A large, abstract graphic on the left side of the page, consisting of several overlapping, curved, semi-transparent shapes in shades of gray, creating a sense of depth and movement.

# SUN JAVA™ SYSTEM CALENDAR SERVER 6.3

White Paper  
August 2009

# Table of Contents

<b>Executive Summary</b> .....	<b>1</b>
<b>Java System Calendar Server Product Overview</b> .....	<b>3</b>
What is the Java System Calendar Server? .....	3
Key Functionality .....	3
Calendar Server Interactions .....	5
Java System Calendar Server Architectural Overview .....	7
Java System Calendar Server Components .....	8
Distributed Horizontally Scalable Architecture .....	9
Java System Calendar Server APIs .....	10
<b>Advantages for Enterprises and Service Providers</b> .....	<b>12</b>
Multiclient Access .....	13
Security at Multiple Levels .....	20
High Performance, High Availability .....	21
Event Notification Service (ENS) .....	25
Hosted Domains .....	27
Calendar Server Administration .....	28
Migration .....	30
<b>Conclusion</b> .....	<b>32</b>
High-Performance Server .....	32
Based on Standards and Proven Technology .....	32
Flexibility and Extensibility .....	32
Feature-rich Server and Client .....	33
<b>Appendix A</b> .....	<b>34</b>
Standards and References .....	34
<b>References</b> .....	<b>35</b>
<b>Abbreviation/Acronym List</b> .....	<b>36</b>

## Executive Summary

Organizations today require a dependable, scalable, cost-effective calendaring solution that integrates with other popular calendaring solutions as well other communications components, such as messaging and addressbook. These organizations need a secure solution that is adaptable to their individual environments and highly available to ensure that they always have access to information about important appointments.

The Sun Java™ System Calendar Server satisfies these requirements. It is a high-performance, Internet standards-based calendar server designed with the scalability to meet the needs of customers ranging from medium- and large-sized enterprises to even the largest Internet, telecommunications, and enterprise service provider. Through a rich integrated AJAX Web interface or connectors to other calendar clients (including Microsoft Outlook), the Java System Calendar Server provides group scheduling and personal calendaring to consumers at home or at work while also enabling them to share calendar information with others over the Internet. The user interface (UI) can be customized to include Web links for e-commerce, banner ads, logo or brand of the calendar server customer, and more. This paper explores the product's architecture, design, deployment features, and benefits.

The Java System Calendar Server provides one of the industry's most open, interoperable, and high-performance time and resource management solutions. Through its scalability, performance, and reliability, it can provide customers with the features they require at a lower total cost of ownership than alternative solutions.

Native support for iCalendar standards allows users to schedule events in a format that is easily shared across the Internet. Sun continues to lead in defining and implementing industry standards across its entire communications product line. The Java System Calendar Server employs standards and protocols such as:

- Internet Calendaring (iCalendar)
- iCalendar Transport-Independent Interoperability Protocol (iTIP)
- Calendar Message-based Interoperability Protocol (iMIP)
- eXtensible Markup Language (XML)
- Lightweight Directory Access Protocol (LDAP)
- HyperText Transport Protocol (HTTP)

The Java System Calendar Server is a component of the Sun Java™ Communications Suite, an open and integrated infrastructure software system that delivers industry-leading e-mail, calendaring, and real-time collaboration functionality for service providers and large organizations worldwide. The Java System Calendar Server architecture is flexible, extensible, and scalable both vertically (by increasing the number of CPUs per system) and horizontally (by introducing additional servers into the network). Simply put, the Java System Calendar Server can be thought of as a system of servers that may be configured to fit a variety of needs. It can remain in isolation as a stand-alone calendar server or can be configured with many instances, having the various services duplicated or split between them. The Java System Calendar Server makes use of plug-ins to obtain external services. It also supports both LDAP- and identity-based deployments and integrates with other components of the Java Enterprise System, such as the Java System Access Manager and Java System Web Server, to provide additional functionality.

## Chapter 1

# Java System Calendar Server Product Overview

## What is the Java System Calendar Server?

The Java System Calendar Server is a high-performance, standards-based solution for time and resource management. It is designed with the scalability to meet the needs of Internet service providers (ISPs), telecommunications providers, and enterprise service providers as well as large enterprises. It provides group scheduling and personal calendaring to consumers at home or at work, and lets them share calendar information with others over the Internet. The Java System Calendar Server can provide customers with lower total cost of ownership because of its scalability, performance, and reliability.

## Key Functionality

The Java System Calendar Server provides the following high-level functionality:

- A modular, pluggable architecture where components and services can be run on a single system or spread over multiple systems. The architecture is scalable vertically by CPU and horizontally by system, and provides server-side APIs and client protocols for third-party client integration.
- Web-based group scheduling as well as personal calendaring. Group scheduling includes free-busy lookup; next available free slot; attendees, invitations, and responses; and conflict handling in cases where an attendee's time may be double-booked. The Java System Calendar Server can also be used to schedule resources such as conference rooms and audio-visual equipment.
- Connectors to additional calendar clients including Microsoft Outlook, Thunderbird/Lightning and Evolution, thereby enabling users of these products to perform group scheduling with Java System Calendar Server users. This is extremely useful for Outlook enterprise users in search of a lower cost calendaring solution that integrates into their existing environments.
- Multiple calendars per user, multiple owners per calendar, the layering of calendars onto other calendars to create composite calendar views, calendar access control, delegated calendar ownership, and daily, weekly, monthly, yearly, or comparison views. Each calendar has a primary owner, and may have other owners who can act on behalf of the primary owner. These owners can invite others to meetings or reply to invitations on behalf of the primary owner. A user can create different calendars, much like mail users can create different folders. Calendars can be secured for availability, schedule, read, modify, or delete access. As a result, calendars can be made publicly accessible to individuals, groups, or everyone, or kept private solely for the owners. Privacy can also be designated at the event level. Individual events can be assigned public, private, or confidential access.

- Includes Sun™ Convergence, a highly configurable, extensible AJAX Web client that combines e-mail, calendaring, address book and instant messaging into a rich integrated Web experience. Single sign-on and tight integration enable users to utilize calendaring functionality from within mail and address book applications.
- An integrated corporate address book that is used to search for potential attendees. The user can enter any part of an attendee's name and the address book searches through LDAP for appropriate matches. The user can then select the correct name and invite the user or group to the meeting.
- Flexible event notification and subscription service that enables processes to register and subscribe to events of interest and perform appropriate actions. The notification payload can be binary, iCalendar, or XML data. The server supports an alarm queue that can send event reminders and task reminders to e-mail addresses. An API enables developers to extend this notification capability to any type of notification, including Short Message Service (SMS) or pager notifications. There is also a server preference to enable or disable this feature.
- Event feeds can be imported and offered as a service to consumers. These event feeds can be produced by individuals or content providers. Consumers may subscribe to event feeds of interest, such as horoscopes or symphony calendars, and even purchase tickets.
- Supports the use of hosted domains. In a hosted-domain installation, each domain shares the same instance of the Java System Calendar Server, allowing a number of domains to exist on a single server.
- Command line administration enables administrators to create simple scripts to perform routine tasks such as calendar or database backups; enabling or disabling resources; starting and stopping the server; checking statistics (number of reads, writes, connections, and so on); creating, modifying, and deleting calendars; creating, modifying, and deleting users; and more. Administrators may also choose to use the Delegated Administrator graphical user interface to provision calendar users and resources.
- Standards-based design to enable interoperability with other standards-based calendar systems. This design provides iMIP interoperability with Microsoft Outlook users.
- A customizable, front-end UI based on Asynchronous JavaScript™ and XML (AJAX) technology. This customization enables customers to specify UI Themes, to incorporate special branding, or to integrate third-party functionality (mashups).
- Client development through the use of WCAP. This enables users to develop client applications that can communicate directly with the Java System Calendar Server; an example is the Lightning extension to the popular Mozilla Thunderbird messaging client.
- Employs an LDAP directory for user provisioning and storing user preferences. The Java System Calendar Server also supports the use of static, dynamic, or nested LDAP groups within event invitations. This enables users to invite groups to events in addition to individuals. Events appear in the group calendar as well as the individual calendars.

- Integrates with the Java System Access Manager to provide identity-enabled calendaring, which incorporates single sign-on and provisioning. Developers can choose to use the Java System Access Manager to provide single sign-on or the Java System Calendar Server APIs to share authentication across other applications.
- Natively supports SSL for secure communications between the client and server.
- Supports high availability through the use of Sun Cluster technology.
- Supports synchronization of events, tasks, and contacts between the Java Communications Suite and Palm OS-based devices, Pocket PC devices, Windows Mobile 5.0 devices and Outlook.
- A set of migration tools are available for migrating previous versions of the Java System Calendar Server to Java System Calendar Server 6.3.
- Wireless access via the Java Mobile Communications software and third-party solutions.

