StudioCentral™ Library Overview

007-3830-003

CONTRIBUTORS
Written by Ardene Kannely and Susan Ringler
Illustrated by Dany Galgani and Chrystie Danzer
Production by Diane Ciardelli
Engineering contributions by Seena Cherangara, Hedieh Yaghami

St. Peter’s Basilica image courtesy of ENEL SpA and InfoByte SpA. Disk Thrower image courtesy of Xavier Berenguer, Animatica.

Cover Design By Sarah Bolles, Sarah Bolles Design, and Dany Galgani, SGI Technical Publications.

Record of Revision

Version	Description
002	Sept 1999 Documentation for StudioCentral 2.0 release.
003	March 2001 Documentation for the StudioCentral 3.0 release.

Contents

Figures

About This Guide

This document contains an overview of the StudioCentral Library digital asset management system:

- Chapter 1, “Functional Overview”
- Chapter 2, “StudioCentral Applications”
- Chapter 3, “Software Components”
- Chapter 4, “Configurations”

Audience for This Guide

This guide is for StudioCentral Library administrators, developers, and others who want an overview of StudioCentral Library and how it is used. This guide is a prerequisite to understanding the documents that describe how to install, configure, and administer StudioCentral Library, and how to develop StudioCentral Library applications.

Related Reading

The following are related documents:

- *StudioCentral Library Installation Instructions* (007-3934-*nnn*) is the booklet that accompanies the StudioCentral Library CD and provides information for performing a basic installation.
- *StudioCentral Library Installation and Configuration Guide* (007-3319-*nnn*) describes how to install and configure a basic, advanced, or custom StudioCentral Library system.
- *StudioCentral Library User’s Guide for Macintosh Clients* (007-3989-*nnn*) contains information on how to use the StudioCentral Library Apple® Macintosh® clients to create and manage digital assets.

- *StudioCentral Library Concepts* (007-3974-*nnn*) describes the basic StudioCentral Library concepts, such as how assets are stored and how the Transfer Manager transfers assets.
- *StudioCentral Library Administrator's Guide* (007-3623-*nnn*) contains information on tuning the database; operating the server; troubleshooting problems; administering multimedia assets, users and groups, and Datamodels; and using the Web-based Admin Tool and command-line utilities.
- *StudioCentral Developer's Guide* (007-3246-*nnn*) is for software engineers who develop end-user applications and plug-ins that extend the system.

Conventions

The following type and symbol conventions are used in this guide:

Convention	Meaning
Italics	Used for filenames, glossary entries (online, these show up as underlined), document titles, new terms, onscreen button names, tools, utilities.
Bold	Used for Datamodel and attribute names.
...	Ellipses indicate that a preceding element can be repeated.
nnn	Placeholder for the version number in the part number: 007-1234-*nnn.*
Right angle brackets (>)	Indicate a path through menus to a menu option. For example, "File > Open" means "Under the File menu, choose the Open option."
StudioCentral	Refers to the StudioCentral Library product.

Following are the differences between the online and print versions of this document:

- Cross-reference links appear as quoted strings on paper and as colored text when viewed onscreen.
- At the beginning of each chapter is a list of topics discussed in that chapter. If you are viewing this book online, you can jump to the discussion of any of the listed topics by clicking an item in the list.
- Glossary entries appear in italics in print and as underlined links in the online document. If you view the document online, you can display the glossary definition by clicking the underlined link.

Important Terms in This Book

You will find a glossary in the back of this book. To save you from having to look up many words as you first start reading, this section contains a partial glossary of some of the more important terms.

StudioCentral Terminology

The following terms refer to StudioCentral features and functionality:

Asset The primary unit of storage in StudioCentral. An asset consists of two parts: *metadata* (descriptive information) and *content* (actual digital data). Assets can contain references to other assets, which allows composite assets to be stored.

Content The actual multimedia data: digital images, sound files, and so on. Content is stored separately from metadata.

Datamodel StudioCentral feature that is used for structuring metadata, which is an asset's descriptive information. Datamodels organize and define an asset's metadata and its relationships to other assets using attributes (name/value pairs). Datamodels can be nested. An asset has one or more Datamodels associated with it. The default Datamodel for assets is AmsBase.

Metadata Descriptive information about an asset, such as the compression format of a video asset. Each piece of information is called an *attribute*. Metadata is stored separately from content. Each asset always has system attributes, and may also have Datamodel attributes.

Repository A storage unit for digital assets. Each repository consists of one database, for storing metadata, and one or more content servers, for storing the content. A StudioCentral server may include multiple repositories in either a single-server or multi-server configuration. The multiple repositories are managed by a single Ams Server process.

StudioCentral Library User Terminology

The following terms refer to persons who use or administer StudioCentral Library:

Administrator Individual who supports end users by performing tasks such as defining Datamodels, starting and stopping servers, giving users access to assets, and so on.

Developer Software engineer who writes end-user application programs and plugins that use the StudioCentral AMP protocol to access StudioCentral.

End user User of StudioCentral Library application programs, such as the Macintosh client.

Chapter 1

Functional Overview

This chapter contains a functional overview of StudioCentral Library (called StudioCentral here), which is a digital asset management system that is used to store, manage, retrieve, preview, and distribute multimedia data. This data includes video clips, movie frames, sound files, animation cells, presentation storyboards, engineering designs, documentation, or any type of data that can be stored in a digital format.

StudioCentral is a complete solution that provides end users with desktop tools for managing their digital assets. The tools can be used with SGI platforms, Web browsers, and desktop computers such as Macintosh, in installations that range from small workgroups to enterprise-wide installations.

Digital assets stored in StudioCentral consist of:

- Content, which is the actual multimedia data, such as the graphic, digital video, or digital audio. Content can be stored in any format desired, such as JPEG or MPEG.
- Metadata, which is descriptive information about the multimedia data. The metadata consists of attributes that describe the asset (for example, its name and format) and its relationship to other assets.

Note: Administrators and application developers can customize the metadata attributes stored with an asset by defining StudioCentral Datamodels.

By separating the descriptive information from the content, each can be stored in a way that optimizes its access. For example, the metadata is stored in industry-standard databases, letting users search the descriptive information for assets that match certain criteria without having to read the content. And, the content can be stored in a variety of devices, including online, hierarchical, and archive storage, enabling efficient use and storage.

All assets in a StudioCentral system have metadata associated with them, but not all assets have content. For example, an asset that is used to group assets that contain different digital formats of the same video content, is a composite asset and may not have content itself. However, the assets that it groups, or points to, do have content.

Important features of the StudioCentral digital asset management system are:

- Asset Versioning: multiple versions of the same asset can be stored, letting users track revisions that are made to the asset.
- Unrestricted data modeling: data can be organized to facilitate workflow or other needs. There are no fixed schemas.
- Flexible configurations: StudioCentral supports a wide range of configurations from a small one-server system with a few clients, to an enterprise-wide multi-server system with remote database and content stores, and many clients.
- Asset transfer: simple or composite assets can be transferred to another StudioCentral system by using the StudioCentral Admin Tool Transfer manager, or the command-line utilities. Transfers may be query-based and many flexible options are supported.
- Powerful metadata search
- Out of the Box and Extensibility: StudioCentral is a complete client-server system with clients for asset creation and asset management. It also can be readily integrated into legacy or other software through use of the Plugin Server, the Search Server (for content search), and XML output support.
- Automatic asset checkin and cataloging for multimedia assets.

The content for an asset is brought into the StudioCentral server by copying it from external files or transferring it from another StudioCentral server. When the content is checked into (that is, stored in) StudioCentral, it is cataloged. During this cataloging, or asset typing, process, if the content file's digital format is one of the types recognized by StudioCentral, the following steps are performed:

- The content file's header is read.
- Information about the asset is extracted from the content file and used to populate the asset's metadata. The attributes that are used are based on the type of the asset. For example, when a video asset is cataloged, the duration, digital format, bit rate, and image size are stored in metadata attributes.
- If appropriate, a thumbnail, which is a pictorial representation of the asset, is generated. For example, a thumbnail is generated for movie assets but not for audio assets.

- The metadata and content are stored in StudioCentral.

When assets are checked in, the end user or StudioCentral administrator can use graphical tools or command-line utilities to enter metadata attribute values, in addition to the ones that are automatically generated during the asset typing process. For example, the user may want to enter keywords, notes for the editor who will be reviewing an asset, or a description of a new version of an existing asset.

Once assets are stored in a StudioCentral system, they can be:

- Queried, where users search for assets. StudioCentral provides the Asset Query Language (AQL) that lets users search for assets based on criteria that they specify. For example, a user can search for all assets checked in by Joe Smith since July 9.
- Browsed, where users scan lists of assets on their desktops. For example, a user can perform a query that returns several assets, and then browse the returned assets before deciding which content to view.
- Viewed, where the asset's content is viewed on the desktop. For example, an asset containing a movie can be viewed by having Kasenna MediaBase stream the content to the desktop.
- Checked out (that is, reserved) or retrieved (that is, not reserved) by another user. This enables sharing of assets by members of a work group.
- Sent to a video server for playout. For example, assets can be transferred to a video server, such as SGI's Video Server, for real-time, frame-accurate playback of broadcast-quality digital media data.
- Sent to a MediaBase server for streaming.
- Transferred to another StudioCentral system. For example, a central StudioCentral system may store all assets managed by a news department. Individual assets can then be transferred from that central system to another StudioCental system in a regional office.
- Moved to an archive system, in either near-line or offline storage, which frees the online storage for the content of frequently used assets.

