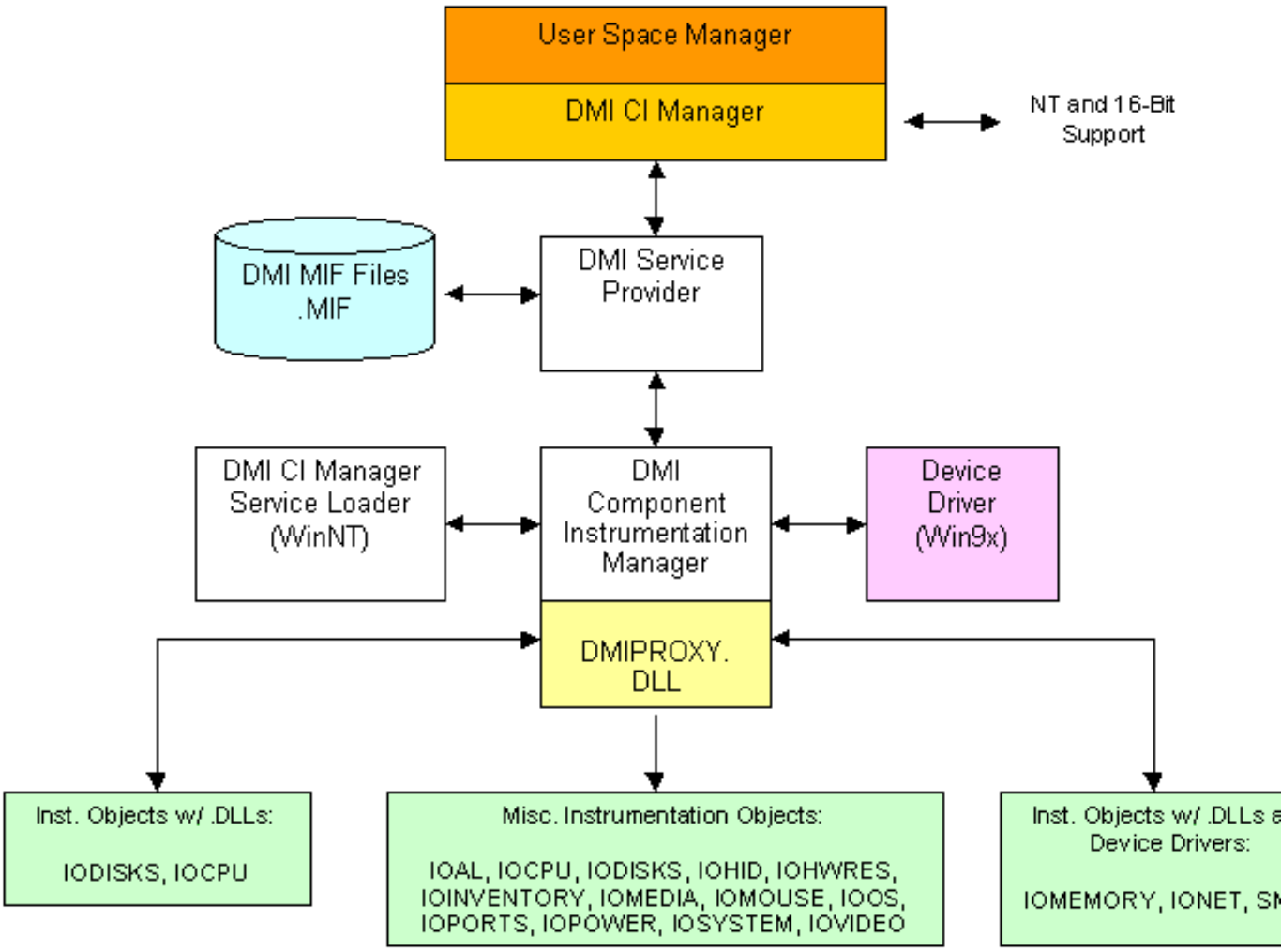
Component Interface (CI)

The Component Interface (CI) relays information about computer system components back to the Service Provider. This information is then stored in a MIF database for use by management applications. The Component Interface controls communication between the Service Provider and manageable products.

Component instrumentation supports interaction with a local DMI 2.0 Service Provider through the Component Interface. The Component Interface (CI) layer collaborates with the SP, usually through an additional layer called the CI Manager, to relay information back to the management application. The CI Manager usually invokes instrumentation to satisfy the requests from the SP. The DMI Service Provider supports direct interface programs; they are always running and registered with the Service Provider to get and set requested attribute values and persistent data.

CI Diagram

The diagram below shows the overall structure of component instrumentation in Client Manager.



About Instrumentation

Instrumentation is the code that provides manageability information. Consider instrumentation in a car: it gathers data on the vehicle's traveling speed, the engine's temperature, and the amount of fuel in the tank. This information can head off problems (by letting you know when it's time to buy gas, for example), or minimize the impact of problems that occur.

Computer instrumentation in Intel processors, motherboards, software, and system components provides data about different aspects of a computer system. For example, a fan instrumentation object instruments data on all fans present on the motherboard.

The information that comes from instrumentation code:

- Is either static (such as system configuration) or dynamic (captured from system health monitoring). Information ranges from mechanical features (voltage and temperature) to high-level items (attempted break-ins and performance bottlenecks).
- Is reported in widely available, standard formats. The operating system or management applications can send it to the user, alert a system administrator, or use it to take preventive action.

Instrumentation makes systems more manageable in these ways:

- Users and IT support staff can predict problems, prevent them, or recover from them quickly.
- IT support staff can reduce the cost and complexity of deploying and managing computers and servers. They can perform many more tasks remotely, from their own desktops.
- IT support staff gains ready access to information about high-level management tasks such as asset and performance management.

Computer Health: Event handling in Client Manager

Health Features

Below are some of the basic computer health features in Client Manager:

- The administrator and client consoles and supporting agents can configure health-contributing attributes and monitor overall health of a client.
- Health monitoring provided in Client Manager 6.x is maintained and supported.
- The administrator and client consoles and instrumentation support the configuration of independent threshold settings for each disk drive.
- Inventory items can be customized or added as health contributors.

Power Management

About NPMS

The Node Power Management System (NPMS) handles all node power management functionality for Client Manager, such as node wake-up, node power-down, and node reboot.

The NPMS consists of these components:

- The Node Power Management System Data Access Component (NPMSDAC). The NPMSDAC exports all NMPS functionality through the DAC interface. Doing so allows web access to the NPMS.
- The Node Power Manager (NPM). The NPM enables node power-down and node rebooting.

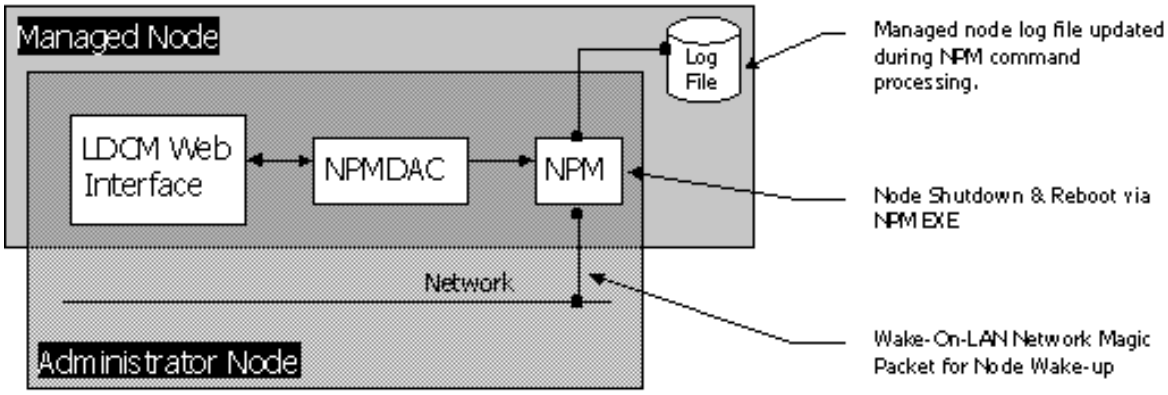


Figure - General NPMS use model

The NPMS system uses the NPMSDAC, the NPM, and a Web-based interface to perform these primary functions:

1. Shut down the local node.
2. Reboot the local node.
3. Wake up a remote node.

Shutdown-Reboot Implementation

Shutdown and reboot are handled differently from wake-up. The reasons for this are:

- A node cannot be remotely rebooted via an OS call on all operating systems
- Wake-up commands cannot be processed in software on a powered-down system.

The NPMS notifies the end-user that the node is about to be shut down or rebooted. The client user can then be given some time to cancel the shutdown/reboot command. Shutdown and reboot functionality are implemented only on the client, to avoid causing the administrator Web interface to time-out while the NPMS system waits for the command to complete.

Remote Shutdown and Reboot

The NPMDAC and NPM are installed on the client to facilitate shutdown and reboot capabilities. The local

node can be shut down or rebooted remotely via the Client Manager Web interface. To shut down or reboot the local node via the Web interface:

1. The Web interface issues a shutdown/reboot command to the NPMDAC on the client.
2. The NPMDAC starts the NPM process, sending the shutdown/reboot information to the NPM.
3. The NPM completes the shutdown/reboot command process.

Remote Wake-Up

Client Manager supports remote power control (Remote WakeUp, Wake On LAN*) for remote power-on and power-off. With the emergence of computer power management technologies, it's important to be able to manage and communicate with a device in a power-down mode. Client Manager also supports AMD's Magic Packet* technology.

The Select Computer page accommodates power management by:

- Displaying computers previously identified as power-manageable.
- Powering them up or down.
- Listing computers after they have been rebooted.

The NPMDAC and NPM are installed on the administrator node to facilitate remote node wake-up.

Remote Wake-up via Client Manager's Web Interface

The remote node can be awakened remotely via the Client Manager Web interface:

1. The Client Manager Web interface issues a wake-up command to the administrator node NPMDAC.
2. The NPMDAC starts the NPM process with the wake-up information placed onto its command line. If there are multiple nodes to wake up, the NPMDAC spawns one NPM process at a time until all nodes wake-up attempts are complete. This one-at-a-time approach is used because the first NPM will incur the image disk-to-memory time hit, while subsequent NPM instances load instantly due to cached image data.
3. The remote node MAC address and network broadcast addresses are processed from the command line. The NPM does not allow multiple simultaneous wake-up attempts to occur, due to command line buffer limitations. Incomplete command line data is not processed.
4. The NPM binds to a Winsock 1.1 socket and sends the Magic Packet to the broadcast network address specified. This completes the remote wake-up command process.

NPMS and Wake on LAN

The NPMS system can wake up a powered-down client only via Wake on LAN technology. Wake on LAN technology enables a networked node to be powered up remotely when it receives a Magic Packet. The Magic Packet is a UDP packet sent via network broadcast to the client's network. The client network card powers up the node when it receives the Magic Packet. The Magic Packet is sent directly from the NPM onto the wire.

Some of the failure points in Wake on LAN technology are:

- A node may not power up if the NIC receiving the Magic Packet does not support Wake on LAN.
- The Magic Packet must be sent via UDP protocol, which is susceptible to packet loss on the network.
- The client receiving the Magic Packet may have Wake-On-LAN technology disabled in the system BIOS.
- The client may not receive the Magic Packet via network broadcast, because the network router is configured so as not to forward broadcast packets.
- The client may actually be unplugged.

Security Access

About Security (HTTP Server Access)

Client Manager can be accessed via the Web using Internet Explorer. This is made possible by the HTTP server, which implements a subset of the HTTP 1.1 protocol. Because important node information can be accessed and transmitted across the Internet or intranets, adequate security is needed. This includes authenticating the client making the request as well as securing the object content being transmitted. Client Manager handles access authentication but does not handle encryption of the content matter.

The basic Client Manager security model is shown below.