## Calendar Server Interactions

In order for online calendar information to be useful, it must be readily available. Though users may have access to a Web browser much of the time, there will be times when the information must be available on other devices such as a cell phone, pager, personal digital assistant (PDA), desktop client, and so on. Figure 1 shows the various devices with which the Java System Calendar Server may interact.

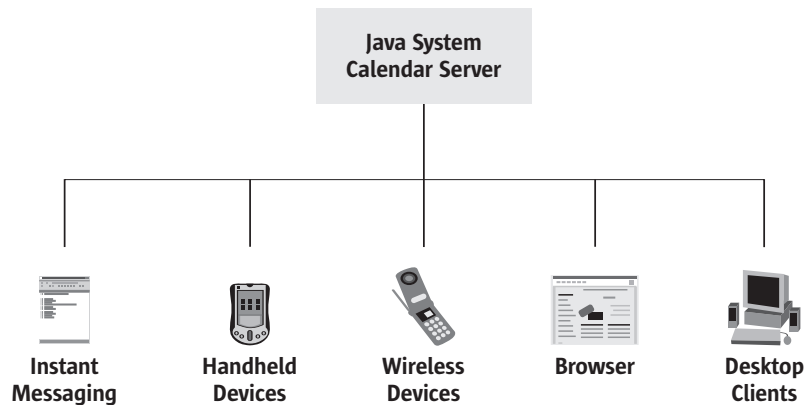
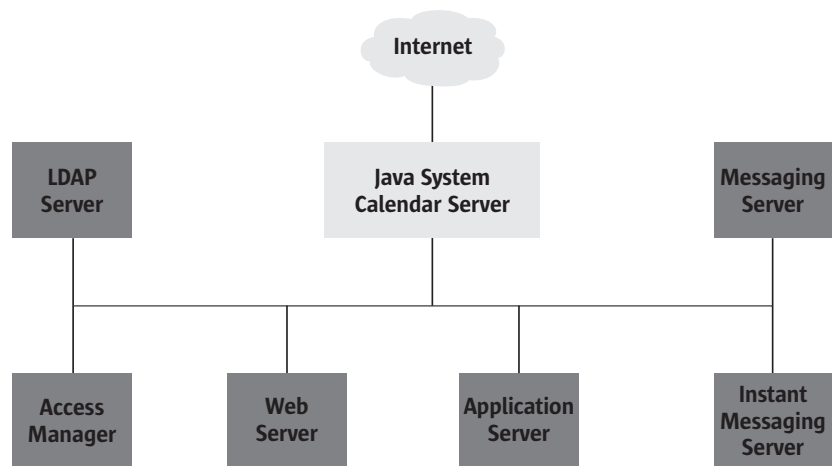


Figure 1. Java System Calendar Server interactions

A calendar server must provide high performance and be highly scalable. It must be highly available so that customers' data is available when they need it. Ideally, it should integrate easily into an infrastructure of other services such as mail and directory.

An ISP or enterprise may wish to add value to services they already offer by adding calendaring features. A calendar server should facilitate development of custom calendar-based applications. The data supplied by the calendar server should be in a standard format.

Figure 2 shows an example of how the Java System Calendar Server can be incorporated with other servers that make up both ISP and enterprise infrastructures.



*Figure 2. Java System Calendar Server relationship with other servers*

The Java System Calendar Server requires external services for authentication and user preference storage. By default, these services are provided by LDAP. However, the Java System Calendar Server can be used in an infrastructure where directory services and user preferences are provided by other mechanisms. There is a plug-in API for replacing many components in the Java System Calendar Server, along with source code for sample plug-ins.

The Java System Calendar Server is built from the ground up on standards. Its data store is based on iCalendar technology, and it supports iMIP Publish, Request, Reply, and Cancel messages for events and tasks. Native support for iCalendar standards enables users to schedule events in a format that is easily shared across the Internet.

The Java System Calendar Server supports the use of a rich integrated AJAX Web client — Sun Convergence — that combines calendaring, address book, e-mail, presence, and instant messaging into an integrated Web experience. Because Sun Convergence is AJAX technology-based, it can be customized for a particular enterprise or ISP. The Java System Calendar Server also supports a more traditional integrated Web client — Java System Communications Express — that integrates calendaring, personal address book, and messaging functionality. Additionally, the Java System Calendar Server provides connectors to other popular calendar clients, thereby enabling a variety of clients and devices to communicate with the Java System Calendar Server.

The Java System Calendar Server provides an open protocol — the Web Calendar Access Protocol (WCAP) — for developing clients that communicate with the Java System Calendar Server. WCAP is extremely useful in cases where calendar information must be provided to nonbrowser interfaces such as phones, PDAs, or other devices.

Built with flexibility in mind, the Java System Calendar Server is capable of running in LDAP-based or identity-based environments. The LDAP-based environment uses the LDAP directory for authentication and calendar server administration tools for provisioning, whereas the identity-based deployment uses the Java System Access Manager for authentication and the Communications Services Delegated Administrator for provisioning.

## Java System Calendar Server Architectural Overview

Figure 3 illustrates the Java System Calendar Server internal architecture.

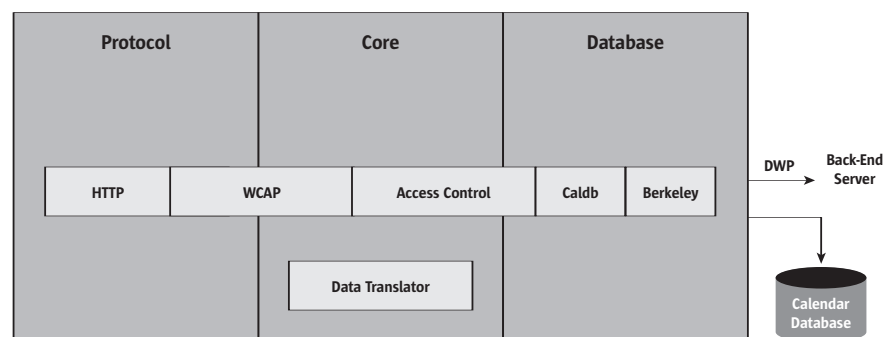


Figure 3. Java System Calendar Server architecture

## Java System Calendar Server Components

The Java System Calendar Server is built, in part, on proven components from the Java System Messaging Server and Java System Directory Server. The design is highly modular. The Java System Calendar Server is implemented by way of a collection of shared libraries that fall into three main categories or subsystems: Protocol, Core, and Database. These shared libraries are bound in various combinations to produce the executable daemons `cshttpd`, `csdwpd`, `csadmin`, `csstored` and `csnotifyd`. In addition, the daemon `enpd` provides an event notification service that is shipped with the Java System Calendar Server.

Commands or requests enter the server through the Protocol subsystem and are passed to the Core subsystem for processing. Database accesses go through the Database subsystem. The role of each of these subsystems is as follows.

### Protocol Subsystem

Commands or requests enter through the HTTP protocol layer. This is a minimal HTTP server implementation, streamlined to support calendar requests. Clients use WCAP to submit requests. WCAP is based on HTTP and is an open protocol that can perform all server commands (except for certain administrative commands). It can be used by clients that need raw, unformatted calendar information.

### Core Subsystem

The core subsystem consists of several other subsystems — access control, data translation and calendar database (Caldb) subsystem — and the Calendar Server API (CSAPI) plug-ins. The core subsystem processes calendar requests and also handles user authentication.

### Database Subsystem

The database subsystem is responsible for storing data to and retrieving data from the database. This subsystem sends the data in a low-level format to the core subsystem for processing. The database subsystem uses the Berkeley DB from Oracle Corporation (formerly Sleepycat Software.) Even though the database API is currently not public, the database is based on the iCalendar standard. The schema used for events and tasks is a superset of the iCalendar standard. An important feature of the database is that it can be backed up without shutting down the server. Event content can be loaded by administrative tools in the back end at the same time users access the server with the Sun Convergence or Java System Communications Express Web clients. The calendar database supports events, tasks (iCalendar to-do's), alarms, and calendar attachments.

## Distributed Horizontally Scalable Architecture

The relationship of the calendar server to other components is illustrated in Figure 4.