This functionality is shown in Figure 1-1.

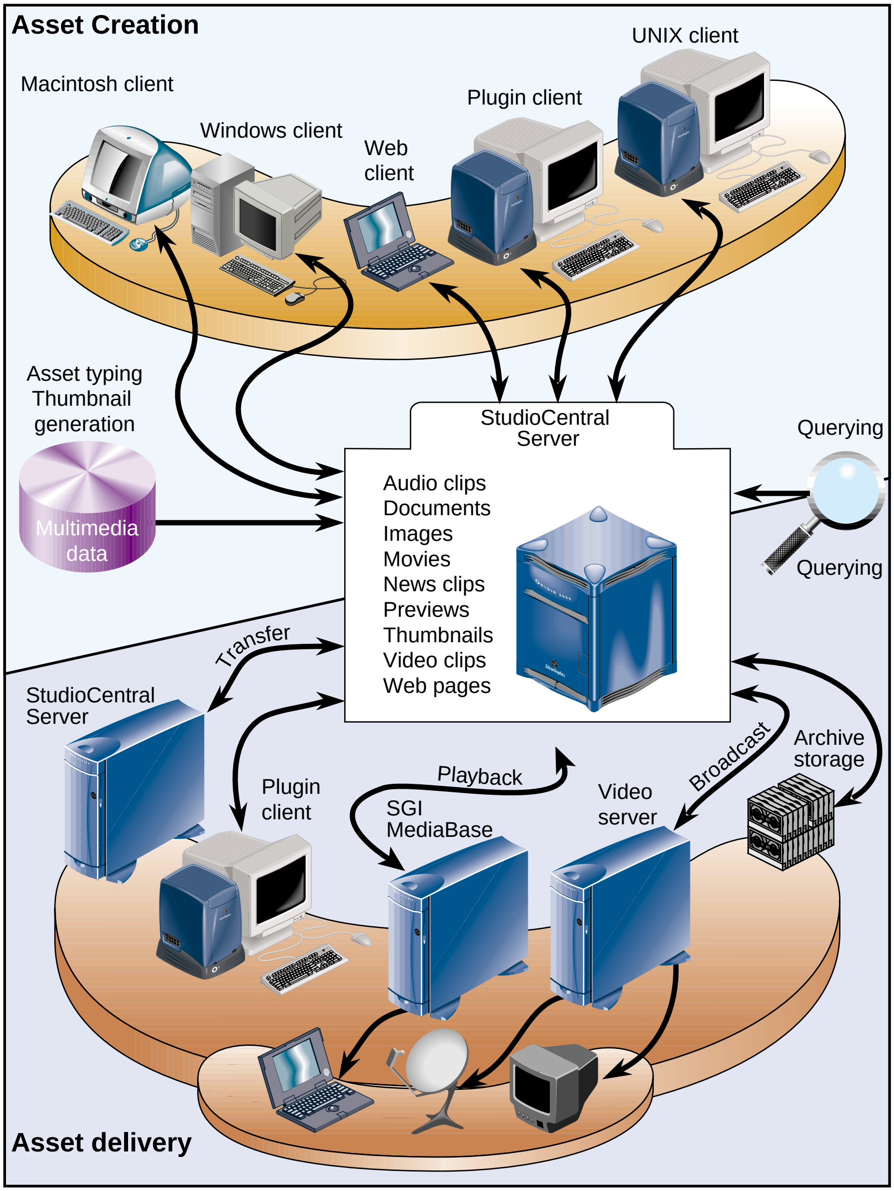


Figure 1-1 Functional Overview

A number of clients are available with StudioCentral, including:

- StudioCentral Macintosh client that runs on Apple Macintosh computers and is used to store, query, view, and manage digital assets. Chapter 2, "StudioCentral Applications," has examples of such client applications.
- Admin Tool, a Web-based client, performs administration tasks such as monitoring system status, transferring assets, managing users, and setting up and administering security. Chapter 2, "StudioCentral Applications," has examples of this tool.
- Third-party applications, developed by SGI partners, perform tasks such as cataloging, archiving, and browsing assets.

Note: StudioCentral also has command-line utilities, which are documented in the *StudioCentral Library Administrator's Guide*, that can be used to perform most of the tasks available through the desktop clients and the Admin Tool.

In addition to these applications, software engineers can use StudioCentral services or the StudioCentral Asset Management Protocol (AMP) to develop applications that are tailored to specific needs. (See "Developing New StudioCentral Components" on page 31 for more information.)

StudioCentral Applications

This chapter presents examples of StudioCentral client applications, which store, query, view, and manage assets, and the Admin Tool, which is a Web-based application to administer StudioCentral repositories, users, Datamodels, and assets:

- "StudioCentral Clients"
- "Admin Tool" on page 18

StudioCentral Clients

The StudioCentral clients allow end users to store, retrieve, query, and manage StudioCentral assets using their native desktop environments. The clients can be used in a distributed environment in which assets are shared by workgroup members.

The StudioCentral clients are integrated with other native desktop applications, allowing assets to be dragged and dropped between StudioCentral and other applications such as Adobe Photoshop. Digital assets can also be stored in StudioCentral by having it monitor a specific directory so that new or modified files in that directory are automatically brought into StudioCentral. For example, a user working in Photoshop can save files into the directory that the StudioCentral Checkin Service monitors. Each time the user stores a new image in the Photoshop directory, the image is automatically stored in StudioCentral, enabling the user to check in assets without leaving the Photoshop application.

Regardless of how assets are brought in, the StudioCentral asset typing facility can catalog them automatically, storing metadata attributes that are appropriate for each asset type.

Once assets are stored in StudioCentral, the client can be used to search for assets that match specified criteria and then to view the assets on the desktop. For example, the user might want to search for and view all Photoshop assets created by a specific user on a given date.

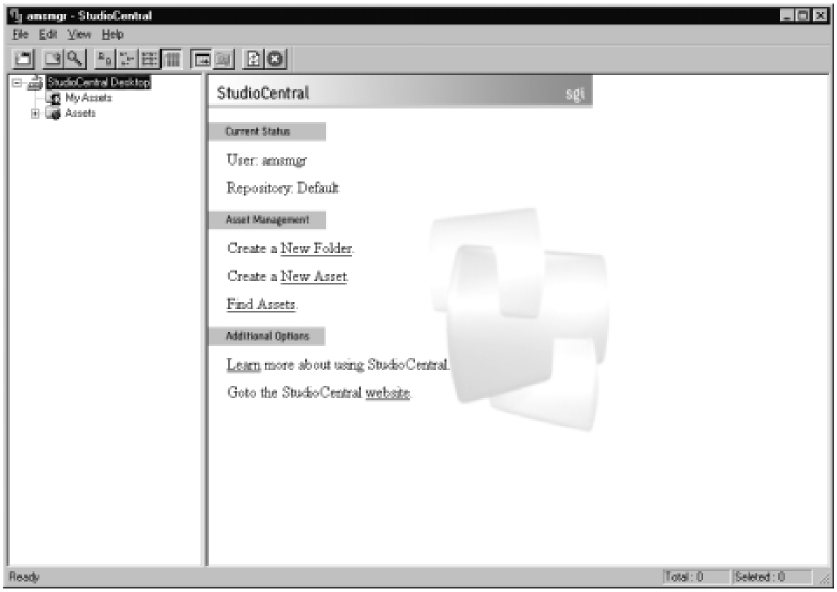
The following sections present scenarios using Macintosh and Windows client applications to illustrate how end users would store and retrieve assets. The same scenario is used for each client:

- “StudioCentral Windows NT Client Scenario” on page 8
- “StudioCentral Macintosh Client Scenario” on page 13

StudioCentral Windows NT Client Scenario

The following scenario illustrates how a StudioCentral Windows NT client helps an artist manage the creation of a new image for an advertisement. The artist needs to select an image to start from and have an editor review it. The artist wants to be able to save several versions of the image and wants to guarantee the work against loss:

- The artist obtains his specifications and logs in to his Windows NT desktop.
- He uses Windows Explorer to create a new directory for his local files. He then uses Adobe Photoshop to create his first image, storing it in the new directory.
- After creating his Photoshop file, the artist is ready to check it in to StudioCentral and have his editor review it. He starts StudioCentral by double-clicking the StudioCentral icon on his desktop, and then he logs in with his StudioCentral username and password. After he is logged in, the StudioCentral Explorer window is displayed, as shown in Figure 2-1.



User: amsmgr

Repository: Default

Create a New Folder.

Create a New Asset.

Find Assets.

Learn more about using StudioCentral.

Goto the StudioCentral website.

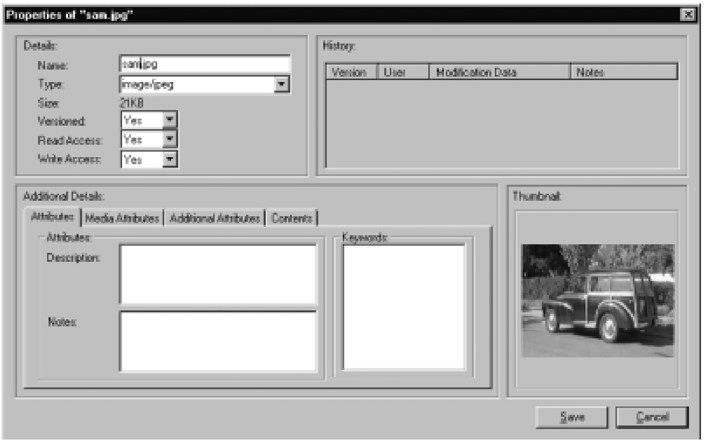
Total : 0

Selected : 0

Figure 2-1 Windows NT Client: StudioCentral Explorer Window

The right frame provides a user-friendly interface for quickly performing basic asset management tasks. When the Explorer is first displayed, it shows only the top-level folders in the left frame. Folders and assets within these top-level folders are displayed when the folder is selected. The artist clicks the Assets folder to list all the folders.