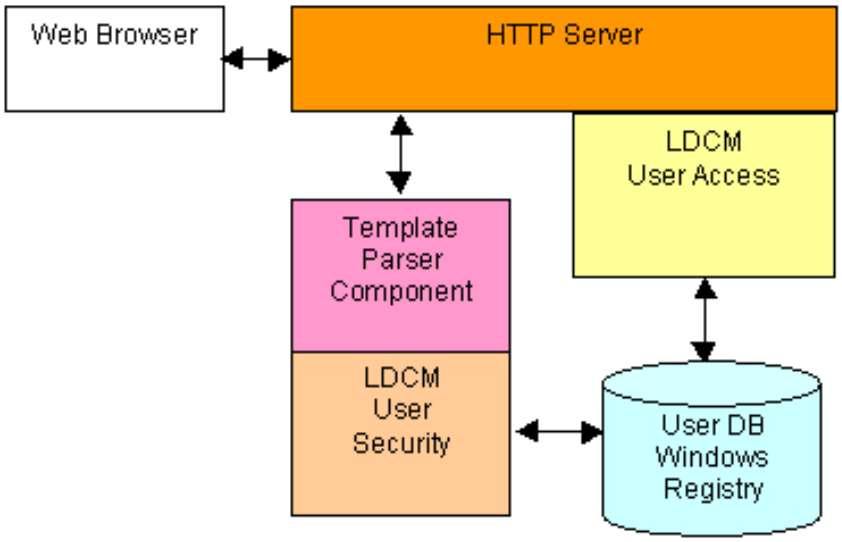


Figure: Basic Client Manager security model

Access Authentication

The HTTP protocol provides a framework for a challenge-response authentication mechanism. This mechanism can be used by a server to challenge a client request and by a client to provide authenticating information. HTTP 1.1 includes the Basic Access Authentication specification. This is based on a password model, where the client must authenticate itself with a user ID and password (no encryption, but it is encoded).

For details on user-level access, see !ACCESS.INI Settings and its related topics.

!ACCESS.INI Settings

You can control user access to Client Manager files and directories by using !ACCESS.INI files. !ACCESS.INI provides the HTTP server with access information about the files available to the server.

Each time a file is requested from the HTTP server, the security system checks the user's access rights against the rights needed to access the requested file. If the user access rights are sufficient to access the file, the server allows access; if not, the server returns an Unauthorized error. This causes the browser to display the username/password dialog.

An !ACCESS.INI file is placed in ...\LDCM\wwwroot and can be in any directory there or below. The entire Client Manager Help system has its own !ACCESS.INI file that grants help file access to anyone without the need for authentication. By default, !ACCESS.INI is installed in the directories mentioned in !ACCESS.INI Installed Directories.

GET and POST Access Levels

The access levels for GET (read) and POST (write) in the !ACCESS file are defined below. By changing these values in an !ACCESS.INI files, you change the access rights for the matching file.

- 0 = authorization not needed; everyone can do the operation
- 1 = browse user
- 2 = user (read plus limited write)
- 3 = power user (read plus more write)
- 4 = admin (power user plus add/remove users)
- 15 = no one can do the operation

!ACCESS.INI allows separate settings for an HTTP GET request and an HTTP POST request. The access rights needed to perform either operation can be set to different values. This enables anyone to GET the requested file and only an ADMIN to perform a POST on the requested file.

Sample !ACCESS.INI File

A sample !ACCESS.INI file is shown below.

```
[ACCESS]
 :default=GET:1|POST:2
 index.htm=GET:0|POST:2
```

```
index.tpc=GET:0|POST:3
```

Default Statement

In the example above, the :default statement specifies browse-user rights for GETs and user rights for POSTs. The defaults apply to all files in the directory except those listed individually. !ACCESS.INI can have one or zero default statements.

Individual Files

In the example above, index.htm has a GET of 0 (everyone can GET) and a POST of 2 (User, Power User, or Admin access). index.tpc has a GET of 0 and a POST of 3. (index.htm and index.tpc are the startup screen files for Client Manager.) Wildcards in individual filenames are not permitted.

Scope of !ACCESS.INI

Below are the rules for how settings in !ACCESS.INI affect files. !ACCESS.INI exists in directories starting at ...\LDCM\wwwRoot and below.

1. There is an !ACCESS.INI file in the directory.
    a. **Individual settings**If there are individual settings in !ACCESS.INI, they apply only to matching files in that directory, not to any files above or below that directory.
    b. **Default settings in the directory**If there is a default setting in !ACCESS.INI, it applies to all files in the directory that don't have individual settings.
    c. **Default settings below the directory**If any directory below this directory has no !ACCESS.INI file, the default setting applies to all files in those directories.
2. There is no !ACCESS.INI file in the directory:
    a. **Inheriting from default settings above**Files in the directory inherit rights from the default statement of the !ACCESS.INI in the closest parent directory above. This is the same kind of principle as 1c.
    b. **Inheriting system defaults**If there is no default statement or !ACCESS file in the directory and none exist above it, the system defaults (GET = 1 and POST = 15) apply to all files in and above that directory.

!ACCESS.INI Examples

Example 1

```
[ACCESS]
 :default=GET:1|POST:15
```

The example above allows any Browse User, User, Power User or Admin to GET all files in the directory. The POST:15 prohibits everyone from POSTing to the file.

Example 2

```
[ACCESS]
 :default=GET:1|POST:15
 postme.tpc=GET:3|POST:3
```

The last line in the example above allows a Power User or Admin to both GET and POST the file postme.tpc. All other files in the directory are subject to the :`default` statement.

Example 3

```
[ACCESS]
 :default=GET:1|POST:15
 test1.tpc=POST:4
```

The above example allows an ADMIN to POST to the file test1.tpc. Because there is no GET setting for test1.tpc, the GET is controlled by the :default settings (GET: 1).

!ACCESS.INI Installed Directories

!ACCESS.INI security files are installed in the following directories. You can modify these files, add them to other directories, or delete them as necessary so your users have appropriate rights to files in any given directory.

- ...\LDCM\wwwroot\
- ...\LDCM\wwwroot\cgi-bin
- ...\LDCM\wwwroot\Common
- ...\LDCM\wwwroot\Common\Media

- ...\LDCM\wwwroot\Common\Media\backgrounds
- ...\LDCM\wwwroot\Common\Media\Topbar
- ...\LDCM\wwwroot\Content
- ...\LDCM\wwwroot\Help
- ...\LDCM SDK\Instrumentation\IOMouse\wwwroot\samples

About Store and Forward

Store and forward enables a disconnected client, such as a mobile computer, to save and send the notifications (local events) that occurred during the disconnection. When the client reconnects to the network and re-establishes communication with the administrator, the notifications are passed on to the administrator.

Operations

Below are the steps that happen when a notification is generated on a disconnected client, and the client tries to notify an administrator.

1. A notification is generated on a client.
2. The client somehow becomes disconnected from the network or can't communicate with an administrator.
3. If the notification came from a local event, the notification is written to a Store and Forward file and to the registry.
4. The client tries to connect to administrators that it knows about.
5. Attempts to forward notifications are made on startup, or when an event occurs, or when an administrator initiates contact with the client.
6. If the client already has a Store and Forward file for that administrator, the newly received client notifications are appended to the existing Store and Forward file. Otherwise, a new file is created for that administrator.

   The order in which a client creates Store and Forward files might not be the order in which they are stored on administrators.

Forwarding to Multiple Administrators

Store and Forward files are sent on a threaded basis, one at a time, to each assigned administrator. Files are sent through the client's Back-end Communications Port (BCP)

For example, a mobile client that is managed by two administrators might have some notifications while disconnected from the network. When the client is reconnected to the network, it will send all unsent events to both administrators.

Installing Client Manager

About Client Manager Installation

Installation Features

The Client Manager 6.3 installation utilities enable a user to install the maximum amount of applicable management functionality on a node with the least amount of user intervention. The installation process can automatically obtain information such as the target platform (server, desktop, workstation, or mobile system), the target platform OS, available manageable devices, and supported management functions such as Wake On LAN*.

The installation utilities include default Client Manager installation (SETUP.EXE), to install versions for administrator, client, or both. Default configurations are provided for applicable management features.

The client installation:

- Provides default health configurations for health contributors and thresholds, on mobile, desktop, and server platforms.
- Enables third-party configuration of installation defaults.
- Supports agents for Intel® LANDesk® Management Suite 6.x (through AMS² included in it), Desktop Manager, and CBA (Common Base Agent).

Adding Custom Files to Client Manager Installations

When you add your custom files, you must put them in the directories where similar Client Manager files exist, or in OEM subdirectories.

File Type	Client Manager Directory to Receive File
General Help files, graphics for Client	...\Client or Admin\wwwRoot\help
Help files for Inventory	...\Client\Html\wwwRoot\help
Help for Admin	...\Administrator\Html\wwwRoot\help

Hardware Support Features

Client Manager uses the Windows Registry to store information about how the hardware is configured for ASIC support. ASIC support refers to the underlying hardware monitoring devices for managing fans, temperatures, voltages and chassis intrusion. Client Manager generically supports a wide variety of ASICs that follow industry-accepted parameters. The Hardware Setup utility enables OEMs to define the setup for a particular type of system and then distribute that same setup data via the Client Manager install. The data is preserved in registry-exported format so it can be easily and quickly imported or installed on the target computer.

The SMBus Implementers' Forum page discusses efforts to develop an ACPI SMBus device driver and access control methods. For more information, see http://www.smbus.org/drivers/index.html

Pre-defined .REG Files for Installation

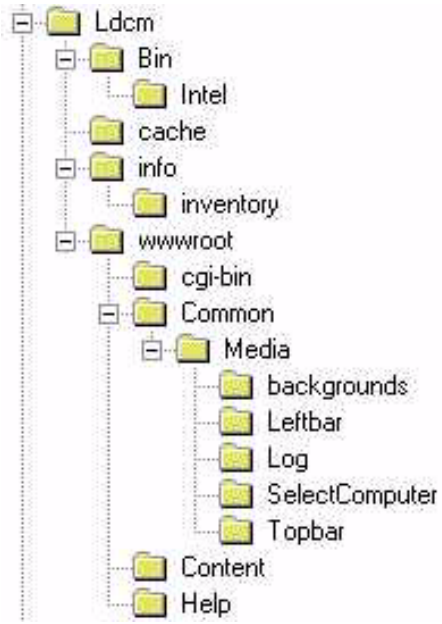
The Asic Configuration registry files are included in Client or Full\AsicCfg directories. They are pre-configured setup data files.

If more than one file is included, Client Manager will display a selection list at install time; if you include only one file, Client Manager will use that file without displaying a list.

Installed Directories and Files

Installed Directories - Admin and Full

The directories installed in the Administrator version of Client Manager are shown below. See also Installed Files.

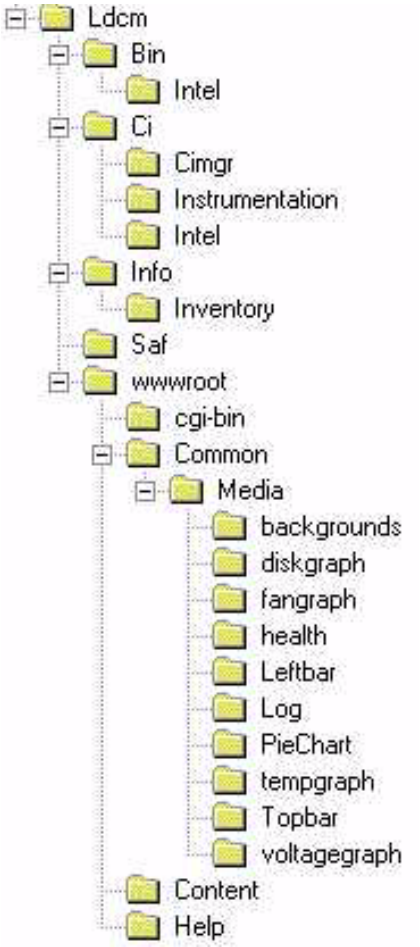


Full Installation (Admin + Client)

The Full version of Client Manager contains all the directories of the Admin and the Client version, plus one addition: in the Full version, the SAF (Store and Forward) directory appears directly above the wwwroot directory.

Installed Directories - Client

The directories installed in the Client version of Client Manager are shown below. See also Installed Files.



Installed Files

The files installed into the Client Manager directory are shown below, with install type (a=administrator, c=client) and description. The DMI directory (client only) contains \DMI\BIN\win32sl.exe (the Win32 Service Provider) and \DMI\BIN\win32.sl.mif (the Service Provider MIF).