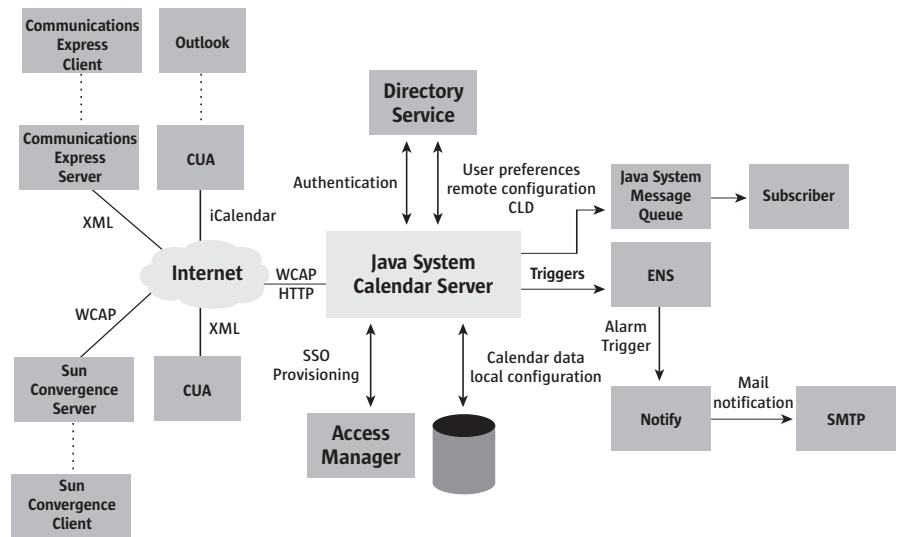


Figure 4. Minimal Java System Calendar Server system

In this diagram, the server is depicted as a single system with connections to a directory service, database, access manager, the event notification service (ENS), and the Internet. Calendar user agents (CUAs) are also shown, illustrating the server's ability to communicate with other devices or agents using WCAP in either the iCalendar or XML format. One such CUA is the client connector, which enables desktop clients such as Microsoft Outlook or Evolution to communicate with the Java System Calendar Server. The Java System Communications Express server is another example of a CUA. Although this diagram depicts the calendar server as a single system, portions of the server can be distributed over several systems.

For example, Web server processes can be run on one machine (the front-end server) and the database subsystem can be run on a separate machine (the back-end server). Furthermore, there can be multiple front- and back-end servers, creating the effect of distributing calendars over multiple calendar servers. For example, a user may wish to distribute all the calendar users over four machines. Machine 1 may serve all calendars in the U.S., the second machine may serve all calendars in Europe, and so forth.

The Java System Calendar Server is scalable both vertically and horizontally. On a single machine it can run in multiple processors, or it can be split up to run on multiple machines. Figure 5 depicts three Java System Calendar Server HTTP front-end services using three other Java System Calendar Server database services. All six of these instances can run on separate machines. The front-end servers communicate with back-end database servers using the Database Wire Protocol (DWP), an internal protocol that provides networking capability to the Berkeley DB.

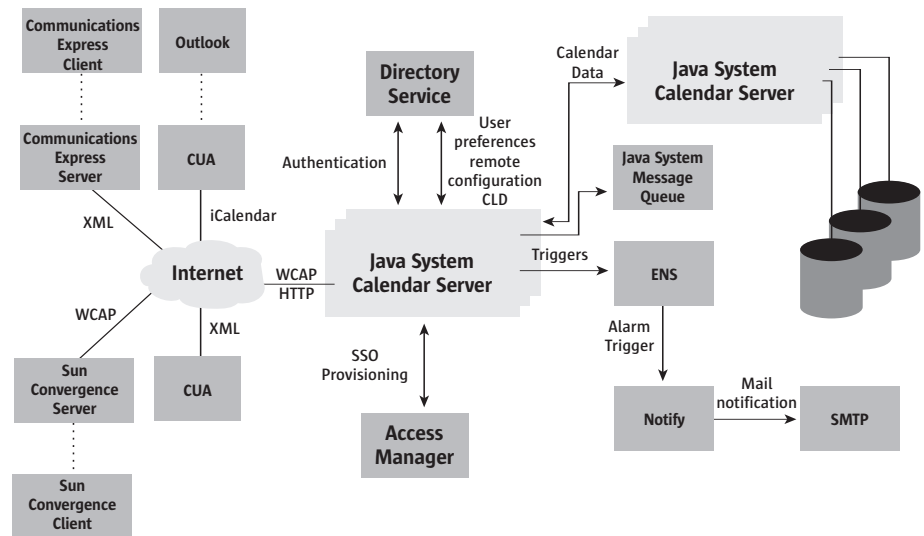


Figure 5. Java System Calendar Server scalability

## Java System Calendar Server APIs

In addition to providing a modular, highly scalable architecture, the Java System Calendar Server provides the following application programming interfaces (APIs) and software development kits (SDKs) to extend the functionality of the product.

### Calendar Server API (CSAPI)

The CSAPI enables developers to modify or enhance the feature set of the Java System Calendar Server. For example, the user can modify or override the default access control mechanism, login authentication mechanism, or even the default calendar lookup mechanism.

### Event Notification Service (ENS) and Java System Message Queue APIs

The ENS API enables developers to create event publishers and event subscribers, and to dispatch these functions within the ENS offered by the Java System Calendar Server. ENS is a generic, publish-and-subscribe service that acts as a central point of collection for certain types of events that are of interest. The Java System Calendar Server may be configured to watch for certain types of events. The ENS API enables a user to extend the list of interesting events beyond what the Java System Calendar Server currently provides. In addition to supporting ENS for event notification, Java System Calendar Server also supports the Sun Java System Message Queue. The Java System Message Queue provides additional APIs for handling event notifications.

### **Proxy Authentication SDK (authSDK)**

The authSDK enables a calendar administrator to log onto the Java System Calendar Server on behalf of a calendar user, which is useful for integrating additional servers or services with the Calendar Server. For example, a portal system may authenticate the user, and the authSDK allows the user access to the Java System Calendar Server without reauthentication.

### **Web Calendar Access Protocol (WCAP)**

The WCAP mechanism enables arbitrary clients to retrieve calendar data from the server. WCAP 4.0 may be used to query the Java System Calendar Server for data or modify data on it. WCAP 4.0 returns data in one of two data formats: iCalendar or XML/iCalendar. This is extremely useful, and enables a wide range of devices to communicate with the Java System Calendar Server.

## Chapter 2

# Advantages for Enterprises and Service Providers

In order for a calendaring solution to meet the needs of today's enterprises and service providers, it must use open protocols, enabling accessibility to real-time calendar data from the Web and popular desktop clients. It must support online and offline modes, and provide synchronization capabilities to maintain the consistency of the calendar data. The interface should be customizable to enable unique branding. Furthermore, the calendaring solution must provide secure access and a means to ensure the privacy of the data. It must be scalable and adaptable to individual customer environments and highly available to ensure that the customer always has access to information about important appointments.

The Java System Calendar Server provides a wealth of features and benefits, including:

- Web-based group scheduling, free-busy lookup, and corporate directory lookup
- Integrated personal address book and e-mail functionality
- Web-based resource scheduling for conference rooms, projectors, and other resources
- Global and domain-level customization (color, login, user interface, logo, branding, and so on)
- Connectors to additional communications clients including Microsoft Outlook, Mozilla Thunderbird and Evolution, enabling these clients to perform scheduling on the Java System Calendar Server
- APIs to extend calendar server interoperability with other clients
- Support for multiple calendars, such as work, family, friends, and more
- Support for public and private calendars as well as public, private, and confidential individual events
- Support for layered calendar views, enabling users to combine two or more calendars into a single view for improved communication and productivity
- Sun Convergence UI supports drag and drop of calendar events, context sensitive menus, event creation using natural grammar, and interactive hover pop-ups
- Flexible event notification system that supports the Java System Message Queue or the Event Notification Server. Automatic e-mail notification of appointments, invitations, and reminders sent to selected recipients; integrates with Java System Instant Messaging to provide automatic pop-up reminders
- Support for multiple owners of each calendar to facilitate communication and productivity with project teams and community groups
- Ability to delegate calendar ownership to others who may act on behalf of the primary owner
- Daily, weekly, monthly, yearly, and comparison views
- Event, invitation, and task views

- Support for hundreds of thousands of users through a scalable, networked, server-to-server, client-server architecture
- Support for Secure Sockets Layer (SSL) encryption, LDAP authentication, authentication plug-ins, and identity-enabled single sign-on (SSO) through the Java System Access Manager
- Text search for locating events based on key words
- Event and task categories for identifying the type of event or task
- Support for standards-based event and calendar data feeds published in XML or iCalendar formats, which improves communication and can offer new revenue opportunities through commerce links and banner ads
- Native LDAP support, with an application programming interface (API) for other directory services
- Support for hosted domains including command line and GUI tools to manage these domains
- Simplified system management, online backup and restore, and entire database backup and restore (including Hot Backup)
- Wireless access via the Java Mobile Communications product
- Synchronization of events, tasks, and contacts with Microsoft Outlook, Palm handheld devices, Pocket PC devices, and Windows Mobile 5.0 handheld devices via Java System Communications Sync software
- Support for attachments in events and tasks
- Support for LDAP groups including static, nested, and dynamic groups
- Lower total cost of ownership (TCO)

## Multiclient Access

Service providers and enterprises today require accessibility to calendar data from a variety of clients, devices, and applications. The more accessible this data becomes, the easier it is for users to collaborate effectively. The Java System Calendar Server supports the use of a wide assortment of clients, including a rich AJAX technology-based Web user interface, a native integrated Web user interface, connectors to Microsoft Outlook clients, Mozilla Thunderbird clients and Evolution clients, as well as synchronization and wireless access.