- From the artists's folder in his Windows Explorer, he drags the image he wants to use and drops it into his folder in the StudioCentral Explorer window. After he drops the file, StudioCentral displays the Properties window, which is shown in Figure 2-2. The Properties window is displayed when the checkin preference is set to manual mode, which is the default setting.



Name: sam.jpg
Type: image/jpeg
Size: 21KB
Versioned: Yes
Read Access: Yes
Write Access: Yes

Figure 2-2 Windows NT Client: Properties Window

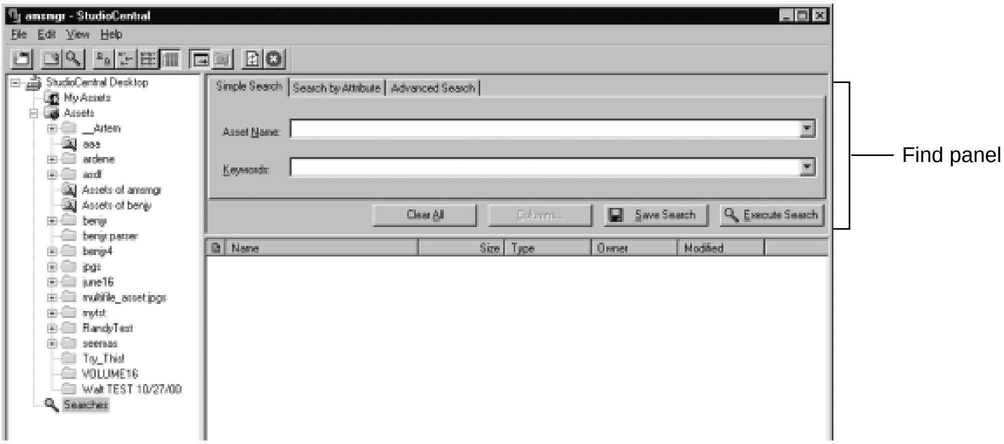
- The Preferences window lets the artist enter information for the asset. He enters a description for the project, Acme car, in the text field. Then, he clicks the ***Save*** button.

  Note: In addition to the information the artist enters during checkin, StudioCentral reads the file and generates metadata attributes that are appropriate for the type of content it contains.

- After the asset is checked in, the artist tells his editor that it is in StudioCentral. The editor can now review his work.

The editor uses her Windows NT workstation to find and review the artist’s work:

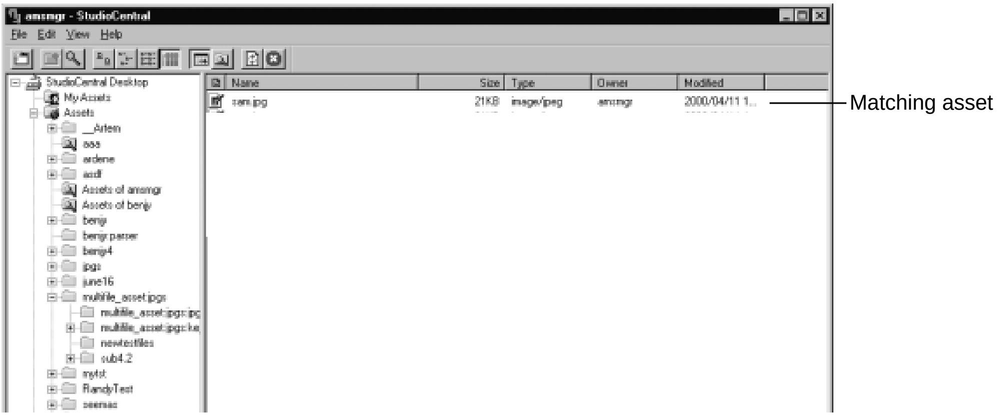
- The editor uses the Find panel in the Explorer window to look for the asset. Figure 2-3 shows the Find panel.



Find panel

Figure 2-3 Windows NT Client: Find Panel

- The editor enters the name of the asset in the text box. After the editor clicks the ***Execute Search*** button, the results of the search are displayed in the Explorer window, as shown in Figure 2-4.

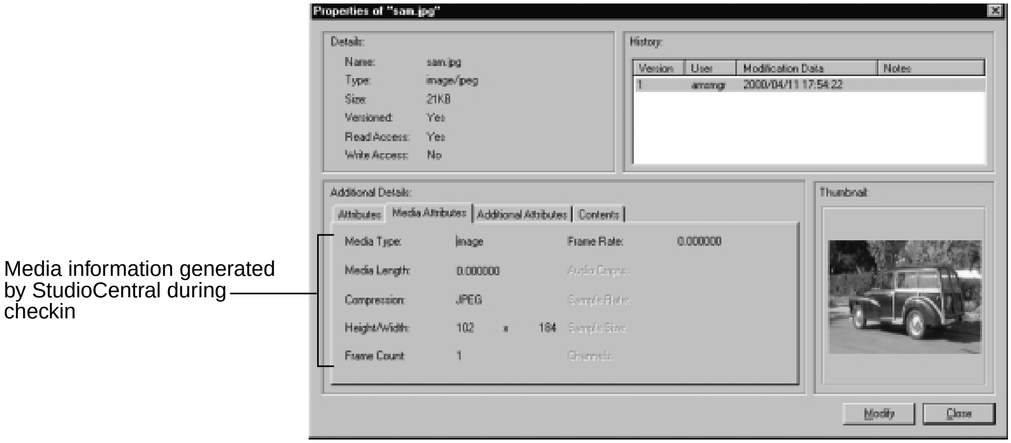


Name	Size	Type	Owner	Modified
sam.jpg	21KB	image/jpeg	amsmgr	2000/04/11 1...

Matching asset

Figure 2-4 Windows NT Client: Find Results Display

- The editor selects the asset and right clicks to invoke the popup menu for asset operations. She selects the Properties menu item to display information for the asset. Figure 2-5 shows the Properties window.



Properties of "sam.jpg"

Details:

Name: sam.jpg
Type: image/jpeg
Size: 21KB
Versioned: Yes
Read Access: Yes
Write Access: No

History:

Version	User	Modification Data	Notes
1	amsmgr	2000/04/11 17:54:22	

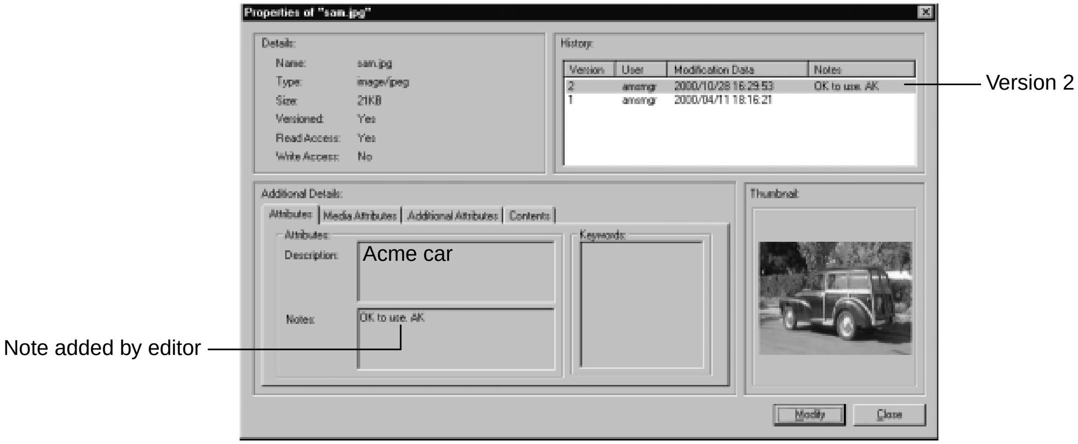
Additional Details:

Media Type: Image
Media Length: 0.000000
Compression: JPEG
Height/Width: 102 x 184
Frame Count: 1
Frame Rate: 0.000000
Audio Chnnls:
Sample Rate:
Sample Size:
Channels:

Media information generated by StudioCentral during checkin

Figure 2-5 Windows NT Client: Properties Window

- The editor clicks the Media Attributes tab to look at the image size and type. This information was generated by the StudioCentral content typing process during checkin of the asset.
- The editor approves the asset, and enters her approval note in the Notes section of the Properties window. After making her changes, she checks the asset back in to StudioCentral. The asset now has two versions as shown in Figure 2-6.