Dir/Filename	Installed in	Description
...\LDCM	a/c	
ams2.cfg, ams2inst.dll	a/c	AMS² configuration
aol*.*	a/c	Uninstall files for Alert on LAN*
ASIC.crm, CRMM.dll	a/c	ASIC Configuration Resolution Module files
enudmist.dll	a/c	DMI Start (English)
enuldcminst.lrc	a/c	Language resource files for install
readme.txt	a/c	Readme for Client Manager
regsvr32.exe	a/c	Register OCX
uninst.isu	a/c	Uninstall file
... \LDCM\BIN	a/c	
AAM.dll	a/c	

bcp.dll	a/c	Backend Communciations Port
bootstrp.dll	a/c	Bootstrap/startup file
HTTP.dll	a/c	HTTP Server
IIDS.exe	a/c	Intel Internet Data Server
NML.exe	a/c	Node Manager Launcher
scriptmap.ini		CGI script mapping tool
ssm.exe	a/c	System Space Manager
usm.exe	a/c	User Space Manager
...\LDCM\BIN\INTEL	a/c	
*.dac	a	Event Management (EM), General Operations (GO), Inventory (Inv), NDE, NLF, NPM, User, Version
*.dac	c	CIM, DMI, DMI Browser
*.dll	a	CManager, cmsnap, enuldcm (Backward Compatibility),
*.dll	a/c	BCPSND (Backward Compatibility), Bootstrp, DCSC / DCSE / DCSS (DMI Connection System), DMIEPS (DMI Event Packaging System), EMAL, enudmist (DMI Start), IAMCD / IAMCH (Notification Monitor), ILocale, IXMLE, IXML, ldcmpds, NDE, NIRL, NLF, NLFE, UsrDir
*.dll	c	Elogapi, enuSCN32, LDCompress, NLFNTM, RDDE, Ygrep32
*.htx	a	HTML scripts for reboot, shutdown, etc.
*.ssp (plug-ins)	a	AMS, BCMsg (Backward Compatibility), DMIREPS, IAMC, NDE, NLF, SNMP, sounds
*.ssp (plug-ins)	c	DMIUCIM, Notification Monitor, Node Health System, Node Log File, Store and Forward, SNMP
(Lang. resources)	a	IAMC, NLFE, NM, NPM
enuscn32.mrl	c	
export.exe		Export Summary
ldappl.ini	c	Inventory INI file
ldcm*.*, ldcontrols.ocx		Backwards compatability
ldiscan.exe	c	Inventory scanner
npm.exe	a/c	Node Power Management
...\LDCM\CACHE		misc.
.. \LDCM\CI\CMGR	c	
cimgr*.exe	c	CI Manager and support applications.
... \LDCM\C\INSTRUMENTATION	c	
(*.exe)	c	Instrumentation objects, IOAPPs to IOVIDEO

(*.mif)	c	MIFs for above
(*.reg)	c	Registry files for above
readme.txt, *.vxd		Files for installing Windows* 9x drivers
... \LDCM\CI\INTEL	c	
16bit*.*	c	16-bit OS instrumentation files
NTUSER*.*	c	NT instrumentation files
...\LDCM\INFO	a/c	
*.toc	a/c	HTML Help table of contents files
leftbar.ndx		Index file to customize left bar in Client Manager
...\LDCM\INFO\INVENTORY		
*.ndx		Index files for applications through Video
*.toc	a/c	Table of contents Help files for applications through Video
...\LDCM\WWWROOT	a/c	
!access.ini	a/c	HTTP security settings for index files
(*.htm)	a/c	HTTP errors
(*.tpc)	a/c	Template Parser index files
...\LDCM\WWWROOT\CGI-BIN	a/c	
!access.ini	a/c	HTTP security settings for TPC
tpc.exe	a/c	Template Parser Component
... \LDCM\WWWROOT\COMMON	a/c	
(*.js)	a/c	JavaScript* files, HTML content
(*.inc)	a/c	Help file string tables
!access.ini	a/c	HTTP security settings for .CSS
defaultcss.*	a/c	Default files for .CSS
... \LDCM\WWWROOT\COMMON\MEDIA -- and subdirectories	a/c	
(*.gif)	a/c	GIF files
... \LDCM\WWWROOT\CONTENT	a/c	
!access.ini	a/c	HTTP security settings for TPC files
(*.TPC)	a/c	TPC files for Alert on LAN through Video
... \LDCM\WWWROOT\HELP	a/c	
(*.gif)	a/c	GIF files for Help
(*.htm)	a/c	HTML Help topics
(*.inc)	a/c	Help file string tables

(*.tpc)	a/c	TPC files for Help and index tabs
enuLDCM*.*	a/c	Help docs in Word* and Acrobat* formats

Silent Installation

To set up a silent install of Client Manager, follow the steps below.

1. Run SETUP.EXE -r
2. Step through the install to record how the silent install should run. For example, if you want the silent install to automatically reboot at the end, you should choose to reboot after installing Client Manager.
3. Search for the SETUP.ISS file on your computer. It's usually in the Windows folder.
4. Place the SETUP.ISS file in the same folder as your other installation files (SETUP.EXE, SETUP.INS, and so on).
5. Run the silent install by running SETUP.EXE -s

   You can combine the "stealth" switch (GUI not displayed) with the silent installation switch. The example below combines both switches:

   SETUP.EXE -s stealth

Silent Installation Example

The following lines show the general content of a SETUP.ISS file. This example has been trimmed in size and is not a complete SETUP.ISS file.

```
[InstallShield Silent]
 Version=v5.00.000
 File=Response File
 [File Transfer]
 OverwriteReadOnly=NoToAll
 [DlgOrder]
 Dlg0=SdWelcome-0
 Count=6
 Dlg1=SdLicense-0
 Dlg2=SdAskDestPath-0
 Dlg3=SdComponentDialog2-0
 Dlg4=SdSelectFolder-0
 Dlg5=SdFinishReboot-0
 [SdWelcome-0]
 Result=1
 [SdLicense-0]
 Result=1
 [SdAskDestPath-0]
 szDir=C:\Program Files\Intel\LDCM
 Result=1
 [SdComponentDialog2-0]
 Common Components (HIDDEN)-type=string
 Common Components (HIDDEN)-count=
 Client - Standard Components-type=string
 Client - Standard Components-count=
 Client - DMI Components\...-type=string
 Client - DMI Components\...-count=
 Client - CIM Components-type=string
 Client - CIM Components-count=
 Client Only - Components (Hidden)-type=string
 Client Only - Components (Hidden)-count=
 Component-type=string
 Component-count=
 Result=1
 [SdSelectFolder-0]
 szFolder=Intel LANDesk Management
 Result=1
 [Application]
 Name=Client Manager
 Version=6.20.00.00
```

```
Company=Intel
Lang=0009
[SdFinishReboot-0]
Result=1
BootOption=3
```

Stealth-Mode Installation

Stealth mode disables GUI and tray icon for Client Manager. This disables all client alerts on the client computer; the alerts are sent to the admin computer only when the admin issues a reboot command to the client. The installation automatically sets the proper registry key entry.

To install Client Manager in stealth mode, use the "stealth" switch (for stealth mode). For example:

SETUP.EXE stealth

You can combine the "-s" switch (for silent install) and the "stealth" switch.

Version Information

When Client Manager Setup is run, IVERSION.EXE executes and creates a version information list for Client Manager files.

To display the version information for files most recently installed by Client Manager:

1. In the upper-right corner of the Client Manager main GUI, click **About**.
2. Click **Version Info**.

The displayed list shows the following types of information:

- Filenames for all files installed on the local computer during Client Manager Setup.
- Size in bytes of file
- Date and time of last modification
- Version information for the file, in the following format: major.minor.patch.build

If any of the files in the list have changed since Client Manager was last installed, they are displayed with an  icon, and the attribute(s) that changed are highlighted in blue.

IVERSION.EXE and User Patches

If you create a user patch of Client Manager that is not part of Client Manager Setup, you must re-run IVERSION.EXE and re-boot the user's computer. Otherwise, Version Info will not display the file changes for that patch. To run IVERSION, create an entry for it in the RunOnce Windows Registry key.

Example

The example below is a **partial** sample of the Version Info list; it does not show all the Client Manager files.

Filename	Size	Date	Version
16bitos.exe	16368	04/09/1999 15:51:18	6.00.0.0
aam.dll	24661	06/28/1999 17:33:24	6.00.0.125
ams2inst.dll	165888	05/03/1999 08:47:14	6.00.730.201
bcp.dll	32852	06/28/1999 17:34:06	6.00.0.125
bcpsnd.dll	28759	06/28/1999 17:34:28	6.00.0.125
bootstrp.dll	28760	06/28/1999 17:38:00	6.00.0.125
bootstrp.dll	28760	06/28/1999 17:38:00	6.00.0.125
cimgr.exe	143491	06/28/1999 17:28:08	6.00.0.125
cimgrldr.exe	20480	06/28/1999 17:27:50	6.00.0.125
cmanager.dll	36864	06/28/1999 17:20:02	6.00.0.125
cmsnap.dll	36964	06/28/1999 17:21:18	6.00.0.125
crmm.dll	49152	06/28/1999 17:26:16	6.00.0.125
dcsc.dll	36979	06/28/1999 17:38:36	6.00.0.125
dcse.dll	24691	06/28/1999 17:38:42	6.00.0.125
dcss.dll	24691	06/28/1999 17:38:48	6.00.0.125
dmieps.dll	28766	06/28/1999 17:41:02	6.00.0.125
dmiproxy.dll	5120	04/16/1998 08:07:34	
elogapi.dll	19456	10/15/1998 18:30:40	
emal.dll	28774	06/28/1999 17:30:42	6.00.0.125

Customizing and Integrating with Client Manager

Important Requirements for Modifying Client Manager

ImportantYou **must** follow the requirements below when customizing the functionality in Client Manager. When you customize Client Manager, you can change **only** the items described in Attribution and Customization below.

Attribution

You can use the OEMSTRINGS.INC file in Client Manager to provide the following OEM attribution strings:

- OEM link (name, URL, and tool tip) in the top bar of the main screen.
- OEM name (text only, such as "for OEM computers") in the middle bar of the main screen.
- An attribution or customer support string in the Help > About screen.

See Creating OEMSTRINGS.INC and Changing OEMSTRINGS.INC.

Customization

You can customize Client Manager Help to include your own OEM information by following the procedures in this SDK section:

- Customizing Client Manager Help

Restrictions

Do not change **anything else** in the Client Manager GUI, including the default HTML style sheet (DEFAULT.CSS). This will ensure that you keep the following elements intact:

- The location and contents of the Intel logo.

- The location and contents of the product name.
- The product's look and feel (i.e., colors, fonts, basic UI elements).

Customizing Client Manager

Creating OEMSTRINGS.INC

When you create the OEMSTRINGS.INC file, you need to place it where Client Manager Setup will find it and copy it in your installations. You can choose the following locations for OEMSTRINGS.INC:

- ldcm\wwwroot\comon
- Wherever your Setup directories are located

A sample OEMSTRINGS.INC is shown below; you can copy it, modify it, and put it in \ldcm\wwwroot. It has sections for all supported languages (Japanese, French Italian, German, Spanish, English, and simplified Chinese). The STRINGS_LOADED statement in each section refers to a corresponding #define in STRINGS.INC, which is the language the user's browser is set to.

See also Changing OEMSTRINGS.INC.

Custom Files and Client Manager Install

When you create or change files that you want to include in customized Client Manager install images, follow the steps in Adding Custom Files to Client Manager Installations.

Sample OEMSTRINGS.INC

```
#if _STRINGS_LOADED == JA
 #define OEM_LINK_NAME  Update oemstrings.inc (japanese)
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #elif _STRINGS_LOADED == FR
 #define OEM_LINK_NAME  Update oemstrings.inc (french)
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #elif _STRINGS_LOADED == IT
 #define OEM_LINK_NAME  Update oemstrings.inc (italian)
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #elif _STRINGS_LOADED == DE
 #define OEM_LINK_NAME  Update oemstrings.inc (german)
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #elif _STRINGS_LOADED == SP
 #define OEM_LINK_NAME  Update oemstrings.inc (spanish)
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #elif _STRINGS_LOADED == ZH
 #define OEM_LINK_NAME  Update oemstrings.inc (simplified chinese)
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #else
 #define OEM_LINK_NAME  Update oemstrings.inc
 #define OEM_LINK       Update oemstrings.inc with your link
 #define OEM_LINK_URL   http://update with your link
 #define OEM_LINK_TITLE Visit <your name here> on the web
 #endif
```

Changing OEMSTRINGS.INC

This topic describes changes you can make to OEMSTRINGS.INC file. The example changes below refer to the English section, but they can also be used in any language section.

OEMSTRINGS.INC #Defines and Changes

`#define OEM_LINK_NAME for OEM English computers`

This is GUI text that appears in the middle bar (blue) of the main Client Manager screen. Example change: "for Acme Brand Computers"

`#define OEM_LINK oem.com`

This is GUI text that appears in the top bar (purple) of the main Client Manager screen, to the right of "intel.com." Example change: "acme.com"

`#define OEM_LINK_URL http://www.oem.com`

This is the URL for the OEM link. Example change: http://www.acme.com

`#define OEM_LINK_TITLE Visit OEM on the Web`

This is the Tool Tip for the link. Example change: "Visit the Acme Brand Web site."