## Sun Convergence

The Java System Calendar Server supports an integrated AJAX Web 2.0 Client also known as Sun Convergence. This client is a component of the Java Communications Suite and integrates personal address book, e-mail, and instant messaging functionality with advanced calendar functionality. It is a single client that presents a unified user interface (UI) to the back-end calendar, messaging, and instant messaging servers. It also

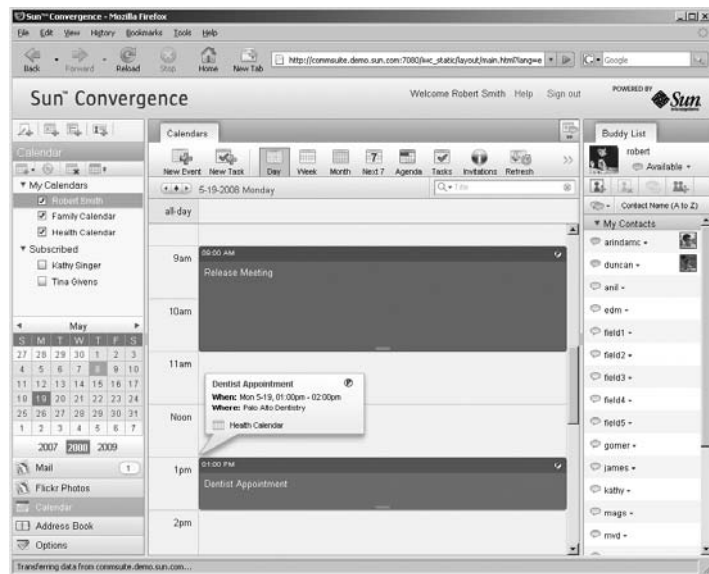


Figure 6. Sun Convergence interface

provides access to a common address book that is shared by the mail, calendar, and instant messaging functions. Sun Convergence is an alternative to the more traditional Java System Communications Express client. Although support for Java System Communications Express is still available in this release of Java System Calendar Server, support for Communications Express will be terminated in a future release.

Sun Convergence uses Asynchronous JavaScript and XML (AJAX) for its mail, calendar, address book, instant messaging and global options interfaces. It is deployed as a Web application using the Java System Application Server. Consequently, it can be deployed on a machine that is physically separate from the machine running the Java System Calendar Server HTTP front-end `cshttpd` services, or it may be deployed on the same machine running `cshttpd`.

Sun Convergence supports calendar events, tasks, and reminders; displays calendars in standard views such as day, week, and month; and also supports object oriented views such as the Agenda, Invitations, and Tasks views that enable the user to view a specific type of calendar data.

The Sun Convergence interface enables users to create and manage multiple calendars. For example, a user can have one calendar for work, a second calendar for a group of coworkers, a third calendar for company holidays, and a fourth calendar for personal activities. One of the most compelling features of Sun Convergence is its support for combining several calendars and displaying them in a single composite view. A user can define any number of calendars within a single view. For example, a user may display a view consisting of a Work calendar, Child's School calendar, Sports calendar, and Family Event calendar. The user may also display a virtual team calendar, in which

the user combines a personal calendar along with those coworkers with whom he or she works most closely. This allows the user to quickly see what everyone else is doing as well as when they can all meet.

In fact, one of the primary features of Sun Convergence is its support for group scheduling. This encompasses a wide range of activities, from inviting members to a meeting and replying to invitations to tracking attendee acknowledgments and providing a free-busy lookup (which enables the user to quickly see when invitees are available for a meeting).

Convergence also provides an integrated address book for inviting individuals to meetings. One can invite users from a personal address book or a corporate directory. Generally, an attendee is invited to a meeting by entering a userID (it is actually the calendar ID (calID) that is invited, but the default calendar ID is the userID unless the administrator changes it) and then clicking the “Add to Invite List” button. However, through the address book integration, the user needs to enter only a portion of the attendee’s name and auto completion is performed. Convergence presents a list of all name matches. The user can select the appropriate names from the list and invite them to the meeting. The address book can also be used to schedule resources such as conference rooms.

Convergence supports configurable access control, enabling users to create public calendars that are readable and writable by others as well as private calendars that are accessible only to their owners. Users can define a separate set of access permissions for each calendar that they own. The owners can also specify a list of users and define a set of access permissions for each. Allowable access permissions include:

- Availability access, which enables a user to view the free/busy availability of the calendar
- Invite access, which enables a user to schedule events into the calendar
- Read access, which enables a user to subscribe to the calendar and view events in it
- Read + Write access, which enables a user to view and modify events in the calendar
- Owner access, which enables a user to view, modify, or delete events in the calendar

Access control for calendars is supported at the individual calendar level as well as the global and system administrator levels. The type of access granted can be differentiated across individuals. For example, a user may wish to grant certain users read-only access while others receive availability and invite-only access. Availability and invite-only access allow a user to schedule an item into the calendar, but the user cannot read appointments in the calendar. As a result, User1 knows when User2 is available for a meeting (and can schedule a new meeting into User2’s calendar), but cannot see individual items on User2’s calendar. In contrast, when a calendar is configured for public read, no access check is made for read requests. This means that a URL to a public calendar can be put into a static Web page, an e-mail message, or a newsgroup message; when the user clicks it, the calendar is displayed without requiring the user to log into the server.

Convergence also provides access control at the individual event level through the use of public, private, and confidential events. Individual event access control enables users to create private events within an otherwise public calendar. This is especially useful for users who would like to keep their default calendar public, but want to control the privacy of certain events within that calendar. Events can be designated public, private, or time and date only.

Designed with worldwide calendar sharing in mind, Convergence supports multiple time zones. The time scale for calendar views can be in any time zone. In fact, multiple time zones can be shown, and a specific time zone can be applied to a calendar.

Recurring events and tasks are supported with a full range of recurrence options, including repeat for a specific number of instances and repeat until a specific date. Events and tasks can also have reminders that may be e-mailed to one or more addresses. Integration with Java System Instant Messaging provides an automatic pop-up notification to remind users of upcoming appointments. Printer-friendly views are also supported. Convergence can import and export calendar information in iCalendar format.

Convergence not only provides superior calendar functionality, but it also provides an integrated solution that is more functionally rich than the individual calendar or messenger user interfaces. Consequently, the Convergence user interface is ideal for those individuals who want a fully integrated rich Web client — not just a standalone calendar client. However, it is not necessary to install the Java System Messaging Server or Java System Instant Messaging to take advantage of Convergence. Convergence can still be used as an alternative interface to the traditional Java System Communications Express even if only the Java System Calendar Server is installed. The Personal Address Book integration would be present with or without the Java System Messaging Server or Java System Instant Messaging. Convergence introduces calendar-specific, value-added features such as color-coded calendars and the ability to use natural grammar to schedule events. Convergence enables users to:

- Invite a contact to an event, or e-mail a contact from within a personal address book
- View an invitee's instant messaging presence while creating a calendar event
- Change a calendar event's date or time using Drag and Drop
- Search for users in a personal address book or remote address book such as the corporate address book from within calendar
- Quickly create an event or a contact using instant event and contact shortcuts from within the calendar or address book
- Generate a list of all outstanding events, invitations, or tasks
- View the email sender or recipient's calendar availability from within the messaging application
- Easily traverse between the calendar view, the mail view, and the address book view as seamlessly as clicking links on a Web page

The Sun Convergence UI is highly customizable. Because it is based on AJAX, the calendar client can be customized to suit the needs of the deployment or the domain. Customization examples include customizing the skin or theme, the login screen, the icons used in the calendar component, the text within Sun Convergence, the application banner and application bar, the branding, the toolbars, the calendar views, the calendar pop-ups, and so on. Customization also includes the integration of 3rd party applications into the user interface not only to enable a new service but to integrate with the existing services as well. To the end user, the application appears in the UI as another component, just like mail or calendar.