Properties of "sam.jpg"

Details:

Name: sam.jpg
Type: image/jpeg
Size: 21KB
Versioned: Yes
Read Access: Yes
Write Access: No

History:

Version	User	Modification Data	Notes
2	amsmgr	2000/10/28 16:29:53	OK to use. AK
1	amsmgr	2000/04/11 18:16:21	

Additional Details:

Attributes:

Description: Acme car

Notes: OK to use. AK

Keywords:

Version 2

Note added by editor

Figure 2-6 Windows NT Client: Second Version of Asset in Properties Window

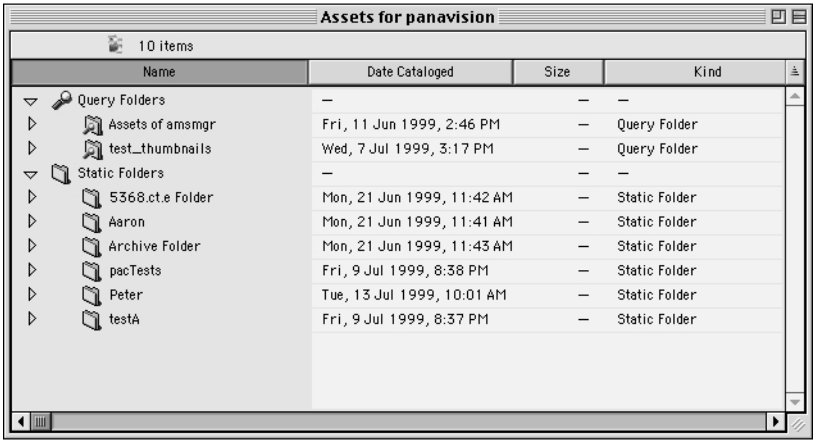
- Now the artist can read the editor’s notes and proceed.

The *StudioCentral Library User’s Guide for Microsoft Windows Clients* describes the StudioCentral Windows client in detail.

StudioCentral Macintosh Client Scenario

The following scenario illustrates how a StudioCentral Macintosh client helps an artist manage the creation of a new set design for a television show. The artist needs to create an image and have an editor review it. The artist wants to have several versions of the image and wants to guarantee the work against loss:

- The artist obtains his specifications and creates a new folder on his Macintosh for his local files. He then uses Adobe Photoshop to create his first image, storing it in the new folder.
- After creating his Photoshop file, the artist is ready to check it in to StudioCentral and have his editor review it. He starts StudioCentral by double-clicking the StudioCentral icon on his desktop, and then he logs in with his StudioCentral username and password. After he is logged in, the StudioCentral Asset Browser window is displayed, as shown in Figure 2-7.



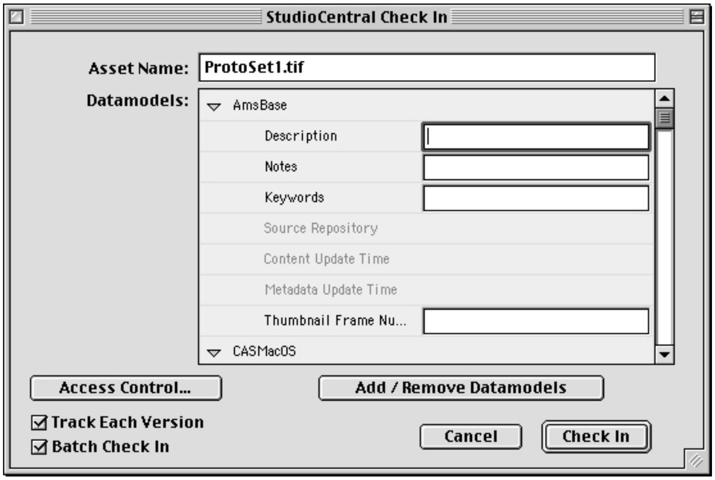
Assets for panavision

10 items

Name	Date Cataloged	Size	Kind
▽ Query Folders	—	—	—
▷ Assets of amsmgr	Fri, 11 Jun 1999, 2:46 PM	—	Query Folder
▷ test_thumbnails	Wed, 7 Jul 1999, 3:17 PM	—	Query Folder
▽ Static Folders	—	—	—
▷ 5368.ct.e Folder	Mon, 21 Jun 1999, 11:42 AM	—	Static Folder
▷ Aaron	Mon, 21 Jun 1999, 11:41 AM	—	Static Folder
▷ Archive Folder	Mon, 21 Jun 1999, 11:43 AM	—	Static Folder
▷ pacTests	Fri, 9 Jul 1999, 8:38 PM	—	Static Folder
▷ Peter	Tue, 13 Jul 1999, 10:01 AM	—	Static Folder
▷ testA	Fri, 9 Jul 1999, 8:37 PM	—	Static Folder

Figure 2-7 Macintosh Client: Asset Browser

- From his Macintosh folder, the artist drags the file he created and drops it into the StudioCentral Asset Browser window. After he drops the file, StudioCentral displays the Check In dialog, which is shown in Figure 2-8.



Asset Name: ProtoSet1.tif

☑ Track Each Version

☑ Batch Check In

Figure 2-8 Macintosh Client: Check In Window

- The Check In window lets the artist enter information for the asset. He adds the keyword for the project, which is Acme, and then he clicks the ***Check In*** button.

Note: In addition to the information the artist enters during checkin, StudioCentral reads the file and generates metadata attributes that are appropriate for the type of content in the asset.

- After the file is checked in, the artist tells his editor that the asset is in StudioCentral. The editor can now review his work.

The editor uses her Macintosh to look for and review the artist's work.

- The editor uses the Find window to look for the asset. Figure 2-9 shows the Find window.

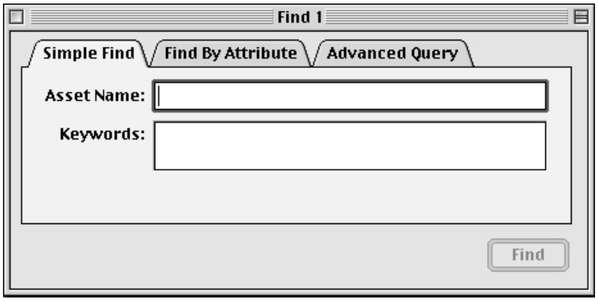
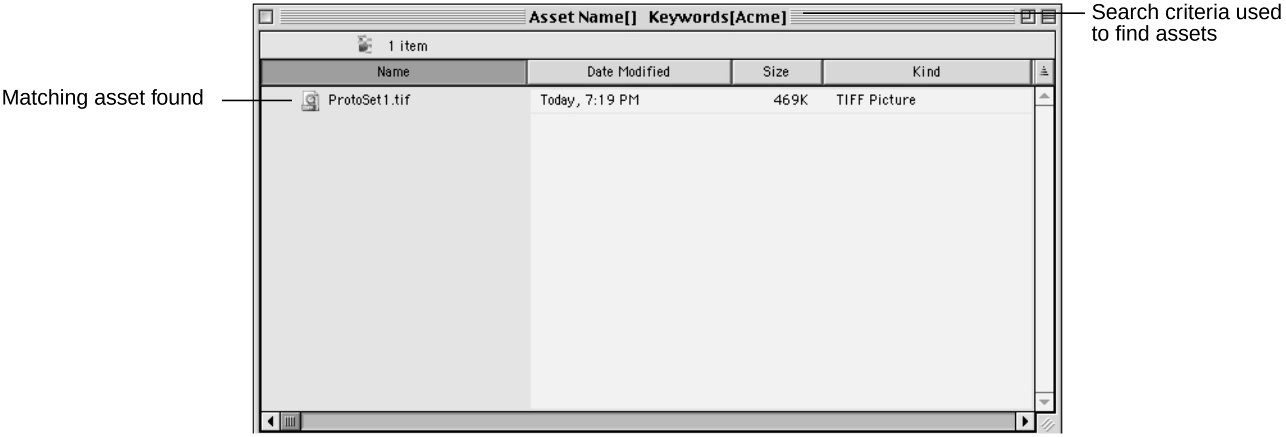


Figure 2-9 Macintosh Client: Find Window

- The editor enters the keyword for the project in the Keywords text box and clicks the ***Find*** button. The results of the search are displayed in a results window, as shown in Figure 2-10.



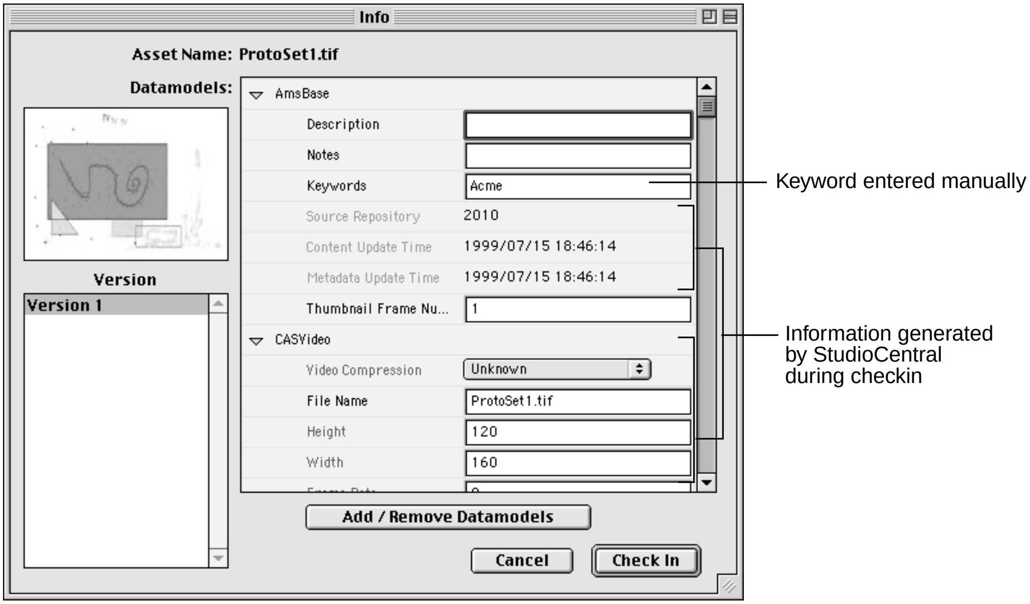
Name	Date Modified	Size	Kind
ProtoSet1.tif	Today, 7:19 PM	469K	TIFF Picture

Search criteria used to find assets

Matching asset found

Figure 2-10 Macintosh Client: Find Results Window

- The editor selects the asset and chooses File > Get Info to look at the information for the asset. Figure 2-11 shows the Information window, which includes the information that was entered manually and that was generated by StudioCentral during the checkin process.