`#define OEM_SUPPORT For support call your computer manufacturer.`

This is GUI text that appears at the bottom of the LANDesk® Client Manager splash screen and the About LANDesk® Client Manager screen. Examples: "For technical support, call 1-888-555-9999" or "Customized for Acme Computers"

Removing Sections from OEMSTRINGS.INC

If you remove one or more language sections from OEMSTRINGS.INC, make sure the resulting #if, #elif, and #endif statements are properly placed to avoid syntax errors.

Customizing Client Manager Help

Overview of Help Customization

If you customize the HTML help system and plan to include your changes in a Client Manager install image for distribution, see Adding Custom Files to Client Manager Installations.

The Client Manager HTML help system is customized for client and administrator audiences and is available in seven languages. If you compare a client-only and an administrator-only (or full) installation, some of the HTML filenames may be identical, but the content will likely be different.

The major components of the HTML help system are:

- **.HTM files**All of the .HTM files start with a standard language identifier. For example, ENU is English and JPN is Japanese. All HTM files link to a cascading style sheet, HTMLHELP.CSS. These .HTM files display in the right frame of a new browser window when the user clicks Help from Client Manager. The default location for all .HTM Help files is WWWROOT\HELP. If you don't put new help files in the WWWROOT\HELP folder, you must create a relative link to the file in the HREF statements of the .TOC file.
- **.TOC files**The client and administrator installations each have their own .TOC file, named HELPTOC.TOC. These two .TOC files (named the same) are the only two .TOC file that ship with Client Manager 6.3. The default location for .TOC files is LDCM\INFO. The information in the TOC file, which includes the table of contents and index for all languages, displays in the left frame of a new browser window when the user clicks Help from Client Manager. The .TOC files also contain the context-sensitive mapping for all languages. You can:
  - Add, change, or remove topics from the existing .TOC files.
  - Change index entries.
  - Create additional .TOC files that will dynamically merge with the main .TOC file whenever the user clicks Help.
- **.CSS file**The cascading style sheet, HTMLHELP.CSS, is referenced in all of the .HTM files. If you move a .HTM file from the WWWROOT\HELP folder, you must also copy the stylesheet to the new folder or update the link.

Adding Help Topics to an Existing TOC File

If you just want to add files and don't want to modify the existing .TOC files, you can create separate .TOC files that will dynamically merge with the main .TOC file whenever the user clicks Help.

If you want to add help topics to an existing .TOC file:

1. Create a Help page (enuOEMIN.htm, for example) and put it in the LDCM\WWWROOT\HELP folder, which is the default location for all .HTM files for the help system.
2. Make a backup copy of the .TOC file you are changing in case there is a problem.

3. Open the .TOC file using a text or HTML editor. The default location for .TOC files is LDCM\INFO. If you are planning on preserving the double-byte language strings in the .TOC file, make sure that your text editor will not harm them when the file is saved.
4. For each Help page (*.htm) you created in step 1, add an [ItemX] section to either the administrator or client version of the HELPTOC.TOC, where X is the sequence number of the Help page in the Table of Contents.
5. In each [ItemX] section, specify a Group name, which is the parent container for the item, as it will appear in the left frame of the Help window. If necessary, create a new [GroupX] section in the .TOC file.
6. In each [ItemX] section, provide an ID name, which is the context-sensitive map information for the topic. The ID that you enter here corresponds to the ID in your .TPC file. See step 9 below. This makes it possible for the user to see the appropriate Help topic after clicking Help from the Client Manager console.
7. In each [ItemX] section, specify the HREFs for the English (and other language) Help files.
8. In each [ItemX] section, specify the Names for the English (and other language) Help files.
9. Edit the .TPC file for the item and specify the variable help ID for the Help page you want displayed. Add the following to the your custom .TPC file:

   ```
   <SCRIPT>
   var helpID='OEMIN';
   </SCRIPT>
   ```

   where "var" signals a variable, and "helpID" is a reserved word, required by the Client Manager 6.x Help system, and "OEMIN" is the ID of the new inventory item specified in that item's TOC file. Note that the ID must be enclosed in single quotes. The line, of course, ends with a semi-colon.
10. Consecutively renumber all [ItemX] and [GroupX] sections in the .TOC file, where X is a number from 1 to X with no gaps.
11. Test your changes by running Client Manager Help. If Help was already open, click Refresh.

Changing or Removing Entries in an Existing TOC file

To change an entry in the Table of Contents for Client Manager Help:

1. Make a backup copy of the .TOC file before you edit it in case there is a problem.
2. Open the .TOC file using a text or HTML editor. The default location for .TOC files is LDCM\INFO. If you are planning on preserving the double-byte language strings in the .TOC file, make sure that your text editor will not harm them when the file is saved.
3. Change any [ItemX] or [Groupx] entries as needed. For example, changing "NAMEEN = Fans" to "NAMEEN = AcmeFans" causes "AcmeFans" to replace "Fans" in the English Help TOC.
4. Delete any [ItemX] or [Groupx] entries that you don't want. If you delete a [GroupX] entry, make sure that you update any [ItemX] entries that referenced it.
5. Consecutively renumber all [ItemX] and [GroupX] sections in the .TOC file, where X is a number from 1 to X with no gaps.
6. When you are done, save the edited file.
7. Test your changes by running Client Manager Help. If Help was already open, click Refresh.

Changing the Help Index

To change the Help index (add, change, or remove an index entry):

1. Make a backup copy of the .TOC file before you edit it in case there is a problem.
2. Open the .TOC file using a text or HTML editor. The default location for .TOC files is LDCM\INFO. If you are planning on preserving the double-byte language strings in the .TOC file, make sure that your text editor will not harm them when the file is saved.
3. In the [IndexN] section, change the NAME, ID, and HREF statements to fit your new index entry.

   For example, changing "NAMEEN = Fans" to "NAMEEN = AcmeFans" causes "AcmeFans" to replace "Fans" in the Help Index.

   To create an additional index item, add or change the next [IndexN] section.
4. To create a secondary index item (sub-index entry indented beneath a main entry), set the ID statement so it shows the main and sub-items. The format is: ID=<MAIN-ITEM>:<SUB-ITEM>). Then create a parent statement: PARENT=<MAIN-ITEM> See the existing .TOC files for examples

of this.

5. Consecutively renumber all [IndexN] entries in the .TOC file, where N is a number from 1 to N with no gaps.
6. When you are done, save the edited file.
7. Test your changes by running Client Manager Help. If Help was already open, click Refresh.

Creating Additional TOC files

The default location for .TOC files is LDCM\INFO. However, the Client Manager help engine will scan any custom subfolders under LDCM\INFO, such as LDCM\INFO\INVENTORY or LDCM\INFO\MYTOCS, and dynamically combine all information it finds in any .TOC files into a single view when the user clicks Help.

You can name the .TOC files with custom names, such as MYTOC.TOC. The default name for the .TOC files that ship with Client Manager is HELPTOC.TOC.

Different .TOC files can reference each other. For example, an [ItemX] entry in one .TOC file can reference the [GroupX] entry in another .TOC file. Every [ItemX] in any .TOC file must have a parent group whether in the same .TOC file or another .TOC file.

To create a new .TOC file:

1. Copy an existing .TOC file and open it using a text or HTML editor. The default location for .TOC files is LDCM\INFO. If you are planning on preserving the double-byte language strings in the .TOC file, make sure that your text editor will not harm them when the file is saved.
2. Add, change, or remove [GroupX] or [ItemX] entries in the sample .TOC file you are viewing.
3. Change index entries as needed.
4. Make sure that all [ItemX] and [GroupX] and [IndexX] sections in the .TOC file are consecutively renumbered , where X is a number from 1 to X with no gaps.
5. When you are done, save the edited file.
6. Test your changes by running Client Manager Help. If Help was already open, click Refresh.

Customizing the Export Utility

To change the export file that the Client Manager administrator console acquires during its group Export operation, you change the Group Operations section of the ...\LDCM\wwwroot\Info\leftbar.ndx file. That section is shown below.

```
[GO1]
 ID=Export_CSV
 HREF=/content/export.csv
 EXTN=csv
 GUID=95A684B1-E740-11d2-B3F1-00105AA325DE
```

To specify a CSV export file other than /content/export.csv, change the HREF statement to point to it. Do not change the ID or EXTN tags; the file format for administrator export must be CSV.

Important Once you change the HREF, you must also change the GUID tag to correspond to the new HREF.

Changing the Export File

- To narrow the output of the export file, you can remove unwanted items from the /content/export.csv file and rename it.
- To add output, you can add items to the export file and point to it as described above, or you can create a "wrapper" file that uses #include statements. For example, a sample wrapper.csv file could contain the following:

```
#include export.csv // keep info from original export.csv
#include oem.csv // add OEM file with new export info
```

Important When you change or add export files, remember to change the HREF and GUID tags in ...\LDCM\wwwroot\Info\leftbar.ndx to point to the new file(s).

You must create a TPC file that does the export. The CSV is an example of how this works. The extension of this TPC file should be the extension you want to give the file. You also need to add the filename to the contents of SCRIPTMAP.INI. To access the TPC file, you must add a link to it on the export page.

Migrating and Integrating Client Manager

Integrating with Intel® LANDesk® Management Suite

Client Manager integrates with Intel® LANDesk® Management Suite 6.x through Desktop Manager. Desktop Manager enables Management Suite to launch Client Manager.

Integrating AMS²

About AMS²

AMS² is a service application that provides alert management for Windows NT* and Windows* 98, Me, and 2000. AMS² includes alert origination, alert binding, and alert handling. AMS² contains the Alert Originator module.

When an application generates an alert, it must package the alert into a structure specified by AMS². The application initializes the alert types and issues alerts though API functions. Once an application has issued an alert, the AMS² Originator Manager finds any binding associated with the alert. When the associated bindings for the alert are found, the alert and its bindings are forwarded to the appropriate Alert Handler for execution. The Alert Handler is responsible for executing the alert and logging it in an AMS² alert log.

A binding is an action associated with an alert, such as a page, e-mail, or SNMP trap. AMS² provides a Binding Console that allows users to bind actions to alerts issued by the integrated application. Client Manager does not provide AMS² or its Binding Console. Users who want to bind alerts issued by Client Manager must do so from Desktop Manager or Intel® LANDesk® Management Suite. Alerts issued with no bindings are simply logged in the AMS² alert log.

Client Manager does not provide an alert handler; instead, it relies on the alert handler installed with Management Suite.

Important Client Manager installs AMS2.CFG and AMS2inst.DLL; it does not install any other AMS² software. Client Manager is compatible with AMS² software installed only by Management Suite or other LANDesk products. Without the AMS² components installed on the system, Client Manager does not attempt to issue AMS² alerts. Client Manager automatically detects the presence of AMS² and subsequently registers and issues AMS² alerts.

AMS² Integration

The integration of AMS² with Client Manager accomplishes the following goals:

- AMS² boosts Client Manager's overall product value.
- Client Manager can issue alerts through AMS² that end users can bind to special actions. These actions are not internally supported by Client Manager.

To successfully integrate Client Manager with AMS² for alerting functionality, install AMS² on the client computers. Client Manager can then issue AMS² alerts.

Integrating SNMP

SNMP Integration

SNMP enables Client Manager to integrate with Enterprise Management applications such as HP OpenView*, Microsoft SMS, CA Unicenter* TNG, and IBM Tivoli* TME 10 NetView.

An SNMP trap is an unsolicited indication of an event. The trap helps Enterprise Management applications receive and manage DMI indications from Client Manager. However, because SNMP traps use the User Datagram Protocol (UDP) to transmit messages, the traps are not a guaranteed delivery; they are a best-effort delivery system.