### Java System Communications Express

The calendar server also supports a more traditional Web client, Java System Communications Express. This Web client is JSP-based and supports calendar events, tasks, and reminders; displays calendars in standard views such as day, week, month, and year; and can also display multiple calendars side by side in a comparison view.

A key feature of Java System Communications Express is its support for the use of the HyperText Markup Language (HTML) in the event title and description. This enables users to incorporate rich HTML content in their events, such as links to other Web sites, static or moving images, and more. A possible use of this feature is the creation of a Sporting Events calendar with links to Web sites to purchase tickets. Another use is providing links to Web sites where meeting agendas and other documentation are located.

The Java System Communications Express interface provides a similar level of calendar functionality as provided by Sun Convergence. One major difference is that Convergence provides a rich Web experience that approximates a fat client experience and includes drag and drop, interactive displays, and context sensitive menus, whereas Communications Express presents a more traditional thin client user interface.

Java System Communications Express uses Java Platform, Enterprise Edition (Java EE) technology such as JavaServer Pages™ (JSP), the eXtensible Markup Language (XML), and the Java System Application Framework (JATO) for its calendar, address book, and global options interfaces. It is deployed as a Web application using either the Java System Web Server or the Java System Application Server. Consequently, it can be deployed on a machine that is physically separate from the machine running the Java System Calendar Server HTTP front-end `cshttpd` services, or it may be deployed on the same machine running `cshttpd`. Java System Communications Express communicates to `cshttpd` over WCAP.

The Java System Communications Express UI is highly customizable. The calendar component of Java System Communications Express uses .jsp pages as the presentation layer. These JSP pages can be customized to suit the requirements of the client. Customization examples include customizing the skin, the login screen, the icons used in the calendar component, the text within Java System Communications Express, the application banner and application bar, the branding, the toolbars, the calendar views, the calendar pop-ups, and so on. But even though the Communications Express UI is highly customizable, it is not the preferred user interface to customize. The preferred UI is Sun Convergence since it is a richer, more sophisticated interface.

### Outlook Connector

In addition to providing native Web clients, the Java System Calendar Server supports the use of connectors to other popular enterprise clients. One of these clients is Microsoft Outlook. The Java System Connector for Microsoft Outlook enables direct connection from Outlook to the Java System Calendar Server, Java System Messaging Server, and Java System Directory Server. Therefore, it eliminates the need for Microsoft Exchange Server for messaging and calendaring. The Java System Connector for Microsoft Outlook is a Messaging API (MAPI) provider that is installed on the Outlook desktop machine. It uses Simple Mail Transfer Protocol (SMTP) and Internet Message Access Protocol 4 (IMAP4) protocols to communicate with the Java System Messaging Server, LDAP to communicate with the Java System Directory Server, WABP (Web Address Book Protocol) to communicate with the personal address book, and WCAP to communicate with the Java System Calendar Server.

The Java System Connector for Microsoft Outlook maintains a persistent connection with the Java System Calendar Server, thereby enabling real-time access to calendar data from Outlook. Because Outlook provides a database or Personal Folder Storage (PST) file for use in offline mode, users of the Java System Connector for Microsoft Outlook can update their schedules while on the road and later synchronize their offline store with their calendar server. Additionally, the Java System Calendar Server supports a mixed environment of clients such that Java System Communications Express users can easily schedule appointments with Outlook users and vice versa.

Key features of the Java System Connector for Microsoft Outlook include:

- Access to mail, calendar, contacts, and tasks
- Support for Outlook corporate/workgroup mode
- Send and receive meeting requests and responses via e-mail
- View free/busy times of invitees
- Calendar sharing and delegation
- Shared folders (mail, calendar, contacts, and tasks)
- Outlook to Communications Express interoperability
- Outlook to Sun Convergence interoperability

- Offline access
- Automated install and configuration
- Seamless upgrade from a previous version
- Desktop data migration and conversion
- Supported platforms: Outlook 2003 and Outlook 2007
- Localized in seven languages

### Evolution Connector

Evolution is a desktop productivity tool that provides e-mail, calendaring, and personal information management for users of Linux and the Solaris™ Operating System. It is an Outlook look-alike, with the same look and feel. Evolution integrates with Microsoft Exchange 2000 Server as well as other messaging and collaboration servers. With the introduction of the Java System Connector for Evolution, Evolution desktop clients can communicate with the Java System Calendar Server, Java System Directory Server, and Java System Messaging Server to provide real-time access to messaging and calendaring.

### Java System Communications Sync

For PDA users, the Java System Calendar Server supports desktop synchronization through the Java System Communications Sync software. This software runs on a Microsoft Windows PC to synchronize calendar, tasks, and contacts data between the Java Communications Suite and another data source such as a personal information manager (PIM) or mobile device. It provides synchronization for Palm OS-based devices, Pocket PC devices, Windows Mobile 5.0 devices, and Outlook.

The Java System Communications Sync software enables users to customize settings, including field mappings, data types, filters for determining the amount of data to be synchronized, and more. It also provides an autosync feature that is event-driven for PDAs and time driven for PIMs. For users who have multiple calendars, it provides the ability to select which calendar they want to synchronize.

### Java Mobile Communications

The Java Mobile Communications software provides mobile device synchronization with Calendar, Tasks, and Contact information on the Java Communications Suite servers. The Mobile Communications software supports virtually any SyncML capable device and also provides SyncML clients for Palm and Windows Mobile devices.

## Other Clients

The Java System Calendar Server has a simple, open protocol (the Web Calendar Access Protocol (WCAP), which sits on top of HTTP) that can be used by new and custom applications. With WCAP, any application that can open a socket and speak HTTP can be a calendar-enabled application. One example of a calendar client that uses WCAP is the Lightning calendar extension to the popular Mozilla Thunderbird messaging client. Lightning is essentially an open source calendar component for the Thunderbird client that provides an integrated messaging and calendaring solution.

Finally, Java System Calendar Server integrates with several third party solutions that extend calendaring functionality to other clients such as SyncML devices, Blackberry devices, Windows Mobile devices, Palm devices, and other clients. For a list of integrated solutions, visit the partner page at: [http://www.sun.com/software/products/communications/partner\\_library/index.xml](http://www.sun.com/software/products/communications/partner_library/index.xml).

## Security at Multiple Levels

Security plays a key role in the day-to-day operations of today's enterprise. Breaches in security cannot only compromise trade secrets, but may also result in downtime, data corruption, and increased operation costs. The Java System Calendar Server provides a number of security levels to protect users against eavesdropping, unsanctioned usage, or external attack. The basic level of security is through authentication. The Java System Calendar Server uses LDAP authentication by default, but also supports the use of an authentication plug-in for cases where an alternate means of authentication is desired. Furthermore, integration with the Java System Access Manager enables the Java System Calendar Server to take advantage of its single sign-on capability.

Security involves not only ensuring the integrity of users; it also means ensuring the confidentiality of data. To this end, the Java System Calendar Server supports the use of SSL encryption for login only or for both login and data. In other words, only the login may be encrypted or the entire session including the login may be encrypted, from the Web client to the server. Integration with the Java System Portal Server Secure Remote Access also provides SSL encryption, but through a proxy gateway. In addition, integration with the portal gateway provides a URL rewriting capability to further insulate the Java System Calendar Server from external entities. The Java System Calendar Server can be deployed with the portal gateway such that there is no direct connection to the Java System Calendar Server without going through the gateway. In this case, every URL is rewritten, thus obfuscating the true URL of the Java System Calendar Server.

Even though a user is authenticated, that does not mean that the user should have access to other calendar users' data. Within a calendar domain, there exists other layers of security to prevent authenticated users from unauthorized access to other authenticated users' calendar data. One security measure is through the Java System Calendar Server access control entries. Access control enables calendar users to specify who can see their calendars, who can schedule events into their calendars, who can modify their calendars, and who can delete events from their calendars. Access control also enables a user to select who can act on his or her behalf to respond to invitations, schedule or modify events, and delete events. Finally, access control can be used to span domains of users, thus preventing (or enabling) users in one domain from scheduling events with users of another domain.

In addition to access control, the Java System Calendar Server provides an additional level of security at the database protocol level for deployments that separate the calendar front end from the database back end. This level of security is referred to as Database Wire Protocol (DWP) authentication, and utilizes a user name/password pair to authenticate a DWP connection. The userID/password pairs on both the front end and database back end must be identical for a DWP connection to be authenticated.

## High Performance, High Availability

The Java System Calendar Server utilizes several features and techniques to create a high-performance calendaring solution. The very nature of its distributed architecture enables the Java System Calendar Server to be deployed with several front-end calendar Web servers communicating to one or more database back-ends. The AJAX Web client utilizes proxy servers to send requests to the Calendar Server. As client requirements increase, scalability can be enhanced by increasing the number of intermediate proxy and/or front-end calendar Web servers.