Asset Name: ProtoSet1.tif

Datamodels:

AmsBase

Description	
Notes	
Keywords	Acme
Source Repository	2010
Content Update Time	1999/07/15 18:46:14
Metadata Update Time	1999/07/15 18:46:14
Thumbnail Frame Nu...	1

CASVideo

Video Compression	Unknown
File Name	ProtoSet1.tif
Height	120
Width	160

Version

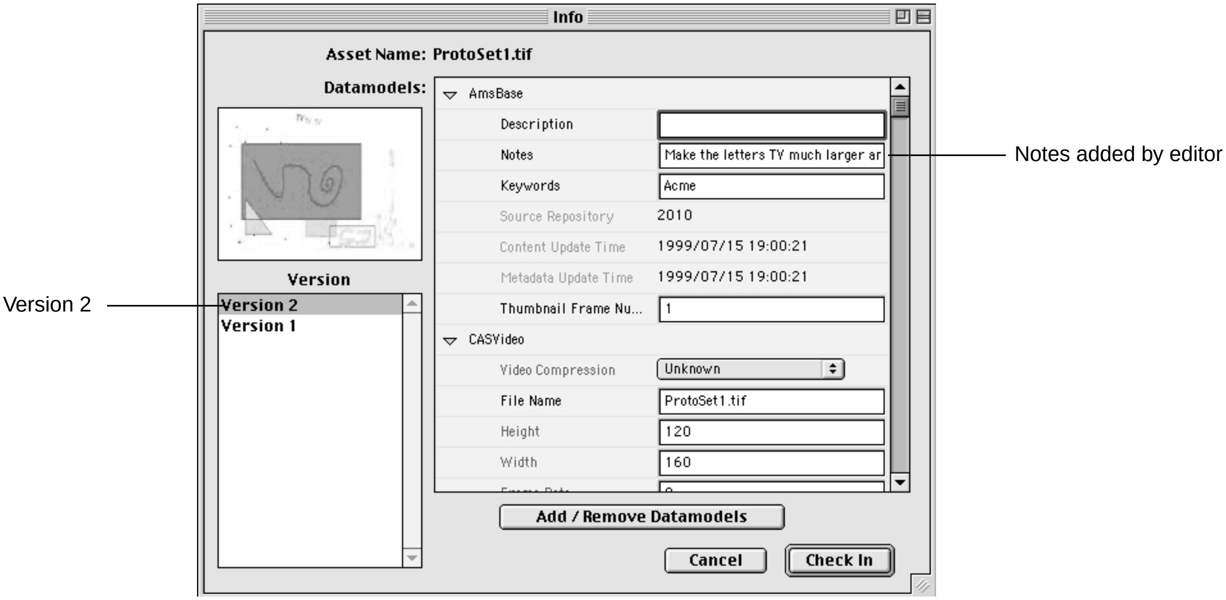
Version 1

Keyword entered manually

Information generated by StudioCentral during checkin

Figure 2-11 Macintosh Client: Information Window

- The editor wishes to make a change to the asset, so she retrieves it by dragging Version 1 from the Version area of the Info window to her Macintosh desktop. Then she clicks the ***Cancel*** button to close the Info window.
- The editor uses her Photoshop application to adjust the colors. After making her changes, she checks the asset back in to StudioCentral, adding comments in the notes field about the changes she made and how she wants the artist to proceed. Checking in the asset again creates a new version of the asset. The asset now has two versions as shown in Figure 2-12.



Asset Name: ProtoSet1.tif

Datamodels:

Version

Version 2
Version 1

AmsBase

Description:
Notes: Make the letters TV much larger ar
Keywords: Acme
Source Repository: 2010
Content Update Time: 1999/07/15 19:00:21
Metadata Update Time: 1999/07/15 19:00:21
Thumbnail Frame Nu...: 1

CASVideo

Video Compression: Unknown
File Name: ProtoSet1.tif
Height: 120
Width: 160

Version 2

Notes added by editor

Figure 2-12 Macintosh Client: Second Version of Asset in Information Window

- Now the artist can read the editor's notes, retrieve the new version by dragging and dropping Version 2 from the Info window to his computer, and make his changes in Photoshop.

The *StudioCentral Library User's Guide for Macintosh Clients* describes the StudioCentral Macintosh client in detail.

Admin Tool

The Admin Tool, which runs in a Web browser, performs repository and asset administration tasks. For example, the tool is used by administrators to give users access to StudioCentral assets, transfer assets, and check the repository for consistency.

Figure 2-13 shows how the tool is used to add a user to a group. (A "group" is useful when several users have the same privileges in StudioCentral because it lets the administrator assign the privileges to the group instead of to each user individually.) In this case, the user "hsmith" is being added to the "production" group.

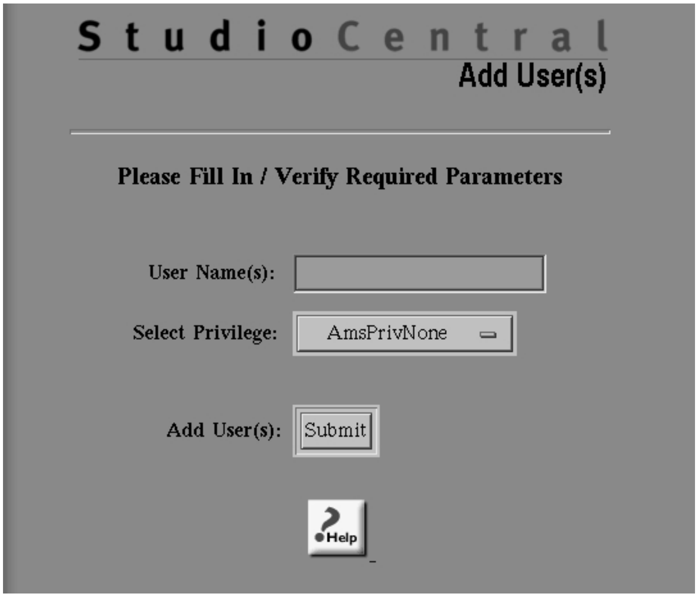
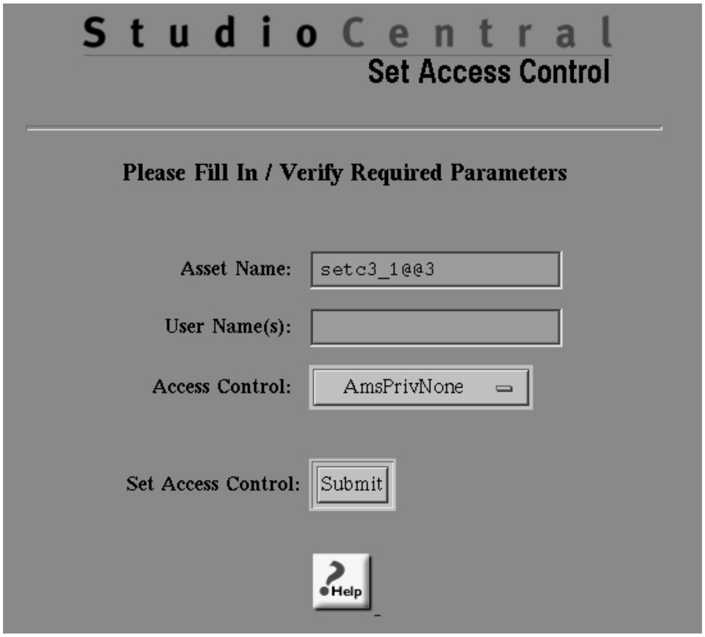


Figure 2-13 Admin Tool: Adding a User to a Group

The tool can also be used to define new users and groups, give users and groups permission to access certain assets, and remove users and groups. For example, Figure 2-14 shows how the user "hsmith" receives permission to read the second version of an asset whose name is "brandNew2." ("@@2" indicates the version number.) In this case, because the permission is given only to the individual user and not to the group, other users in the group cannot read the asset unless permission to do so is given to those users individually or to the group as a whole.