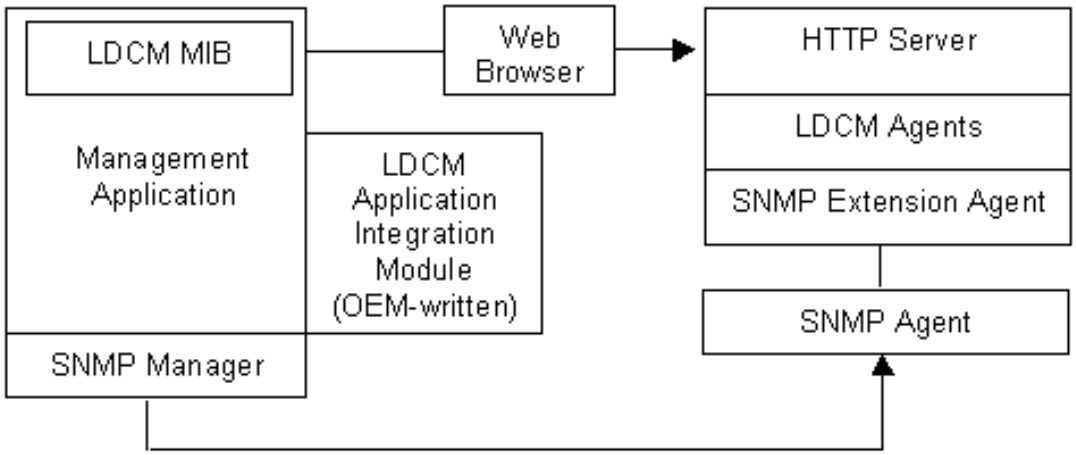


Figure: Third-party Application Integration Architecture

Once an application has received SNMP trap information, it can load an HTML page that specifies the URL of the computer to be managed, then browse the information on the managed computer.

Integration Requirements

To successfully integrate with Client Manager SNMP trap functionality, you need to install instrumentation that generates events. When the System Space Manager receives events, SNMP traps are generated, based on the user's configuration.

For information on viewing and configuring SNMP traps, see the Client Manager Help.

Client Manager Information Available through SNMP Gets

Client Manager's SNMP agent supports several SNMP GETs. These can be used to obtain the Client Manager version, HTTP port, and current computer health of a remote client. These variables are defined in the Client Manager MIB.

The variables are useful in several situations. For example, you can do an SNMP GET on a group of computers to determine which versions of Client Manager each computer is running, so you can do upgrades. You can also do an SNMP GET on a group of computers to determine which computers are in a critical state by obtaining their computer health.

Plug-in

To interface Client Manager with a third-party application, a plug-in can be used, although it is not required that the implementation be done through a plug-in. The plug-in integrates with the third-party application and performs SNMP GETS on Client Manager managed computers.

Client Manager SNMP Extension Agent

Note Client Manager and the Microsoft Windows* SNMP agent must be installed on the computer to be managed before the SNMP extension agent can be used.

SNMP traps are supported on the Windows platform by Extension Agent DLLs. These DLLs are loaded by the SNMP service upon startup. In Windows NT*, this is handled by the Service Manager.

The SNMP extension agent passes trap information from the managed computer's agents to its SNMP agent. The traps are currently defined as "health traps" that relate to the health of the computer or the threshold of events. These traps are sent only when a health event changes. The information contained in the trap conveys the new health status. The health changes when user-definable thresholds are crossed in health parameters.

LDCM.MIB defines the information that the SNMP extension agent can provide and the traps that can be generated. The third-party application can use the Client Manager MIB file to format incoming trap information.

Format for SNMP Traps

SNMP Traps generated by Client Manager have the following format:

Field	Description
Computer Name	Computer that generated the trap

Date and Time	Date and time the trap was generated
Type	Type of DMI indication
Symptom	Description of the DMI indication
Severity	Severity level (OK, Non-Critical, Critical)

For a detailed description of the format, see Client Manager MIB Definition.

Client Manager MIB Definition

The LDCM.MIB file defines the Client Manager SNMP trap Protocol Data Unit (PDU).

```
LDCM-MIB DEFINITIONS ::= BEGIN

 IMPORTS
   enterprises, Counter, TimeTicks
        FROM RFC1155-SMI

   DisplayString
        FROM RFC1213-MIB

   TRAP-TYPE
        FROM RFC-1215

   OBJECT-TYPE
        FROM RFC-1212;


 -- Intel OIDs

 intel            OBJECT IDENTIFIER ::= {enterprises 343}
 products         OBJECT IDENTIFIER ::= {intel 2}
 network-products OBJECT IDENTIFIER ::= {products 5}
 lanDesk          OBJECT IDENTIFIER ::= {network-products 1}
 ldcm             OBJECT IDENTIFIER ::= {lanDesk 2}


 -- Get Information
 version          OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..10))
                        ACCESS     read-only
                        STATUS     mandatory
                        DESCRIPTION "Intel® LANDesk® Client Manager's version."
                        ::= {ldcm 1}

 pcHealth         OBJECT-TYPE
                        SYNTAX     INTEGER { unknown(0), normal(1), warning(2),
critical(3) }
                        ACCESS     read-only
                        STATUS     mandatory
                        DESCRIPTION "Intel® LANDesk® Client Manager's current PC
Health."
                        ::= {ldcm 2}
 httpPort         OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..10))
                        ACCESS     read-only
                        STATUS     mandatory
                        DESCRIPTION "Intel® LANDesk® Client Manager's management HTTP
port."
                        ::= {ldcm 3}
```

```
 -- Trap Information

 computerName     OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..20))
                        ACCESS     not-accessible
                        STATUS     mandatory
                        DESCRIPTION "The source computer that originated the event."
                        ::= {ldcm 101}

 eventTime      OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..255))
                        ACCESS     not-accessible
                        STATUS     mandatory
                        DESCRIPTION "The date and time the trap was sent from the source
computer in GMT."
                        ::= {ldcm 102}

 eventType      OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..255))
                        ACCESS     not-accessible
                        STATUS     mandatory
                        DESCRIPTION "The type of event."
                        ::= {ldcm 103}


 eventSymptom     OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..512))
                        ACCESS     not-accessible
                        STATUS     mandatory
                        DESCRIPTION "The description of the event."
                        ::= {ldcm 104}

 eventSeverity    OBJECT-TYPE
                        SYNTAX     DisplayString(SIZE(0..255))
                        ACCESS     not-accessible
                        STATUS     obsolete
                        DESCRIPTION "This is obsolete - check the specific SNMP Trap
type for the severity."
                        ::= {ldcm 105}



 -- Trap Macros

 ldcmUnknownTrap  TRAP-TYPE
                        ENTERPRISE  ldcm
                        VARIABLES   {computerName, eventTime, eventType, eventSymptom}
                        DESCRIPTION "Intel® LANDesk® Client Manager has reported an
Unknown event."
                                #SUMMARY "LANDesk® Client Manager has reported an event
on the computer %s: %s"
                                #ARGUMENTS {0, 3}
                                #SEVERITY NORMAL
                                #TIMEINDEX 1
                                #STATE OPERATIONAL
                                #CATEGORY "Status Events"
                        ::= 1
ldcmInfoTrap     TRAP-TYPE
                        ENTERPRISE  ldcm
                        VARIABLES   {computerName, eventTime, eventType, eventSymptom}
                        DESCRIPTION "Intel® LANDesk® Client Manager has reported an
Informational event."
                                #SUMMARY "LANDesk® Client Manager has reported an event
```

```
on the computer %s: %s"
                                #ARGUMENTS {0, 3}
                                #SEVERITY NORMAL
                                #TIMEINDEX 1
                                #STATE OPERATIONAL
                                #CATEGORY "Status Events"
                        ::= 2
ldcmOkTrap       TRAP-TYPE
                        ENTERPRISE  ldcm
                        VARIABLES   {computerName, eventTime, eventType, eventSymptom}
                        DESCRIPTION "Intel® LANDesk® Client Manager has reported a
severity OK event."
                                #SUMMARY "LANDesk® Client Manager has reported an event
on the computer %s: %s"
                                #ARGUMENTS {0, 3}
                                #SEVERITY NORMAL
                                #TIMEINDEX 1
                                #STATE OPERATIONAL
                                #CATEGORY "Status Events"
                        ::= 3

ldcmWarningTrap  TRAP-TYPE
                        ENTERPRISE  ldcm
                        VARIABLES   {computerName, eventTime, eventType, eventSymptom}
                        DESCRIPTION "Intel® LANDesk® Client Manager has reported a
severity Warning event."
                                #SUMMARY "LANDesk® Client Manager has reported an event
on the computer %s: %s"
                                #ARGUMENTS {0, 3}
                                #SEVERITY MINOR
                                #TIMEINDEX 1
                                #STATE OPERATIONAL
                                #CATEGORY "Status Events"
                        ::= 4

 ldcmCriticalTrap TRAP-TYPE
                        ENTERPRISE  ldcm
                        VARIABLES   {computerName, eventTime, eventType, eventSymptom}
                        DESCRIPTION "Intel® LANDesk® Client Manager has reported a
severity Critical event."
                                #SUMMARY "LANDesk® Client Manager has reported an event
on the computer %s: %s"
                                #ARGUMENTS {0, 3}
                                #SEVERITY CRITICAL
                                #TIMEINDEX 1
                                #STATE DEGRADED
                                #CATEGORY "Status Events"
                        ::= 5
ldcmFatalTrap    TRAP-TYPE
                        ENTERPRISE  ldcm
                        VARIABLES   {computerName, eventTime, eventType, eventSymptom}
                        DESCRIPTION "Intel® LANDesk® Client Manager has reported a
severity Fatal event."
                                #SUMMARY "LANDesk® Client Manager has reported an event
on the computer %s: %s"
                                #ARGUMENTS {0, 3}
                                #SEVERITY CRITICAL
                                #TIMEINDEX 1
                                #STATE DEGRADED
                                #CATEGORY "Status Events"
                        ::= 6

 END
```

General Reference

CIM Connection System Overview

Web-based Enterprise Management (WbEM) is an industry initiative to develop a standardized technology for accessing management information in an enterprise environment. WbEM is designed to implement the Common Information Model (CIM), a schema of managed objects. WBEM includes a central component called the CIM Object Manager (CIMOM) (see CIM Data Access Component).

The CIM Connection System (CCS) connects Client Manager with a CIM Object Manager.

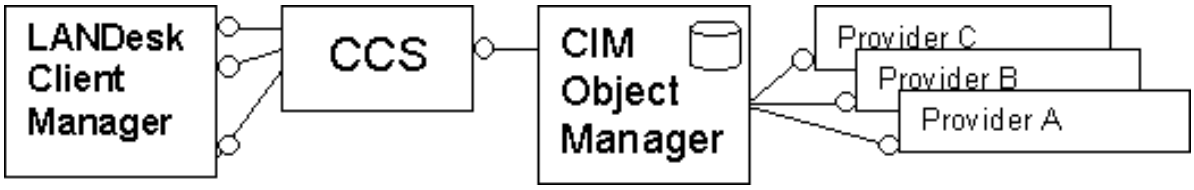


Figure: General CCS use model

The CCS encapsulates CIMOM functionality and abstracts management applications from details of the CIMOM.

DMI Connection System Overview

The DMI Connection System (DCS) is a collection of modules that provide DMI v2.0s connection functionality for Client Manager.

The DCS components comprise a system of data pathways between the client application (Client Manager) and the Intel implementation of the DMI specification, called the Windows 32-bit Service Provider (Win32SL).

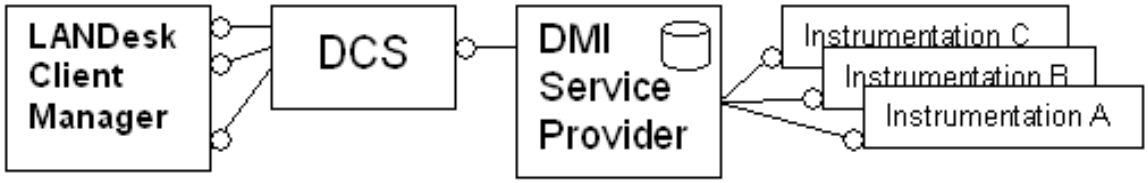


Figure: General DCS use model

Client-to-Administrator Communication

Client-to-Administrator Communication

Occasionally connected computers (dial-up connections or occasionally powered computers on the network) pose a special problem for the discovery process. These nodes might not be available when the administrator node performs its discovery process. Before an Client Manager administrator node can manage a client node, the administrator must first know the client exists.

To solve this problem, the client can help the administrator node by communicating with it. This communication enables the client to

- Announce its presence to the administrator node.
- Send information the administrator node might need (such as the client IP address and port information) in order to connect to the client.

The figure below shows the general communications flow, moving from right to left.