In addition to client-focused techniques to enhance performance and scalability, the Java System Calendar Server provides a number of plug-ins and tuning parameters to enhance performance. When multiple database back ends are deployed, it could be a challenge to the calendar server to determine where a given user's calendars reside. The calendar server could employ an algorithmic method, where calendars that begin with the letters A through M are deployed on calendar server 1 and calendars that begin with the letters N through Z are deployed on calendar server 2. This algorithm must be consulted for each calendar retrieved; the more complex the algorithm, the greater the impact could be on calendar server performance.

If the calendar database grows very large and it becomes necessary to add more database back ends, a percentage of the calendars in the large databases must be migrated to the new back-end machines. The algorithm has to change, and the administrator must ensure that all calendars using the old algorithm are migrated per the new algorithm.

This is not an easy task. Furthermore, an algorithmic approach is highly inefficient if the enterprise is dispersed geographically. It stands to reason that if people in the U.S. have their calendars located on a database back-end in the U.S., people in Europe should have their calendars located on a database back-end in Europe, and so on.

To address these problems, the Java System Calendar Server provides an LDAP calendar-lookup database plug-in to locate calendars in a multi-database environment. This allows the administrator to specify an LDAP attribute for each user that identifies the location of that user's calendars. So the attribute could specify a calendar database that is consistent with a user's geographic location. The LDAP attribute can also be automatically provisioned with the appropriate database back end when the user initially logs into the Java System Calendar Server.

The Java System Calendar Server provides a number of tunable parameters that enable the administrator to improve performance. Some of these parameters involve LDAP indexing for certain Java System Calendar Server attributes, enabling the cache for the LDAP calendar lookup database plug-in, and more. Furthermore, the Java System Calendar Server provides utilities for monitoring calendar server statistics such as an assortment of counters.

Counter statistics may be listed about a specified Java System Calendar Server subsystem. For instance, the Java System Calendar Server collects statistics about disk usage; HTTP usage such as the current number of active connections or average connection response time; database usage such as the total number of database reads; and much more. This monitoring is calendar-specific and can be used in addition to general UNIX® or platform-specific monitoring tools that monitor the server environment.

To ensure reliability, the Java System Calendar Server can be configured to be highly available by using Sun Cluster technology. A Sun Cluster is a loosely coupled system of nodes that provides a single system image to clients of network services or applications such as Web, mail, or calendar services. A key feature of Sun Cluster software is that upon detecting a fault, it transparently relocates a service's daemons from one node to another, including host name, Internet Protocol (IP) address, and access to devices configured as part of the service, thus providing a single system image. This capability provides automatic failover when a system shutdown or failure occurs.

The Java System Calendar Server supports all high-availability topologies that are supported by Sun Cluster technology in asymmetric, symmetric, and N+1 configurations.

- **Asymmetric Configurations.** In the asymmetric configuration, the calendar service daemons run only on the primary node and not on the secondary. Detection of a fault in any of the resources (such as storage, host system, or calendar daemons) causes the calendar service to be stopped on the primary and started on the secondary. Each node in the cluster is a complete Solaris™ Operating System (OS) installation, with its own disk that contains the OS, the Java System Calendar Server binaries,

and Calendar Server cluster agents. Each node has at least one network interface connected to the public network and at least two additional private network interfaces that connect to corresponding private network interfaces on the other cluster members; these interfaces are used by the Sun Cluster software on each node for system status monitoring and cluster configuration data-sharing. Only one pair of private network interfaces is in use at any given time; the other is a redundant interface, so there is no single point of failure.

Each node has a connection to the disk cluster that contains the calendar database and configuration files. Although both nodes are continuously connected to the disk cluster, the volumes in the disk cluster are most likely mounted on only one of the nodes at any given time using HAStoragePlus technology.

Only one node in the cluster runs the Java System Calendar Server at any given time. In an asymmetric HA configuration, the configuration files and the calendar database reside on a shared disk. As a result, when a failover occurs, the disk may be unmounted from the failing system and mounted on the surviving system. The logical IP address is then configured on the public network interface of the other system. Users and calendar agents on the public network always connect by using the logical IP address. Reconnecting after a failure automatically connects to the other system. A failover appears to be a very quick crash and reboot of a single system.

Additionally, the secondary node in the cluster is idle as far as Java System Calendar Server usage is concerned. This node, however, remains a fully functional Solaris OS system, and is available for other work as long as procedures to terminate or limit the other work after a failover are in place. When the CPU speed and memory size of the two nodes in the cluster are alike (recommended) in this asymmetric configuration, performance does not suffer during a failover.

- **Symmetric Configurations.** In the symmetric configuration, two calendaring services are configured to run, each connected with its own logical host. Normally, logical hosts are mastered by two different nodes; when one service fails, both logical hosts are mastered by the same physical node, which usually results in a degradation of performance. However, the services keep running. Put another way, in the symmetric configuration, both nodes are active and each node is typically connected to its own database and its own configuration within the shared disk. When one node fails, the load of the failed node is assumed by the surviving node and the surviving node must then handle its own load in addition to the load of the failed node. Hence, the surviving node must be capable of handling the increased load for the time it takes to replace or repair the failed node. Of course, the administrator should repair the failed node as quickly as possible.

*A symmetric configuration makes sense when resources are scarce or when large servers that cannot be dedicated as backups are involved.*

- **N+1 Configurations.** In the N+1 configuration, multiple calendar servers are configured to run as active nodes. Each node typically has its own configuration; there is no sharing of configuration data between these nodes although the disk itself is a shared resource. Essentially, then, each node mounts its own filesystem within the shared disk. In addition to the N active nodes, there is also a single idle node that serves as the backup node. If any of the active nodes fails, the backup node assumes the load of the failed node. Using HAStoragePlus, the failed node most likely unmounts the filesystem to which it is connected, and the standby node mounts the filesystem. Hence, the filesystem becomes a part of the resource group (the collection of resources that the cluster manages as a unit) and is mounted locally on the node that belongs to that resource group.

All Calendar Server Sun Cluster configurations support both HAStorage and HAStoragePlus resource types for making filesystems highly available within a Sun Cluster environment. HAStoragePlus is the recommended resource type because it can work with any filesystem that the Solaris OS supports (UFS, VxFS, etc) and results in significant disk-I/O performance improvement over HAStorage.

### Database Availability

Database availability is a key component of high availability. The Java System Calendar Server provides multiple mechanisms to ensure database consistency and rapid database recovery should there be a database corruption. In terms of database backup and recovery, the Java System Calendar Server supports Archive Backup as well as Hot Backup capabilities. The Java System Calendar Server system records each transaction for the calendar database (additions, modifications, or deletions to calendars and their properties) in a transactional log file. Archive Backup snapshots the live calendar database once each day and then copies the log files from the live database to the archive directory at a given frequency as determined by the system administrator. The Hot Backup function also creates a snapshot of the live database once each day and then copies the log files from the live database to the archive directory at a given frequency as determined by the system administrator. It then applies these transactional log files by initiating a database recover operation on the backed-up database.

This guarantees that in case of a database corruption, there will always be a recoverable database that is at most n seconds old (where n is the frequency determined by the system administrator). The advantage of having both Archive and Hot Backups is that the administrator has two means to recover from a database failure. If, for instance, the live database and the Hot Backup database are corrupt, the administrator can still quickly generate a recovered database from the Archive Backup.

## Event Notification Service (ENS)

ENS is a key feature of the Java System Calendar Server. It is a generic, publish-and-subscribe service that can be used by other applications (not only the Java System Calendar Server) to:

- Notify anyone interested of an event
- Subscribe to upcoming events
- Unsubscribe or cancel existing subscriptions

The Java System Calendar Server uses ENS as its default notification mechanism to provide alarm notifications and server notifications. An alarm notification could be a reminder that a calendar appointment is about to happen or a task is due. A server notification could be an ENS message that a calendar has been created or an event has been modified. There are two parts to the format of a notification: The event reference, which is the URL identifying the event, and the payload, which is the data describing the event. The Java System Calendar Server supports three different payload formats: Binary, text/calendar, and text/XML.

In the case of alarm notifications, the Java System Calendar Server implements the publisher as a daemon, `csadmin.d`. The subscriber is implemented as a daemon, `csnotify.d`. Figure 7 illustrates the interaction between the two Java System Calendar Server daemons and the ENS daemon.

The `csadmin.d` daemon submits notifications to the notification service in the form of event alarms to ENS. The daemon manages the Java System Calendar Server alarm queue and implements a scheduler that lets it know when an alarm has to be generated. At such a point, `csadmin.d` publishes an event. ENS receives the event and dispatches an event notification to all subscribers (in this case, `csnotify.d`).