StudioCentral
Set Access Control

Please Fill In / Verify Required Parameters

Asset Name: setc3_1@@3

User Name(s):

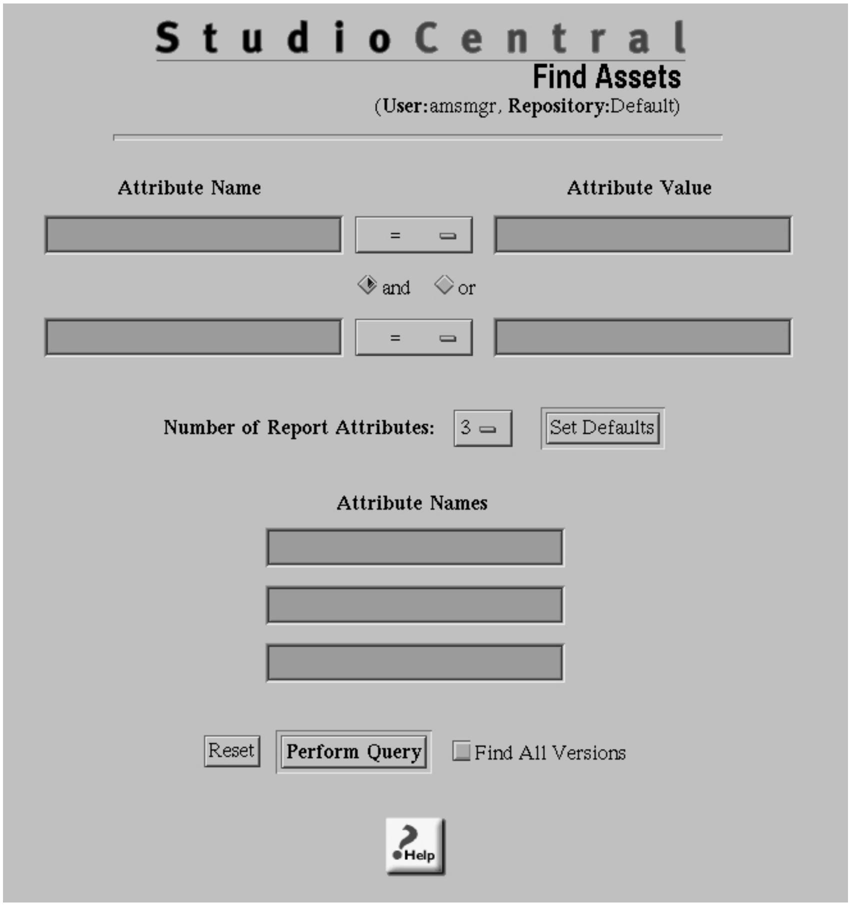
Access Control: AmsPrivNone

Set Access Control: Submit

Figure 2-14 Admin Tool: Setting Permissions for an Asset

The Admin Tool can search StudioCentral for assets that match criteria that are either predefined or entered by the user. For example, Figure 2-15 shows how to search for assets whose content is at least 10 gigabytes (that is, 10,000,000,000 bytes). If this search is performed frequently, the user can easily add it to the Admin Tool as a predefined search. This lets the user perform the search again, without having to enter the criteria.

When the search in Figure 2-15 is performed, a table that identifies the assets that satisfy the search criteria is displayed. The table displays the name of each asset as a hyperlink, and the size of each asset's content (content_length). Clicking the hyperlink displays the metadata for the asset.



StudioCentral

Find Assets

(User:amsmgr, Repository:Default)

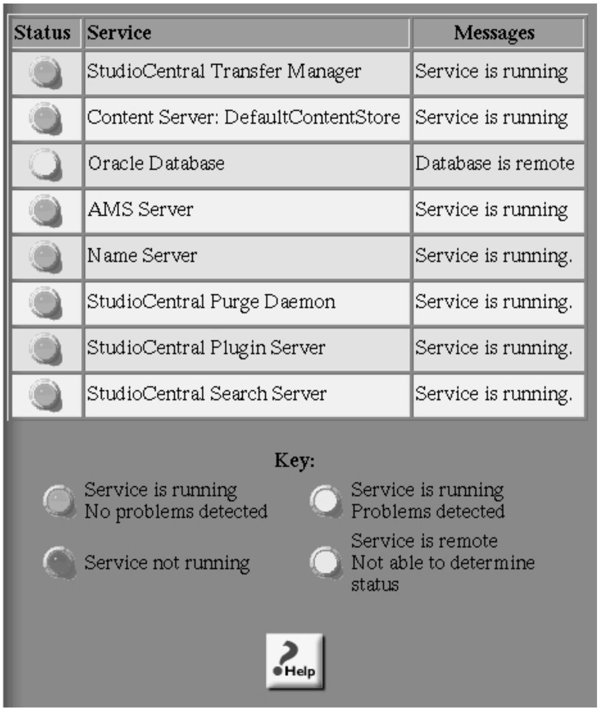
◆ and ◇ or

Number of Report Attributes: 3

□ Find All Versions

Figure 2-15 Admin Tool: Searching for Assets

The Admin Tool can also start, stop, and determine the status of services that run as part of StudioCentral. For example, Figure 2-16 shows the page that displays the status of services. For each service, a colored bullet to the left of the service name indicates its status. The color of the bullet indicates whether the associated service is up and running, running but problems have been detected, running on a remote server, or not running.



Status	Service	Messages
●	StudioCentral Transfer Manager	Service is running
●	Content Server: DefaultContentStore	Service is running
○	Oracle Database	Database is remote
●	AMS Server	Service is running
●	Name Server	Service is running.
●	StudioCentral Purge Daemon	Service is running.
●	StudioCentral Plugin Server	Service is running.
●	StudioCentral Search Server	Service is running.

Key:

● Service is running No problems detected

○ Service is running Problems detected

● Service not running

○ Service is remote Not able to determine status

Figure 2-16 Admin Tool: Determining Status of Services

The Admin Tool is based on industry-standard practice for Web browsers and Web servers, and is easily modified and customized. It is described in detail in the *StudioCentral Library Administrator's Guide*.

Chapter 3

Software Components

This chapter describes the StudioCentral Library software components and the ways in which they can be used to develop new applications:

Software Components

StudioCentral software contains the following components:

- *Core software*, which provides both the essential functionality that enables assets to be stored in, and retrieved from, StudioCentral and the infrastructure software (plugin server, search server) that facilitate the rapid development of complex data systems around a StudioCentral core.
- *Asset Management Protocol* (AMP), which provides developers a rich command set for accessing the content and metadata of StudioCentral assets. AMP can be used to quickly develop end-user applications since it can be embedded in any language.
- *Asset Services*, which provide optional functionality that extends the breadth of services StudioCentral can provide (transfer management, content management).
- *StudioCentral Client software,* which provides interfaces for many concurrent users on industry-standard desktop platforms to store, retrieve, query, and manage assets and to administer an enterprise-wide StudioCentral installation.

Figure 3-1 shows the primary software components that make up the StudioCentral software.

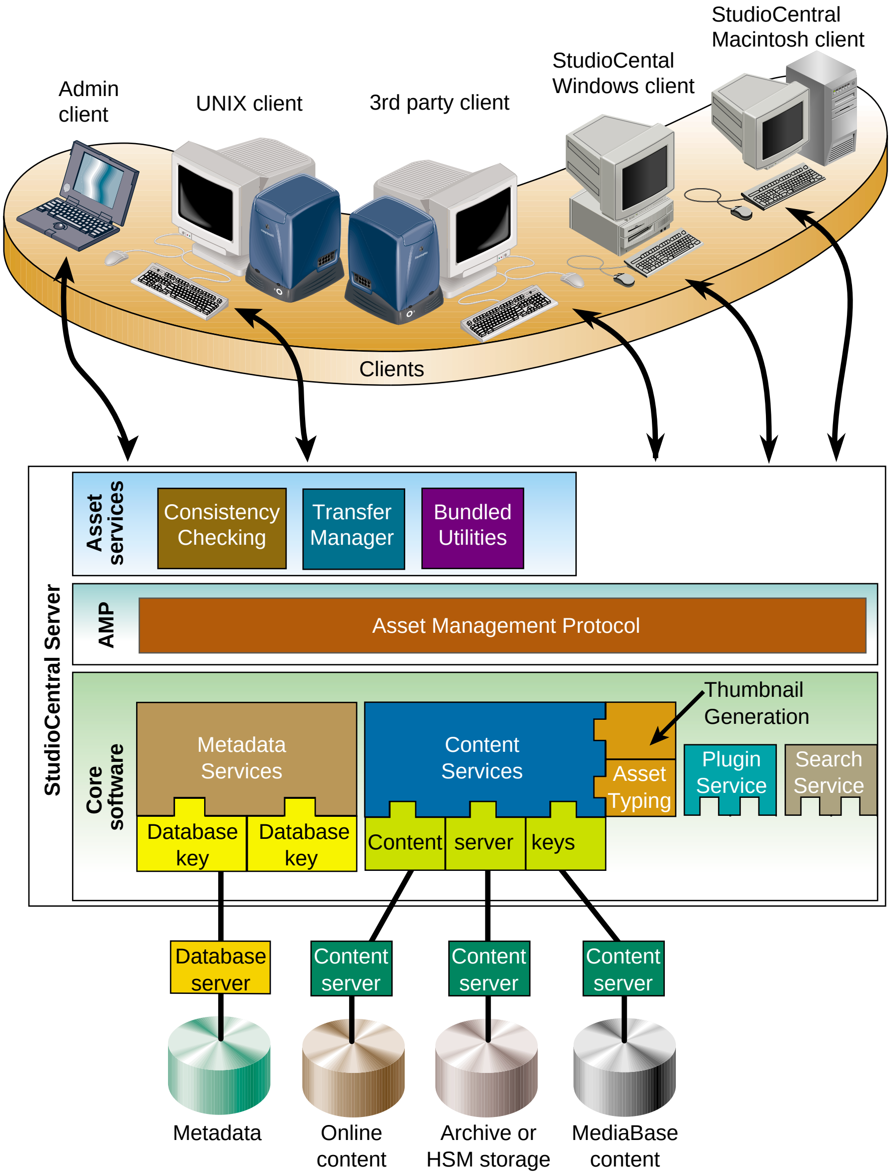


Figure 3-1 StudioCentral Software Components

The remainder of this section describes the components shown in Figure 3-1:

- “Core Software” on page 25
- “Asset Management Protocol” on page 28
- “Asset Services” on page 28
- “StudioCentral Clients” on page 29

Core Software

This subsection describes the core software, which provides the essential functionality for the StudioCentral system:

- “Metadata Services” on page 25
- “Content Services” on page 26
- “Database Keys” on page 27
- “Content Server Keys” on page 27
- “Plugin Server” on page 27
- “Search Server” on page 27
- “Asset Typing” on page 27
- “Thumbnail Generation” on page 28

Metadata Services

The StudioCentral metadata services store, retrieve, and manage the metadata for StudioCentral assets. These services provide fully flexible data modeling which allows the mapping of StudioCentral metadata into industry-standard database management systems without imposing a fixed schema. Metadata services also include a feature-rich query language (AQL - Asset Query Language) to facilitate speedy data retrieval from StudioCentral clients. The metadata services are grouped together in the Ams Server process, which is the core process of the StudioCentral server. A particular StudioCentral server has a single Ams Server process, but can have multiple repositories, and multiple content servers.

The StudioCentral Ams Server is independent of the specific database used. The database-dependent code exists in a software component called a *database key*, which acts as an interface between the Ams Server and the actual database management system.

These database keys permit StudioCentral to take advantage of features specific to a given database.

StudioCentral currently has database keys for the following:

- Informix Dynamic Server (IDS)
- Informix Dynamic Server with Universal Data Option
- Oracle Server

Content Services

Content Services store, retrieve, and manage the digital content of StudioCentral assets. This content can be images, audio files, or any format that can be digitally stored.

As with Metadata Services, Content Services are independent of the specific content storage that is used. The content storage-dependent code exists in software components called *content server keys*, which are interfaces between the Ams Server and the content servers that manage the actual content stores. These content server keys enable all StudioCentral applications to be independent of the storage used. They also permit StudioCentral to take advantage of features specific to a given storage device.

Supported storage types include:

- *Online storage*, which provides the fastest access to asset content and is normally used for currently active assets. Online storage is used primarily for frequently accessed content.
- *Hierarchical storage*, where assets can be moved between online, near-line, and offline (archive) storage, either automatically based on use, or manually at the user’s request. Hierarchical storage is provided by the Data Migration Facility (DMF) system developed by SGI. It provides access to media content by quickly bringing offline content onto a disk cache for fast access.
- *Archive storage*, which enables online storage to be freed for just the content that is currently active or most likely to be accessed.
- *Kasenna MediaBase*, which allows assets stored in StudioCentral to be streamed to end-user desktops.

Database Keys

Database keys allow StudioCentral clients to access metadata that is stored in industry-standard databases. Additional database keys for other database systems can be written. Contact SGI Professional Services for information.

Content Server Keys

Content server keys allow StudioCentral clients to access digital data that is stored in industry-standard devices and storage management systems. Additional content server keys for other storage devices or storage management systems can be written. Contact SGI Professional Services for information.

Plugin Server

The StudioCentral Plugin Server provides developers the infrastructure that allows rapid extensibility of the core StudioCentral functionality. The plugin server provides a common interface for invoking user-defined plugins at specific checkpoints during StudioCentral operations and also manages plugin traffic. Custom, or legacy software can interact with core StudioCentral services by means of plugin server keys written by system integrators and application developers.

Search Server

The StudioCentral Search Server provides a generic infrastructure to support indexing and searching of digital content using search keys. Each third-party search engine would require a separate search key written by system integrators or application developers. For example, a search key could be written to perform searches on digital images.

Asset Typing

Asset Typing automatically generates metadata attributes by analyzing the digital contents of an asset. For example, asset typing of an MPEG file generates values for many attributes including number of tracks, number of frames, and compression type. StudioCentral provides asset typing for a wide range of file types and plugins for other file types can be written.

Thumbnail Generation

The StudioCentral Thumbnail generation feature automatically generates a small low resolution image (a *thumbnail) when an asset with visual* content (images, video) is checked into StudioCentral. For videos, the thumbnail can be generated *from either the first frame of the video* asset or from a user-specified frame number. A plugin to provide custom high-resolution thumbnails or other related feature can be written.

Asset Management Protocol

The Asset Management Protocol (AMP) provides a language-independent, lightweight, protocol that can be used to quickly develop thin client applications. AMP is a TCP-based interface for storing and retrieving the metadata and content of assets stored in StudioCentral repositories and covers the full range of StudioCentral core and asset services. To facilitate communication between StudioCentral and plugin applications, AMP command output is available in XML format.

Asset Services

Asset services, which provide a service to either applications or other services, consist of the following:

- “Transfer Manager Service” on page 28
- “Consistency Checking Service” on page 29
- “Backup and Restore Features” on page 29

Transfer Manager Service

The Transfer Manager performs background tasks that transfer assets between StudioCentral repositories. For example, the Transfer Manager can transfer assets from a centralized StudioCentral server in a broadcast network to other StudioCentral servers at regional stations or affiliates for local storage and playback.

Asset transfer can be performed on a large number of assets which can be simple assets or composite ones. (Composite assets are assets that contain references to other assets.) As with other asset operations, assets may be selected for transfer by name, by ID number, or on the basis of a query.

Consistency Checking Service

Consistency Checking searches for and flags inconsistencies between content location stored in asset metadata, and the location of content files in the content servers. For example, Consistency Checking can identify content files that are not pointed to by assets (orphan content) and assets for which one or more content files are missing.

Backup and Restore Features

Backup and Restore functionality is provided in StudioCentral by two command-line utilities: one that takes a snapshot of the entire metadata and content of a StudioCentral repository, and another that restores that data. The Restore feature retains the database schema and restores the content files for all assets. The Backup and Restore functionality can be used only for content that is stored in a Filesystem Content Server.

StudioCentral Clients

StudioCentral supports the following clients:

- “Macintosh Client” on page 29
- “Windows NT Client” on page 30
- “Web-based Administration Client” on page 30
- “Command Line Client” on page 30

The StudioCentral Development System supports the development of additional clients, see “Developing New StudioCentral Components” on page 31 for details.

Macintosh Client

The StudioCentral Macintosh client allows end users to store, retrieve, query, and manage StudioCentral assets using their native desktop environment. The StudioCentral Macintosh client is integrated with other Macintosh applications, allowing assets to be dragged and dropped between StudioCentral and other applications. The Macintosh clients can be used in a distributed environment in which assets are shared by workgroup members. For a quick overview of the Macintosh client, see “StudioCentral Macintosh Client Scenario” on page 13. For detailed information on the StudioCentral

Macintosh client see the *StudioCentral Library User's Guide for Macintosh Clients* (007-3989-*nnn*).

Windows NT Client

The StudioCentral Windows NT client allows end users to store, retrieve, query, and manage StudioCentral assets using their native desktop environment. The StudioCentral Windows NT client is integrated with other Windows applications, allowing assets to be dragged and dropped between StudioCentral and other applications. The Windows clients can be used in a distributed environment in which assets are shared by workgroup members. For a quick overview of a Windows client, see "StudioCentral Windows NT Client Scenario" on page 8. For detailed information on the StudioCentral Windows NT client see the *StudioCentral Library User's Guide for Microsoft Windows Clients* (007-3621-*nnn*).

Web-based Administration Client

The StudioCentral Administration client is a key component of any StudioCentral installation. This client is web-based to facilitate StudioCentral administration over a WAN and provides a graphical user interface for all essential administration tasks including starting and stopping services, administering users and groups, assigning asset privileges, and transferring assets. For detailed information on the StudioCentral Administration client see the *StudioCentral Library Administrator's Guide* (007-3623-*nnn*).

Command Line Client

The StudioCentral command-line client is available on the StudioCentral server and can be installed on any other IRIX host. The command-line utilities are executable, ready-to-use programs that perform a full range of StudioCentral tasks. The utilities are based on AMP and can be used individually, in scripts, or in the creation of a client application. The command-line client is used most frequently in development environments and for batch processing involving a large number of assets or StudioCentral operations.

Developing New StudioCentral Components

StudioCentral provides interfaces for building applications and services that use StudioCentral, or for integrating existing applications into StudioCentral. Detailed information about how to use these interfaces is provided in the *StudioCentral Library Developer's Guide*. This section provides an overview of how to use the interfaces to develop the following components:

- "Client Applications" on page 31
- "Writing Plugins" on page 32
- "Writing Search Keys" on page 32

Client Applications

The tasks required to perform asset management in any client application are generally the same:

- Create and store asset contents and metadata.
- Retrieve asset content and metadata.
- Modify information describing the asset content.
- Delete asset content and its metadata.

Each of the interfaces described in this section enable you to create client applications that accomplish these tasks. Detailed information about how to use these interfaces to develop applications is provided in the *StudioCentral Library Developer's Guide*.