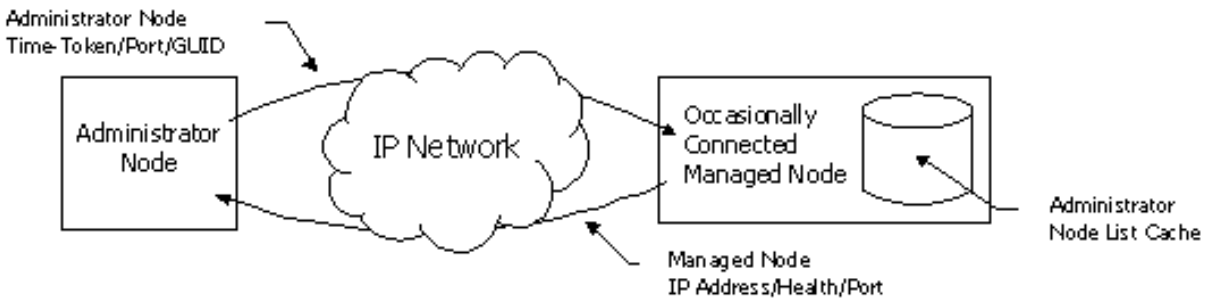


Figure - Client-to-administrator communication flow

Expiration of Entries

The client uses a mechanism to expire administrator node cached entries when it is unable to contact the administrator node in an acceptable amount of time. Each administrator node specifies the frequency with which the client must "check in." Once time has expired with no contact from that administrator node, its entry will be removed from the client's list of known administrator nodes.

Managed Node Announcement

The managed node tells the administrator about its presence on the network when it determines that a time-token received from an administrator node needs to be refreshed.

The managed node announcement includes information used by the administrator node to locate the managed node. This information includes the managed node IP address.

Administrator Identification

The administrator node sends an identification packet to the managed node when one of the following conditions is met:

- The administrator node is responding to a managed node announcement.
- The administrator node discovers the managed node for the first time.
- The administrator node initially comes online after being disconnected from the network.

The administrator node identification packet will include information used by the managed node to determine administrator node uniqueness, location, and time-token (indicating the time to live).

Identification Aging

The managed node maintains a list of time-tokens from administrator nodes it has been in contact with. The time-token is refreshed at specific intervals prior to the time-token's expiration date in order to age each time-token. The basic algorithm for time-token aging is:

- The managed node attempts to refresh the time-token when 1/2 of the time-token's lifespan has elapsed.
- If the managed node is unable to refresh the time-token, the lifespan is halved.
- If the time-token has expired, the managed node will make one last attempt to refresh the time-token. If the time-token cannot be refreshed, it and all associated administrator node information are removed from the managed node's cache of administrator nodes.

Discovery Technologies

PDS

About PDS and PDS2

PDS (Ping Discovery Service) uses protocols to discover and communicate with remote applications on the network. PDS is included only for backwards compatibility with earlier clients.

PDS2 is very similar to PDS. It can coexist with PDS on the same computer because PDS2 is implemented as .DLLs instead of services. Client Manager uses PDS2 because it takes less disk and memory space and

it makes new features available.

Discovery

Discovery is a client/server process where the client and server nodes can be of any type, not just file servers and application servers. The client side of an application initiates discovery by sending ping packets to the server (discovered) side of the application. The server side responds by sending back pong packets. These packets contain enough information to establish communications between the client and server sides of an application. (Do not confuse the client and server sides of an application with Client Manager's clients and administrators.)

Discovery is implemented in these ways:

- As a service for Windows NT*, Windows* 2000, and XP
- As an executable for Windows 98 and ME

Discovery Functions

Discovery performs these functions:

- Opens well-known communication channels and waits for pong packets.
- Waits for registration and de-registration messages.
- Consults its internal table of registered applications when a ping packet is received. If the application is found, a pong packet is formatted and sent back to the client.
- Handles the sending of ping packets and receiving of pong packets.

The discovery process creates threads to listen for pong packets. These threads are part of the client application, with memory allocated from the application's memory pool. The callback occurs in the context of the client application (one of its threads). The threads remain active until the discovery session ends.

Ping and Pong Packets

Sending Ping Packets

One application query can be sent per ping packet. Ping/pong packets are sent over UDP/IP. The application communicates with PDS through APIs and callback functions.

The client side builds a list of targets for ping packets. It scans this list and sends ping packets to the potential targets.

Receiving Pong Packets

If the target application is registered with PDS and found on the server side, discovery responds with a pong packet. Otherwise, the ping packet is discarded, and no response is returned.

Pong packets are received asynchronously; the client side should not wait for a pong, as the target may not respond for some reason. Because packets may be lost in transmission, the target list is scanned more than once. However, once a pong has been received for a target, that target should be flagged as "discovered" so those unnecessary ping and pong packets won't be sent in the additional passes.

The client retains a list of targets that respond. Future scans use this potential target list to greatly reduce network traffic and speed discovery. That way, a full scan of all possible targets is needed only the first time Client Manager starts and after new potential targets have been installed. A quick scan of targets that have previously responded is otherwise sufficient.

Node Discovery Engine (NDE)

About NDE

To manage nodes on the network, the administrator version of Client Manager uses the Node Discovery Engine (NDE) to do the following:

- Discover new manageable nodes on a network.
- Manage multiple nodes on the network through a managed node list.
- Add or remove specific nodes from the node list.
- Gather node state information
- Access connection-oriented and status information about all nodes in the node list.
- Allow safe asynchronous access to the node list for all instances of NDE objects.

The NDE is the single endpoint for obtaining connection-oriented and status information about any managed node. The NDE does not provide its own user interface, but it safely interacts with the node list on the administrator computer. The general NDE use model is shown below.

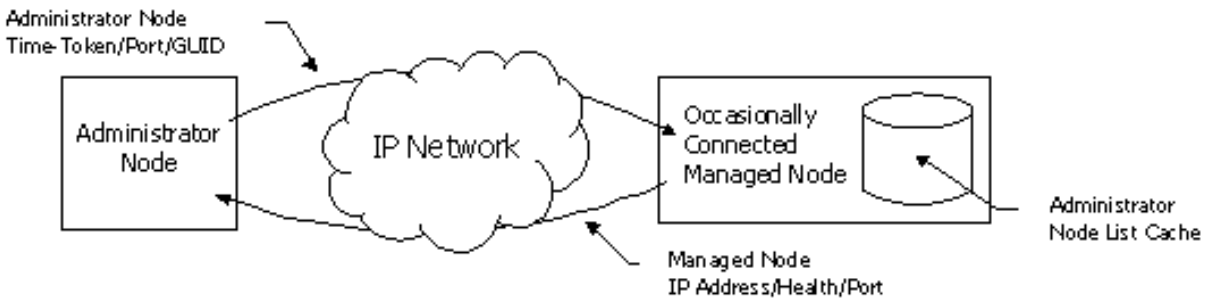


Figure - General NDE use model for Client Manager

NDE Discovery

A manageable node is discovered when:

1. The NDE sends a ping packet to the managed node.
2. The managed node uses the Ping Discovery System (PDS) to determine the associated application pong data.
3. The managed node returns the appropriate pong packet.

The NDE can originate two discovery methods:

- Node List Update, where the NDE tries to discover nodes that have already been discovered. This is used primarily to update a node's connection and status information.
- Broadcast Discovery, where the NDE attempts to discover nodes by a ping packet broadcast, as well as by pinging nodes already in the node list.

NDE and Administrator Components

The following Client Manager administrator components are consumers of the Node Discovery Engine (NDE):

- The NDE Data Access Component (NDEDAC)
- Notification Monitor

When NDE processes a pong packet, a point-to-point Client Manager back-end connection is established with the managed node. The connection forces the administrator node time-token and GUID into the managed node administrator list. See also Client-to-Administrator Communication and related topics.

NDEDAC

The NDEDAC provides a common interface into the node list, via the NDE. This enables web-based management as part of the Client Manager administrator package. The Template Parser Component (TPC) loads the NDEDAC to merge node list information with an HTML display template. The merged data and display template are sent to back to the Web Browser via the HTTP Server. See also NDEDAC Overview and related topics.

Notification Monitor

Notification Monitor displays notifications to the user.

Node List Management and Information

The NDE enables large node list additions via broadcast discovery. The NDE also allows individual nodes to be added and removed from the node list. This means the administrator can choose which nodes appear in the node list. Client Manager executes the node list management requests for all nodes in the list.

Node Information

Below are the kinds of data retrieved and cached from the pong packet of the discovered node. All this data is considered status information, unless indicated as connection-oriented information below. This data is available even when the computer is shut off.

1. A GUID that uniquely identifies the node in the list (connection-oriented).
2. The FQDN (Fully-Qualified Domain Name).
3. The name associated with the node on the network. This enables other Client Manager agents to communicate with any given node via name resolution protocols in the operating system.

4. A network (IP) address (connection-oriented), used to facilitate communications with any given node.
5. The node's MAC (network card) address. This enables the NDE client application to remotely wake up a powered-down node. Wake-On-LAN technology uses the MAC address to target and wake up nodes via the LAN.
6. An HTTP Server port and point-to-point communications port (connection-oriented). The HTTP Server port enables the administrator Web Interface to access remote node information through the Internet. The point-to-point communications port is used by other Client Manager agents establish Client Manager back-end communications between the administrator node and other managed nodes.
7. The current health state of the node.
8. Information indicating whether the node supports Wake-On-LAN* technology.
9. The node's operating system type.
10. The node's Client Manager version information.

Node State Information

A node can be in any one of the states described below. Normal, Warning, and Critical imply a node that it is available for management.

- **Unavailable**Upon the last discovery attempt, the node did not respond and cannot be awakened.
- **Unavailable but supports Wake-On-LAN technology**This type of node is wakeable via Wake-On-LAN* technology. The pong packet indicates whether the client supports Wake-On-LAN technology. This information helps the user determine what unavailable nodes in the node list can be awakened.
- **Normal health**The node is operating within normal tolerances.
- **Warning**The node is unhealthy, operating at a smaller degree outside of specified tolerances.
- **Critical**The node is unhealthy, operating at a smaller degree outside of specified tolerances.

Pong Packet Information

Pong packet data is available only when the computer is on:

- Timestamp, relative to Greenwich Mean Time, indicating when the node last went online.
- Operating system type and version information. This data enables the end-user to quickly identify which nodes contain what type and version of operating systems.
- Version of Client Manager installed on the node.

Instrumentation, ASIC, and BIOS Reference

Heceta Tachometer Fans

About Tachometer Fans

Client Manager displays information on up to three tachometer fans in a computer chassis. Tachometer fans have another wire output taken directly from the fan's coil driver circuits. The output signal is routed to an Heceta chip that tracks the fan's speed.

The Client Manager instrumentation code:

1. Configures the Heceta chip the first time it is booted on a computer
2. Gets dynamic fan speed information from the ASIC
3. Relays fan problem alerts back to the user and/ or administrator.

Because Client Manager can be used with Intel or non-Intel motherboards, some hardware and software issues arise with the Heceta-compliant ASIC. This SDK discusses tachometer fan issues; the information might be useful to you when designing motherboards, specifying fans, or building custom images of Client Manager.

Note Because the Heceta-compliant ASIC fans are not meant to be a high precision device, don't interpret their indications into Client Manager as exact. The Heceta-compliant ASICs do alert the user when a parameter is dangerously out of range.

Heceta Tachometer Fan References

The following references were used for the Heceta information:

- *Heceta Head Target Specification*, Edward L. Davis, Revision 1.5c, July 30, 1996.
- *LM78 Microprocessor System Hardware Monitor*, National Semiconductor, September 30, 1996.
- Intel LM78 code, Revision 1.8, May 20, 1996.
- *W83781D H/w Monitoring IC*, Winbond Electronics Corp., Oct. 1997.
- *ADM9240 System Hardware Monitor*, Genesys Logic, Oct. 1997.

Tachometer Fan Requirements

For optimal performance with Heceta-compliant ASICs, fans should have the following characteristics:

- A tachometer lead (required)
- A nominal RPM between 1000 and 8800 RPM (required)
- A symmetrical tachometer wave form around the voltage midpoint

Tachometer circuitry designs are highly fan-specific. Changing a fan in the power supply or on the CPU might require changing tachometer circuitry as well. It is possible to design slightly higher-cost tachometer conditioning circuitry that can work with virtually any model of dual-pulse tachometer fan. For more information, contact your Intel Technical Marketing Engineer.

Tachometer Fan Duty Cycle

The Heceta-compliant ASIC internal clock counts the amount of time the fan pin is held high. This poses a hardware limitation: there is no way to measure the duty cycle of a tachometer fan. Client Manager assumes the duty cycle to be 50%, and that the cycle is centered around the 1.4-volt threshold. If this is not the case, RPM measurements will be wrong.