The `csnotify.d` daemon expresses interest in particular events and receives notifications about these subscribed-to events from ENS. Upon receiving a notification, `csnotify.d` sends notice of the event or task to its clients by e-mail.

As shown in Figure 7, ENS acts as an alarm dispatcher. This decouples alarm delivery from alarm generation and allows for the use of multiple delivery methods such as e-mail, pop-up notifications, and wireless. ENS provides a companion set of APIs that may be used by developers to implement separate publishers and subscribers, extending the capabilities already provided by the Java System Calendar Server. For example, a developer could create a subscriber that is capable of performing a multitude of actions based on the type of notification received. Because ENS is a generic service, it can be used by other applications as well. ENS provides the following APIs:

- Publisher
- Subscriber
- Publish-and-Subscribe Dispatcher

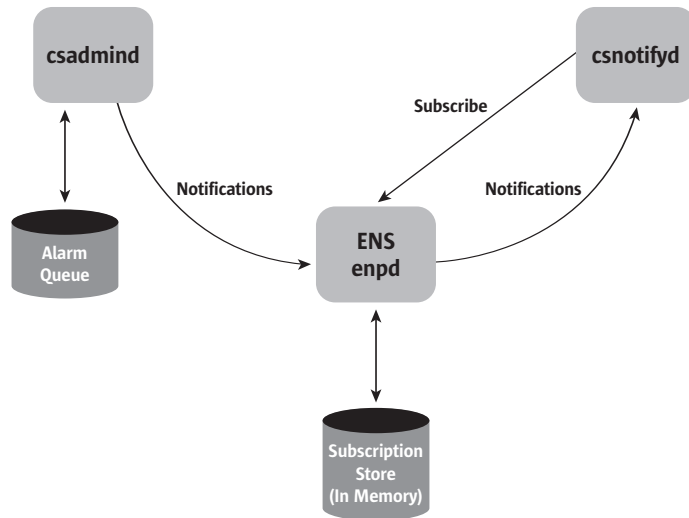


Figure 7. ENS overview

These APIs empower the developer to create very sophisticated integrated services. For example, a developer could create a service that publishes a notification to ENS whenever a certain group — for example, the Internet Engineering Task Force (IETF) — schedules technical conferences. The developer could also create a service or application that subscribes to ENS and, based on a user’s interest, automatically schedules appointments into the user’s calendar that match the user’s criteria. These appointments would include information on the date and location of the IETF conference. Another example could be a stock service that publishes notifications to ENS when certain stocks go above or below a certain threshold. A subscriber service could notify the interested parties of these events by pager, e-mail, SMS, or other means. With respect to Java System Communications products, integration with Java System Instant Messaging is another example of the use of ENS. In this case, a calendar agent subscribes to Java System Calendar Server events and generates an instant messaging alert to the Instant Messaging Server, which manifests itself as a pop-up notification or calendar reminder on the client desktop. The advantage of using an agent to generate the calendar alert is that this approach is modular, distributed, and extensible — without changing the server. In addition to providing alarm notifications, ENS provides server notifications when users perform certain actions on the server. This notification service may be enabled or disabled by setting parameters in the server configuration file.

Currently, the Java System Calendar Server is capable of emitting notifications based on the following actions:

- Calendar is created, modified, or deleted
- Event is created, modified, or deleted
- Task is created, modified, or deleted

Even though ENS is currently the default notification mechanism for Java System Calendar Server, the Java Communications Suite is beginning to embrace a more standards-based approach to event notification. As a result, the Java System Calendar Server also supports the use of the Java System Message Queue to provide a similar notification capability. The Message Queue service implements the Java Messaging Service (JMS) specification, providing a message broker, interfaces to create clients that produce or consume messages, and administrative services and control. JMS provides a more standard, flexible, and reliable approach to event notification than ENS. Although both notification mechanisms are supported in the current version of the Java System Calendar Server, JMS will become the primary notification mechanism and support for ENS will eventually be phased out in a future version of this product.

Developers planning to create their own notifications should consider the Java System Message Queue notification capability rather than ENS. Other benefits of Java System Message Queue include:

- Users can produce messages to a topic or a queue or to both of these delivery methods. Anyone subscribing to the topic will receive the message. If no one subscribes to the topic, the message is discarded. Messages delivered to a queue remain in the queue until either the message times out or a consumer retrieves the message. Only one subscriber can receive the message even if there are multiple consumers waiting for messages in the queue.
- Java System Message Queue offers enhanced load balancing during message distribution, especially when messages are produced to a queue.
- Java System Message Queue supports multiple notification plug-ins that can produce messages to a topic, a queue, to the Event Notification Service, and so on.
- Java System Message Queue provides a reliable notification delivery mechanism. The queue can be configured to be persistent such that if a server goes down, the messages can be retrieved and made available to the appropriate consumer.

For more information on the Java System Message Queue, please consult the *Sun Java System Message Queue 4.1 Technical Overview* at <http://docs.sun.com/app/docs/doc/819-7759>.

## Hosted Domains

Some businesses may choose to outsource their messaging and calendaring needs to large service providers capable of supplying many enterprises with these services. For the service provider utilizing this business model, the ability to host many user domains is critical to the success of the business. The Java System Calendar Server provides support for hosted domains, with each domain sharing the same instance of the software. This means that many domains can exist on a single server.

The Java System Calendar Server supports both LDAP Schema v.1 and LDAP Schema v.2. In LDAP Schema v.1, the LDAP directory organization for a hosted-domain installation includes two trees for domain management: A DC tree and an Organization (OSI) tree. LDAP Schema v.1 can only be used in LDAP-based calendar deployments where the Java System Access Manager is not managing the LDAP directory server. In contrast, LDAP Schema v.2 can be used in both identity-based deployments and LDAP-based deployments, but is required for identity-based calendar deployments. LDAP Schema v.2 supports two modes for directory organization: native mode and compatibility mode. In native mode, the LDAP directory organization for a hosted-domain installation includes a single OSI tree. In compatibility mode, the LDAP directory organization includes both the OSI and DC trees. Because the Java System Calendar Server supports both versions of LDAP Schema, hosted-domain support is available for both LDAP- and identity-based calendar deployments.

The nature of hosted domains is that the end user does not know about users on another domain. The information contained within the domain is not shared with anyone outside the domain unless there is an agreement in place to do so. By default, users may search only within their own domains to invite other users and groups to meetings. However, it may be desirable for users in one domain to invite users in another domain to a meeting. An enterprise may wish to host a calendar server for its partners and establish hosted domains for each of the partners. Some of the partners may wish to collaborate with each other. The Java System Calendar Server provides the ability for users to perform cross-domain searches if both domains have agreed to allow this. In order for cross-domain searches to occur, a specific LDAP attribute for a given domain must be set to the external domain or domains to which users on a given domain have access. Furthermore, the access rights for each domain must allow searches.

For those who wish to take advantage of the hosted-domain support of the Java System Calendar Server and are using a previous version of the Java System Calendar Server that did not support hosted domains, a migration tool is available that modifies the calendar database and LDAP directory to support hosted domains.

## Calendar Server Administration

The Java System Calendar Server provides a flexible administration mechanism that supports both LDAP- and identity-based deployments. Consequently, the Java System Calendar Server may be administrated through either the Communications Services Delegated Administrator graphical user interface, the Communications Services Delegated Administrator command line interface, or the Calendar Server command line interface, depending on the type of deployment. In an identity-based deployment, the Communications Services Delegated Administrator provisions users, resources, and domains. Conversely, the Calendar Server command line interface provisions users,

resources, or domains in an LDAP-based deployment, but it is also used to perform calendar-specific administrative tasks such as creating or deleting calendars, assigning access rights to calendars, starting or stopping the calendar server, and so on.