If there is a 40% duty cycle, the voltage stays above 1.4 for 40% of the time, so Client Manager reports a faster RPM than there actually is. Conversely, if there is a 60% duty cycle, the speed reported by Client Manager is slower than actual.

Fan Speed and Voltage Wave Pulses

Heceta-compliant ASICs through version 5 use their 22.5 kHz internal clock to time each fan rotation. The clock is software/hardware divisible by 1, 2, 4 or 8, allowing for fans of different speeds. Heceta 6-compliant ASICS use a 90 kHz clock and do not use a divisor setting since the fan tachometer values are stored in a word rather than a byte.

Conditioned tachometer fan voltage patterns enter the Heceta-compliant ASIC fan through their respective

input pins. When the fan turns, the signal is a train of square-wave voltage pulses. Because the fans are designed for “dual-pulse” tachometer fans, the voltage goes high twice per RPM. The Heceta-compliant ASIC fans have an on-chip oscillator clock running at 22.5 kHz. An 8-bit digital counter for each input pin counts the clock cycles (“ticks”) during the periods when the fan tachometer wave pulses are high. The counter resets automatically when the signal goes low. The clock tick total is stored in Current Value Registers 28, 29, and 30h. The Heceta-compliant fans store the number of clock ticks that occur during half a rotation; the actual number of RPMs is neither directly measured nor stored.

Component Instrumentation Functionality

Component instrumentation is instrumentation that mediates between the Heceta-compliant ASIC and the Component Interface of Client Manager’s Service Provider. The first time Client Manager is booted on a new motherboard, Component instrumentation sets up the ASIC with the following fan values and writes them to the fan MIF:

- Nominal RPM
- Threshold RPM
- Threshold Minimum RPM
- Fan Setup Registers

Component instrumentation gets these values from your Hardware Setup file or from its own auto-detect algorithm.

Using Auto-Detect Fan

Auto-detect is enabled and component instrumentation attempts to auto-detect them. With a dual pulse fan, the minimum detectable and usable RPM is about 1000; with a single pulse fan, about 2000. The resolution of the RPM reduces by a multiple of 4 each time the fan speed doubles. If the counter is at 250, a +/-1 count means a difference of about +/-5 RPM. At 125 counts, a +/-1 count means about a +/-20 RPM difference; at 64 counts, the difference is about +/-82 RPM. Auto-detect sets the detected counts as high as possible to give the least amount of error, while allowing sufficient counts for the threshold values to operate correctly.

The auto-detect feature does the following:

1. Reads the number of clock-ticks in the Heceta-compliant ASIC’s Current Value register.

   If a particular fan is not present, its tachometer input will always be high and a divide-by factor will have no effect. In that case, component instrumentation assumes that the fan is not present and write that information to the MIF database.

   If a fan is present, component instrumentation recursively calls its divide-by 1/2/4/8 function to adjust the tick count so the register won't overflow. The Current Value register will then contain a realistic count. (**Does not apply to Heceta 6**)
2. Stores the correct divisor in the Fan Setup register. (**Does not apply to Heceta 6**)
3. Multiplies the number of adjusted clock ticks in the Current Value register by .8 to obtain the Threshold Minimum clock tick value.
4. Stores the Threshold Minimum clock tick value in the Heceta-compliant ASIC’s Threshold Minimum Value register.

After auto-detect finishes, Client Manager turns the feature off.

Fan Instrumentation and the Computer Health Indicator

When a user double-clicks on the Computer Health Indicator icon, component instrumentation calculates a fan’s nominal RPM from Current and Setup register values to convert clock-tick values into approximate RPMs for display.

Note All calculated RPM values are approximate.

Adding, Removing, or Replacing Fans after Initial Boot

Component instrumentation sets up the Heceta-compliant ASIC and the static fan MIF the first time Client Manager runs, but not thereafter. Therefore, all fans must be properly installed before Client Manager runs the first time. If a fan is installed afterwards, Component instrumentation still reads the static fan MIF, which indicates that the fan is not present. This means component instrumentation won’t interrogate Heceta-compliant ASIC registers for that fan.

If a fan is removed after Client Manager has booted the first time, Client Manager will signal that the fan has

stopped working the next time the computer boots. It won't recognize that the fan is missing, since the fan MIF group is static and only set up the first time Client Manager boots.

If a replacement fan has a different tachometer, the Heceta-compliant ASIC's Divide-By, Current, and Threshold Minimum values can be wrong for the new fan and generate incorrect data. The MIF data for that fan may also be incorrect.

ImportantWhenever you add, change, or remove a fan or processor, you need to update the registry with the new settings. If you do this, you will not need to re-install Client Manager. Be sure to re-enable auto-detection in Hardware Setup if necessary.

Reporting Event Log Information

Single-bit ECC Errors

Client Manager compares the count of single-bit ECC errors to a threshold set into the Windows registry from the Hardware Setup configuration. If the count is greater than the threshold, Client Manager generates a Critical DMI Parity Error event. If the count is not zero, but still under the threshold, Client Manager generates an informational DMI Parity Error event, although by default DMI informational errors are not displayed or logged.

Multi-Bit ECC Errors

Multi-bit ECC errors cause the computer's hardware to crash. Client Manager, however, usually has time to generate a "non-recoverable" DMI event what will be logged.

POST Errors

Client Manager will generate a DMI event. The message will read "An unknown POST error has occurred." Client Manager does not support the optional Log Variable Data field.

Intel OPSD BIOS Custom Type, Virus Detect

Client Manager will generate a DMI critical event if a virus is detected in the BIOS.

Event Log Viewer Enhancement

Client Manager enables the viewing of the BIOS event log for computers with BIOSes that support event logging. The event log can be viewed from either the client or administrator console.

On the Client Manager BIOS page, the event log is displayed as a table that displays the type of event, date and time of the event occurrence, and any supplementary data. This enhancement was created in accordance with the SMBIOS 2.3.1 specification.

Event Types and Data Format

The BIOS event log viewer supports 24 types of event log entries (see Event Log Types in the SMBIOS specification). Entries can be errors (such as POST or ECC) or general information (such as system configuration changes). The viewer also supports seven different methods for formatting detailed data related to the entry (see Event Log Variable Data Format Types in the SMBIOS specification).

The BIOS chooses how to display details for each event type through descriptors in SMBIOS structure 15. For example, a single-bit ECC memory error (event log type 1) can use a multiple-event counter (format type 2) to show the number of errors that have occurred. POST errors (event log type 8) and system management errors (event log type 16) can use special formatting types to display details. POST errors can supply a bitmap in the first two DWORDs of the data portion of the entry. System management notifications can use the first DWORD in the data portion of the entry to describe temperature, voltage, or cooling threshold violations.

Clearing Event Log Information

The BIOS event log holds system health data, such as POST errors and memory ECC errors. Client Manager 6.3 can clear the BIOS event log remotely from an administrator console.

To facilitate the clearing, the following components were changed:

- CISMBIOS driver (CISMBIOS.VXD in Windows* 98 Second Edition and Me and CISMBIOS.SYS in Windows NT* and Windows 2000/XP)
- GUI for the BIOS page
- String table
- New DMI instrumentation for Intel|System Management BIOS|

Enabling BIOS Event Log Clearing

By default, functionality to clear the event log is disabled. To enable it, set the following registry key value to 1:

Key: HKEY_LOCAL_MACHINE\SOFTWARE\Intel\LANDesk\Client Manager\CurrentVersion

Value Type: DWORD

Value: EnableBIOSEventLogClearing

Important Attempting to read or clear event log information on a BIOS that does not support the event log can cause crashes or unpredictable results.

Disabling Event Log Reading

Some BIOSes have trouble reading the event log. This is usually because they store the event log in General Purpose Non-Volatile memory (GPNV), or at least report through structure Type 15 that they store the event log in GPNV. This can cause crashes or errors on 32-bit Windows systems, some of which have garbage in the memory address pointed to by Type 15.

To avoid these problems, you can disable all event log access by setting the following registry key value to 1. (Disabling this feature usually removes the BIOS from the Health section in the GUI.)

Key: HKEY_LOCAL_MACHINE\SOFTWARE\Intel\LANDesk\Client Manager\CurrentVersion

Value: NoEventLogSupport

Value Type: DWORD

Important Attempting to read or clear event log information on a BIOS that does not support the event log can cause crashes or unpredictable results.

Instrumentation and GUI

Instrumentation Design and Implementation

In IOSystem.mif , the Intel|System Management BIOS| group has been updated. Attribute 5 is an enum that outlines the specific method the driver implements to clear the BIOS event log. Attribute 6 is a read-write type. If the handler function in the instrumentation is passed a non-zero value, the instrumentation directs the driver to clear the event log using one of the two new interfaces.

GUI Design and Implementation

If a POST error is detected (or anything is contained in the event log), a new button appears on the BIOS health page (see the figure below). Though the BIOS may support clearing the event log, the button does not appear unless the Health state is Warning or Critical. When the user clicks the button, the DMIDAC is directed to write a non-zero value to attribute 6 in the Intel|System Management BIOS| group, initiating the entire event log process.

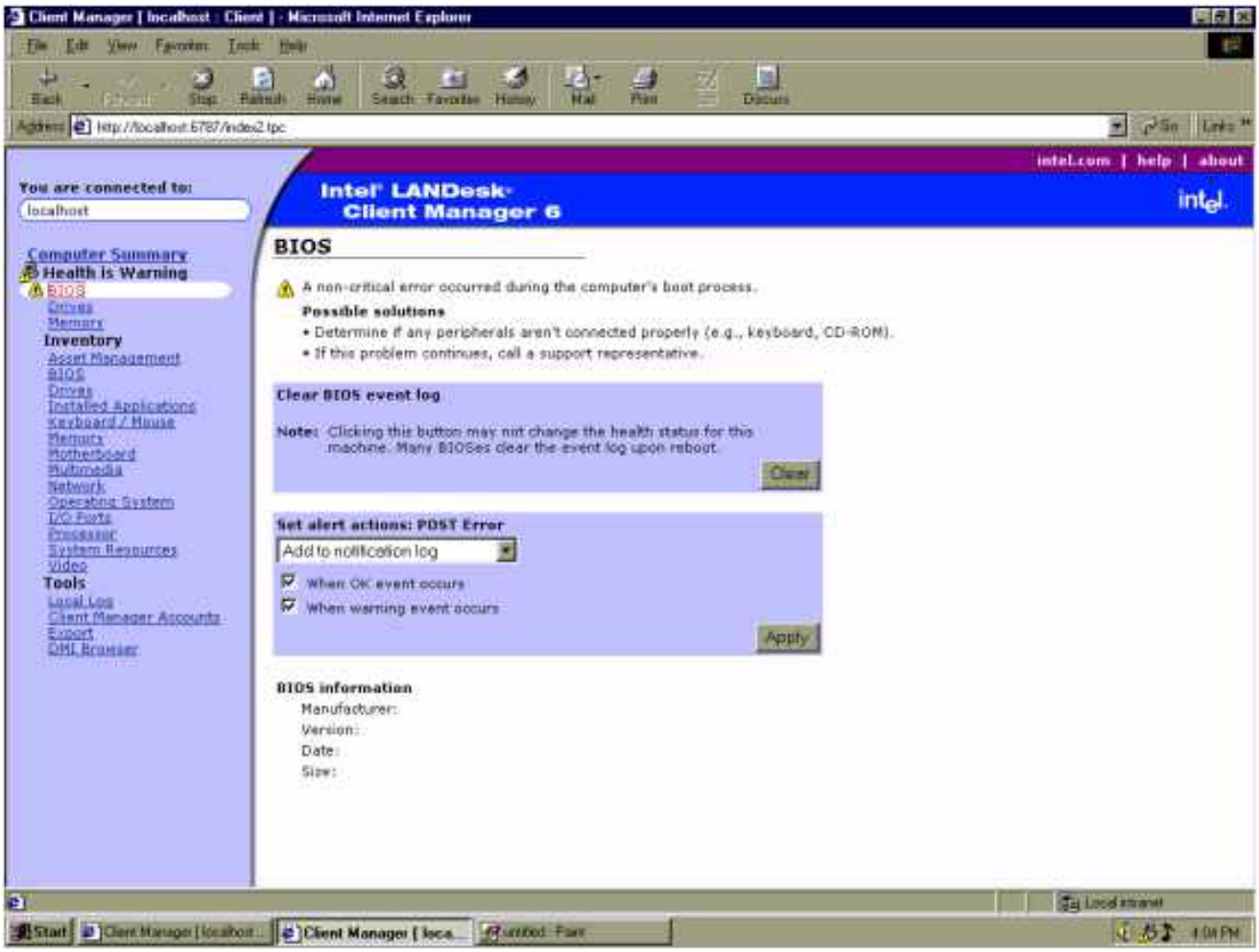


Figure - BIOS page GUI in Client Manager 6.3

Alert on LAN Support

Client Manager Support for Alert on LAN

Client Manager 6.3 facilitates Alert on LAN* functionality by:

- Supporting the client (sending side) configuration via the Client Manager GUI.
- Displaying Alert on LAN messages on the Client Manager administrator (receiving side) console.