## The Command Line Interface

The command line interface for performing calendar-specific administrative tasks enables an administrator to develop simple scripts to automatically perform administrative functions. There are a number of command line tools available. One or more users can be assigned as calendar server administrators who typically perform the following functions:

- Start and stop the server
- Create, modify, and delete calendars
- Create, modify, and delete users (also performed by identity-based administration)
- Manage content feeds
- Back up and restore the calendar database
- Create, modify, and delete calendar resources such as conference rooms (also performed by identity-based administration)
- Create, modify, and delete domains (also performed by identity-based administration)

## Administrative Functions

The Java System Calendar Server supports the following administrative functions through the command line interface:

- Manage calendar attributes in the LDAP server
- Back up individual calendars, a user's calendars, or the entire calendar database
- Restore individual calendars, a user's calendars, or the entire calendar database
- Manage calendars and their properties
- Manage events and tasks in a calendar
- Manage calendar databases
- Export/import a calendar in the iCalendar or XML format
- View, enable, or disable configured CSAPI plug-ins
- Manage calendar resource objects such as conference rooms and video equipment
- Manage schedule entries in the Group Scheduling Engine queue (includes listing and deleting entries in the queue)
- Start and stop the Java System Calendar Server
- Monitor calendar statistics such as disk usage, server response time, server session status, and so on
- Refresh one or more services within the server
- Manage calendar users (includes disabling users, creating users, deleting users, or listing a user's calendar attributes)
- Manage calendar domains (includes domain creation, deletion, and modification)
- Create or modify calendar server administrator passwords

## Delegated Administrator Functions

The Communications Services Delegated Administrator provides an intuitive graphical user interface for provisioning calendar users and resources. An administrator can perform the following functions using this tool:

- Create, modify, and delete organizations
- Create, modify, and delete users
- Create, modify, and delete calendar resources such as conference rooms
- Create, modify, and delete user groups
- Manage service packages that define the type of service applied to a user
- Assign organization administrator roles to users and distribute administrative capabilities to these organization administrators

In addition, the Communications Services Delegated Administrator provides a command line utility for performing most of the functions performed by the graphical user interface.

## Migration

Migration from one calendar system to another can be an expensive ordeal and, unless the organization has the right tools and procedures to perform a smooth migration, the cost of migration could seriously impact business and offset any previously perceived benefits. Some service providers and enterprises may choose to migrate to the Java System Calendar Server. Other businesses may already have an earlier version of the Java System Calendar Server deployed, but would like to migrate data to the new Berkeley Database (DB).

To support these scenarios and more, the Java System Calendar Server provides a variety of migration tools. The migration tool `cs5migrate` migrates a Java System Calendar Server 5.x database and LDAP schema to the Java System Calendar Server 6 format and upgrades the calendar database from Berkeley DB version 2.6 to version 4.2. If the service provider or enterprise is currently running the Java System Calendar Server 6.0, the migration to the current Berkeley DB version is seamless and occurs during the installation. In addition, there are other migration tools for migrating data to support the LDAP calendar lookup database plug-in, upgrading an installation to use hosted domains, migrating data from the iPlanet™ Calendar Server 2.x product, migrating LDAP data from Sun LDAP Schema 1 to Schema 2 in preparation for using Access Manager for authentication services, or migrating users and data from Microsoft Exchange Server onto Java Communications Suite servers to support Outlook connectivity to the Java System Calendar Server. Such a migration involves enhancing the recurrence model to support Outlook. The `cs5migrate` utility also creates master records with exceptions for Outlook compatibility.

Finally, in addition to these migration utilities, Sun provides an Enterprise Messaging Migration Service to ensure a successful migration from Microsoft Exchange Server to the Java System Calendar Server. Customers may take advantage of this migration service in order to expedite their migration using a low-cost, low-risk, custom-tailored migration path that takes into account each customer's unique requirements.

## Chapter 3

# Conclusion

The Java System Calendar Server is a compelling solution for any business seeking to provide calendar access to its employees, partners, or customers. It offers several key features that make it an exciting component of the Java Communications Suite.

## High-Performance Server

The Java System Calendar Server is built on an extensible, distributed, and scalable architecture. The server scales both horizontally across machines and vertically across CPUs, and can be enhanced to support an ever-growing number of end users. The distributed nature of the architecture enhances the overall scalability of the calendaring solution, since various calendar server components and services can be distributed throughout the network and run on dedicated hardware if necessary.

## Based on Standards and Proven Technology

The Java System Calendar Server is built on standards such as iCalendar, XML, HTTP, iMIP, iTIP, LDAP, and SMTP. As a result, services and applications that speak these protocols can be easily integrated into the Java System Calendar Server environment.

The Java System Calendar Server integrates seamlessly with the proven, highly scalable Java System Directory Server, Java System Access Manager, Java System Messaging Server, and Java System Instant Messaging products. The integration with other products produces an overall communications solution that is greater than the sum of its parts. Integration with Java System Instant Messaging leverages the alert capability of instant messaging to provide pop-up notifications for calendar appointments. Integration with the Java System Messaging Server produces a tightly integrated communications solution that includes calendar, messaging, and address book.

## Flexibility and Extensibility

The Java System Calendar Server supports multiple devices and clients through the Java Mobile Communications gateway as well as Microsoft Outlook and Evolution connectors. Its open APIs enable extensibility of both the server itself and clients that communicate with it. Examples of clients that have taken advantage of this extensibility include the Lightning calendar extension for the Mozilla Thunderbird client and device synchronization solutions from third party vendors like Notify Technology Corporation and Synchronica.

Java System Calendar Server enables organizations to consolidate to a single calendar server and still use a wide variety of clients such as Thunderbird, Outlook, Evolution, Communications Express, Convergence, and a wide assortment of devices including Blackberry and SyncML devices. The more flexible the calendar solution, the more likely that solution will be able to adapt to individual customer environments and changing requirements.

### **Feature-rich Server and Client**

The Java System Calendar Server provides a solution that is feature-rich on the server and on the client. It addresses the needs of today's organizations that require online and offline support, device synchronization, client customization, secure access, and high availability. Finally, the Java System Calendar Server provides all of this with a lower total cost of ownership than similar solutions available today.

## Chapter 4

# Appendix A

## Standards and References

This section provides a list of relevant standards (under the technology/function categories as indicated) that are supported by the Java System Calendar Server.

### Calendaring

- iCalendar (RFC 2445) — Describes a standard schema for calendar objects on the Internet
- iTIP (RFC 2446) — iCalendar Transport-Independent Interoperability Protocol
- iMIP (RFC 2447) — iCalendar Message-based Interoperability Protocol
- XML — eXtensible Markup Language (REC-XML)
- WCAP — Web Calendar Access Protocol

### Messaging

- IMAP (RFC 2060) — Internet Message Access Protocol
- SMTP (RFC 821) — Simple Mail Transfer Protocol

### Directory Services

- LDAP (RFC 2251) — Lightweight Directory Access Protocol (RFC 2251)

### HyperText Transfer Protocol (HTTP)

- HTTP 1.1 — HyperText Transfer Protocol 1.1 (RFC 2068 and RFC 2616)

## Chapter 5

# References

The following Web sites provide additional information about the Java System Calendar Server and relevant technologies:

### **Sun Products, Tools, and Technologies Documentation**

- [docs.sun.com](http://docs.sun.com)

### **Java System Calendar Server Product Documentation**

- <http://wikis.sun.com/display/CommSuite>

### **Java System Calendar Server**

- [sun.com/software/products/calendar\\_srvr/index.xml](http://sun.com/software/products/calendar_srvr/index.xml)

### **Java System Messaging Server**

- [sun.com/software/products/messaging\\_srvr/index.xml](http://sun.com/software/products/messaging_srvr/index.xml)

### **Java System Instant Messaging**

- [sun.com/software/products/instant\\_messaging/index.xml](http://sun.com/software/products/instant_messaging/index.xml)

### **Java Mobile Communications**

- [sun.com/software/products/java\\_mobile\\_comms/index.xml](http://sun.com/software/products/java_mobile_comms/index.xml)

### **Java System Directory Server Enterprise Edition**

- [sun.com/software/products/directory\\_srvr\\_ee/index.xml](http://sun.com/software/products/directory_srvr_ee/index.xml)

### **Java System Access Manager**

- [sun.com/software/products/access\\_mgr/index.xml](http://sun.com/software/products/access_mgr/index.xml)

## Chapter 6

**Abbreviation/Acronym List**

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
authSDK	Proxy Authentication SDK
CAP	Calendar Access Protocol CSAP Calendar Server API
CUA	Calendar User Agent
DB	Database
DWP	Database Wire Protocol
ENS	Event Notification Service
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transport Protocol
ICAL	Internet Calendaring (also iCalendar)
IETF	Internet Engineering Task Force
JMS	Java Messaging Service
IMAP	Internet Messaging Access Protocol
iMIP	iCalendar Message-Based Interoperability Protocol
IP	Internet Protocol
ISP	Internet Service Provider
iTIP	iCalendar Transport-Independent Interoperability Protocol
LDAP	Lightweight Directory Access Protocol
MAPI	Messaging Application Programming Interface
PAB	Personal Address Book
PIM	Personal Information Manager
PST	Personal Folder Storage
SDK	Software Development Kit
SHTML	Server-Side Include HyperText Markup Language
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
UI	User Interface
URL	Uniform Resource Locator
VoiceXML	Voice eXtensible Markup Language
WCAP	Web Calendar Access Protocol
XML	eXtensible Markup Language
XHTML	eXtensible HyperText Markup Language