Note The Alert on LAN agents that Client Manager 3.3x supported are not forward-compatible. You must upgrade to the new Alert on LAN included with Client Manager 6.3 to communicate with Client Manager 6.x proxies and consoles.

Alert on LAN II Support

What's New in Alert on LAN II

The Alert on LAN* II ASIC is a third-generation management device. It works with the 82559 Fast Ethernet Multi-function PCI Controller and an environmental integrated circuit to achieve advanced manageability support. These three components combined provide the management interface between a management server and managed clients.

Alert on LAN II contains the following new features.

- New alert event (CPU2 missing). Alert on LAN II now supports two processors

- New event timer (SNMP Time Alive Timer)
- The E-PROM size has grown from 128 bytes from 256 bytes
- APM support (power-down, power-up, reset system)

MIF Changes for Alert on LAN II

A new MIF file for Alert on LAN II will need to be created. It will differ from Alert on LAN I with the following proposed group changes, and many additional attribute changes.

New Groups:

- Intel|Alert on LAN Control|001 - Enumerating, enabling/disabling , executing of Control Functions/Client Capabilities such as reset, reboot, CMOS clear.
- Intel|Alert on LAN Events|001 - The events that occur on the client are now available through an interface similar to the proxy.
- Intel|Alert on LAN Configuration|002
- Intel|Alert on LAN Alert Settings|002
- Intel|Alert on LAN Proxy Server Configuration|002
- Intel|Alert on LAN Capabilities|001
- Intel|Alert on LAN Subcapabilities|001

Removed Groups:

- Intel|Health Contributor|Alert on LAN EEPROM Status|001
- Intel|EventGeneration|IntelAlert on LAN EEPROM Status|001,
- Intel|Alert on LAN System Alert Mapping|001

Features and Benefits of Alert on LAN II

The features and benefits of Alert on LAN* II are described below.

Hardware alerts in OS independent power states:

- Temperature
- Fan
- Voltage

Security alerts:

- Processor missing
- Chassis intrusion
- LAN leash tamper (link loss)

OS alerts through watchdog support:

- OS hang
- Boot failure
- Presence heartbeat
- System Management Interrupt (SMI) support
- Supports advanced manageability beyond Wired for Management (WfM) initiative
- Identifies problem systems
- Increases system protection and asset management
- Allows OEM system differentiation

Receive functionality:

- Alert acknowledgment
- Presence ping support
- Remotely managed console response

SMBus master capabilities:

- SMBus polling support for SMB and I 2 C** sensors and devices
- SMBus write support for reset and power options
- Interfaces with SMBus-compliant devices

Advanced Configuration and Power Interface

- (ACPI) state indication
- Provides detailed computer status for trouble-shooting
- Advanced Power Management (APM) support supports legacy systems

- Remote reboot with state-based security supports remote manageability
- Remote power-off with state-based security enables off-hours power savings
- Flexible packet structure for SNMP and remote management and control support
- Supports standards based network alerting
- Auto-acknowledge enables UDP reliability for receive and alert packets
- Power-on Reset allows cost savings

Alert on LAN II Software Architecture

The software architecture for Alert on LAN* II is pictured below. The Managed Client is a computer with Alert on LAN hardware; the Management Console is a computer that interprets Alert on LAN events on a remote system.

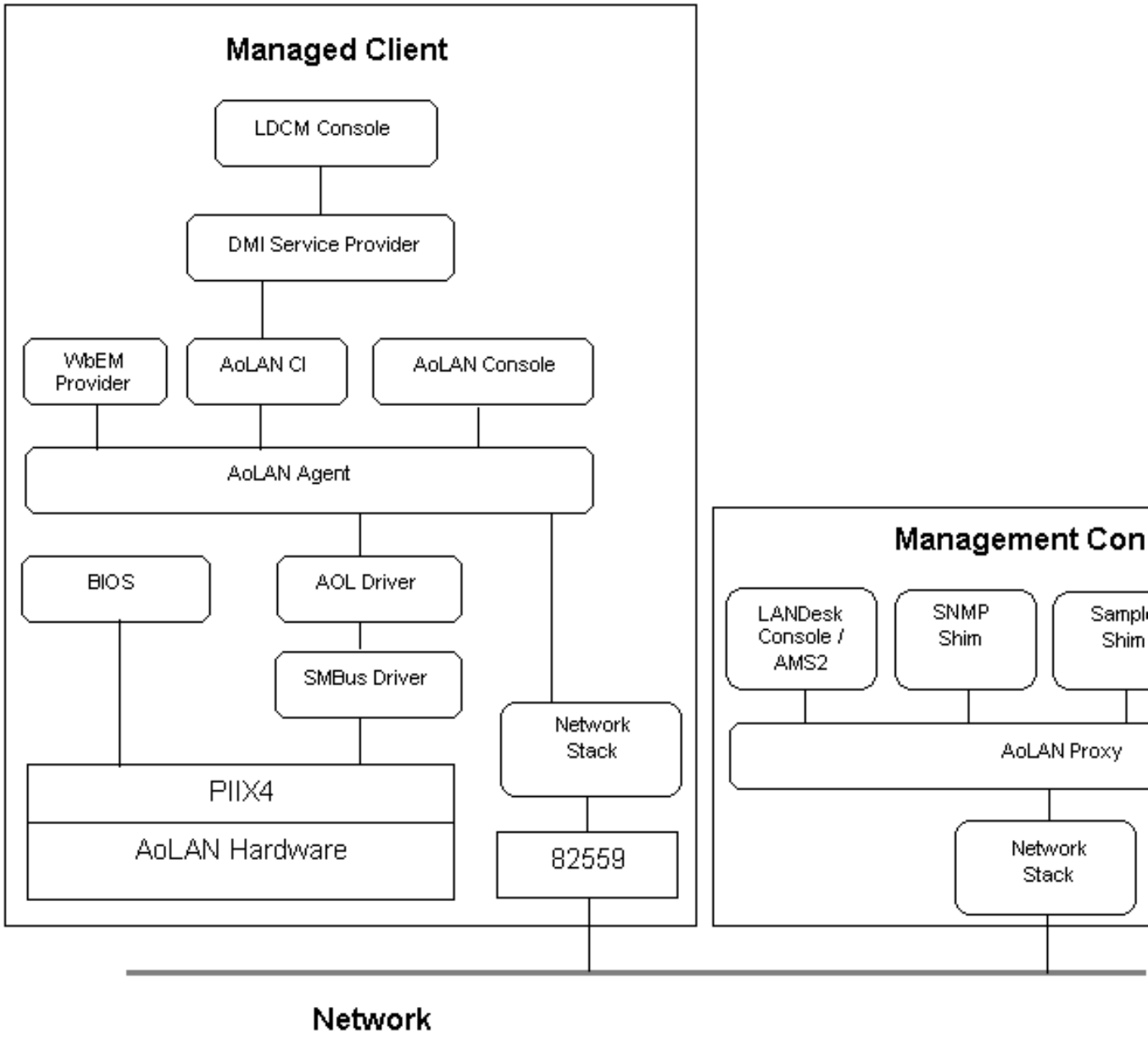


Figure - Alert on LAN II software architecture

BIOS Implementations

Note This information on Alert on LAN* II BIOS information is taken from the Alert on LAN II SDK.

The Alert on LAN II SDK is based on the generic Alert on LAN BIOS model described in the Alert on LAN II Software Developer's Manual. Each sample implementation includes test procedures written in assembler that exercise the target Alert on LAN platform hardware and provide reference code to simulate BIOS functionality.

Alert on LAN II ICH BIOS Sample

Reference source code is provided in the bios\ICH directory to help create an ICH Alert on LAN-enabled BIOS. For a implementation details of the source code, please refer to the Alert on LAN Software Developer's Manual.

The sample source code exists in the bios\samples\ich directory and will generate the executable bios\samples\ich\ichbios.exe.

Alert on LAN II Cape Lookout BIOS Sample

Reference source code is provided in the BIOS\CL directory to help create a Cape Lookout-enabled BIOS. For a implementation details of the source code, see the Alert on LAN Software Developer's Manual.

The sample source code exists in the bios\samples\cl directory and will generate the executable bios\samples\cl\clbios.exe.

Alert on LAN II Combined ICH/Cape Lookout BIOS Sample

Reference source code is provided in the bios\samples\combined directory to help create a combined ICH/Cape Lookout-enabled BIOS. For a implementation details of the source code, see the Alert on LAN Software Developer's Manual.

The sample source code exists in the bios\samples\combined directory and will generate the executable bios\samples\combined\aolbios.exe.

Caching Alert on LAN Data

The Alert on LAN* Client Instrumentation improves its performance by caching Alert on LAN EEPROM values. The caching algorithm prevents, for example, writing data to the EEPROM that is identical to existing data, or reading data that was recently read. These parameters can be modified by creating the DWORD registry values "Cache Enabled" and "Cache Lifetime" under the registry key

`HKEY_LOCAL_MACHINE\Software\Intel\cimgr\Instrumentation\Intel.IOAol.2.1`

- "Cache Enabled": The internal default is 1. Setting to 0 disables all caching and "Cache Lifetime" values will be ignored.
- "Cache Lifetime": The internal default is 5. Higher numbers may slightly improve performance. Lower numbers may improve accuracy.

Upgrading to the Alert on LAN SNMP Alerter

Before you upgrade the Alert Proxy Server (installed with Client Manager) to the Alert on LAN* SNMP Alerter, you must:

- Stop all running programs that are using the Alert Proxy Server (including Client Manager and related services)..
- Remove any existing versions of the Alert Proxy Server on the computer.

To upgrade the Alert Proxy Server to the Alert on LAN SNMP Alerter, follow the steps below.

Client Manager Running on Windows NT

1. Open the Services control panel, and stop the Client Manager Notification Monitor service.
2. Locate the executable apuninst.exe (usually found in the C:\Program Files\Intel\Alert on LAN\winnt\agent directory).
3. From the command line, or the Start->Run menu, execute:

   ```
   [path to apuninst]\apuninst.exe "ldcm"
   ```
4. Run the SETUP.EXE that shipped with the Alert on LAN SNMP.
5. Reboot the computer. The Client Manager Notification Monitor automatically restarts.

Client Manager Running on Windows 98, Windows ME, or Windows 2000/XP

1. Find DMIStart.exe and rename it to DMIStart.bak.
2. Reboot the computer.

   Upon the next reboot, Windows will display a dialog indicating it could not start DMIStart. You can safely ignore this message.
3. Locate the executable APUNINST.EXE (usually in the C:\Program Files\Intel\LDCM directory).

4. From the command line, or the Start | Run menu, execute:

   ```
   [path to apuninst]apuninst.exe "ldcm"
   ```

5. Locate and run the SETUP.EXE that shipped with the Alert on LAN SNMP Alerter.
6. When you are prompted to reboot the computer at this point, click No.
7. Rename DMIStart.bak back to DMIStart.exe.
8. Reboot the computer.

About BSA Packages

Client Manager 6.3 enables you to set up Intel Bootstrap Agent (BSA) packages on an http, https, or outside proxy Web server. A Client Manager administrator can then remotely download and install these packages onto client computers. By default, all clients have BSA installed as part of Client Manager 6.3.

These packages are designed to typically contain OEM-specific BIOS or driver updates. They must be digitally signed by Intel before being available to customers. A Client Manager administrator can't select an unsigned package for downloading to clients.

The general process for setting up a BSA package is as follows:

- Create a BIOS or driver update package that includes .EXE files, command-line options, system requirements, and environmental variables necessary to properly install the package on a BSA-enabled client.
- Send these package files to Intel to be turned into a manifest (.MFS) file that is digitally signed. This layer of security is necessary so that the manifest can't be altered.
- Post the .MFS file to an http, https, or outside proxy Web server for your customers to download whenever they need BIOS or driver updates.

For additional information regarding BSA package creation, and the availability of this service with the product version that you have purchased, please visit the Intel desktop board LDCM web site or contact your Intel Customer Representative.