The interfaces for developing client applications are:

- "AMP" on page 31
- "C++" on page 32
- "Command-Line Utilities" on page 32

AMP

The Asset Management Protocol (AMP) gives developers a direct means of managing and querying assets. AMP is a rich command set that covers the full range of StudioCentral core and asset services, and can be incorporated into Perl scripts, java, C++, or applications in most any other computer language to create a client application. To facilitate communication between StudioCentral and Client applications, AMP command output is available in XML format.

C++

The C++ language is used to develop StudioCentral plugins and search keys for content-based search.

Command-Line Utilities

The StudioCentral command-line utilities are executable, ready-to-use programs that are based on AMP commands and perform a variety of operations on assets. You can use these utilities in asset operations, and to perform administrative tasks. They can be used from the command-line or grouped together in scripts.

Writing Plugins

StudioCentral Plugin libraries are written in C++ and communicate with StudioCentral through the Plugin Server process which manages traffic between the plugins and the Ams Server. For details on writing plugins, see the *StudioCentral Developer's Guide* (007-3246-*nnn*).

Writing Search Keys

StudioCentral content search keys are written in C++ and communicate with StudioCentral through the Search Server process which indexes content files and manages traffic between the search engine and the Ams Server. For details on writing a search key, see the *StudioCentral Developer's Guide* (007-3246-*nnn*).

Configurations

This chapter describes the StudioCentral Library configurations and the tasks required to install, configure, and support them:

- “Configuration Types”
- “StudioCentral Installation and Administration Tasks” on page 38

Configuration Types

StudioCentral can be used in a distributed configuration, where many users store assets in centralized repositories. This lets multiple users work on assets under the version control and security of StudioCentral.

This section describes the following StudioCentral configurations:

- “Single-Server Configurations” on page 33
- “Multiple-Server Configurations” on page 34

Single-Server Configurations

In the single-server configuration, which is shown in Figure 4-1, an SGI server running the IRIX operating system is the host machine for StudioCentral. The StudioCentral repository and other core software, the third-party database, and the filesystem content server(s) are all located on a single server. This server is part of a network that can include Windows and/or Macintosh computers, other servers, and Web clients. Assets that are stored in StudioCentral can be accessed by users on the network.

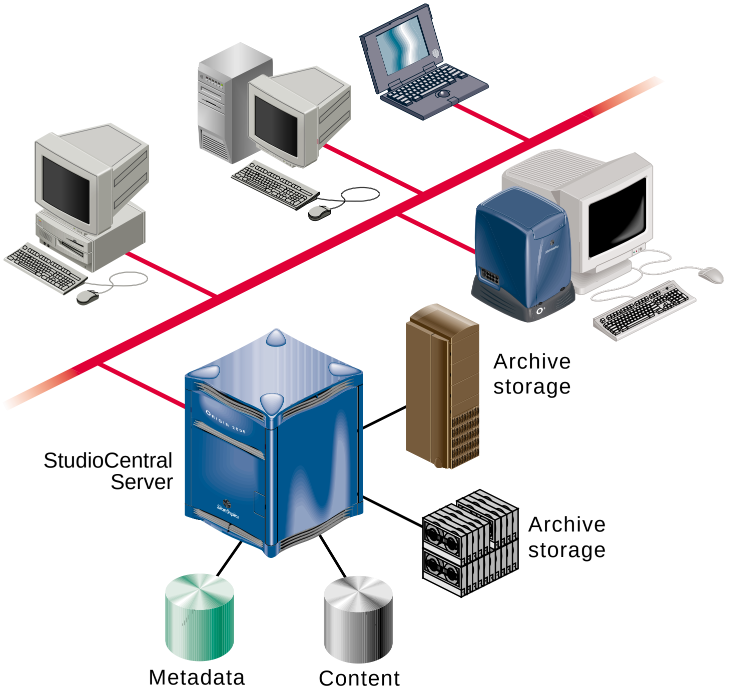


Figure 4-1 Single-Server Configuration

The StudioCentral Configuration Tool, which is part of the StudioCentral Admin Tool, is used to install and configure the StudioCentral configuration shown in Figure 4-1. It provides an easy-to-use web-client interface. Using the tool to install and configuring a basic, single-server configuration is described in the *StudioCentral Library Installation and Configuration Guide* (007-3319-*nnn*).

Multiple-Server Configurations

StudioCentral has the ability to run in a fully distributed environment on multiple server systems. The most common multi-server configurations are:

- StudioCentral with a Remote Content Server
- StudioCentral with a Remote Database
- StudioCentral with a Remote Content Server and Database

Each of these configurations is described in this section.

Installation and configuration of a multiple-server configuration requires a knowledgeable system administrator or integrator, as some manual configuration of StudioCentral services is required. (The *StudioCentral Library Installation and Configuration Guide* describes how to do this.)

StudioCentral Configuration with a Remote Content Server

In this configuration, shown in Figure 4-2, an SGI server running the IRIX operating system is the host machine for StudioCentral. The StudioCentral server software, the third-party database, and a filesystem content server are all located on a single server. Due to storage requirements, a second content server is located on another IRIX host machine connected to the same network. Assets that are stored in either StudioCentral content server can be accessed by users on the network. An example of a remote content server is a MediaBase content store.

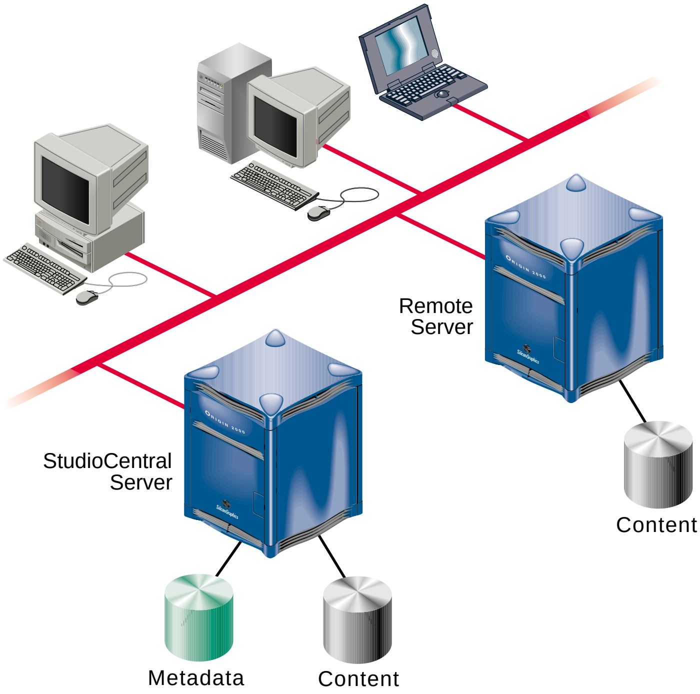


Figure 4-2 Configuration with a Remote Content Server

Whether the remote content server is a Kasenna MediaBase content server, a content server for a third-party archival storage system, another filesystem content server, or any other content server, it is customary to configure StudioCentral to have a local filesystem content server as well as a remote content server. This allows the administrator to configure a basic StudioCentral configuration (using the web-based Configuration tool) and then to add an additional content server manually on the remote machine.

StudioCentral Configuration with a Remote Database

In this configuration, shown in Figure 4-3, an SGI server running the IRIX operating system is the host machine for StudioCentral. The StudioCentral server software, and one or more filesystem content server(s) are located on a single server. Due to storage requirements, the third-party database is located on another IRIX host machine connected to the same network.

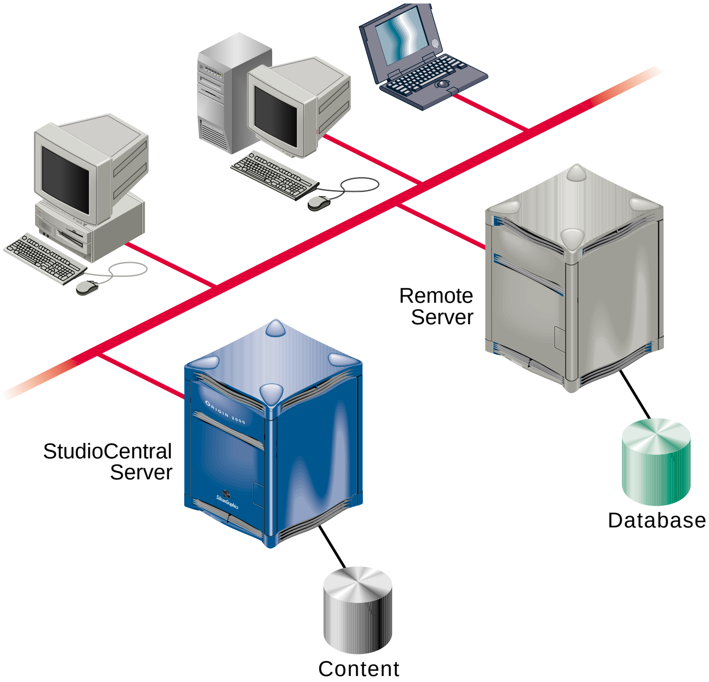


Figure 4-3 Configuration with a Remote Database

StudioCentral Configuration with a Remote Content Server and Database

In this configuration, shown in Figure 4-4, an SGI server running the IRIX operating system is the host machine for StudioCentral. The StudioCentral server software, and one or more filesystem content server(s) are located on a single server. The third-party database is located on another IRIX host machine, where a remote content server is also located. Both servers are part of a network that can include Windows and/or Macintosh computers, other servers, and Web clients.

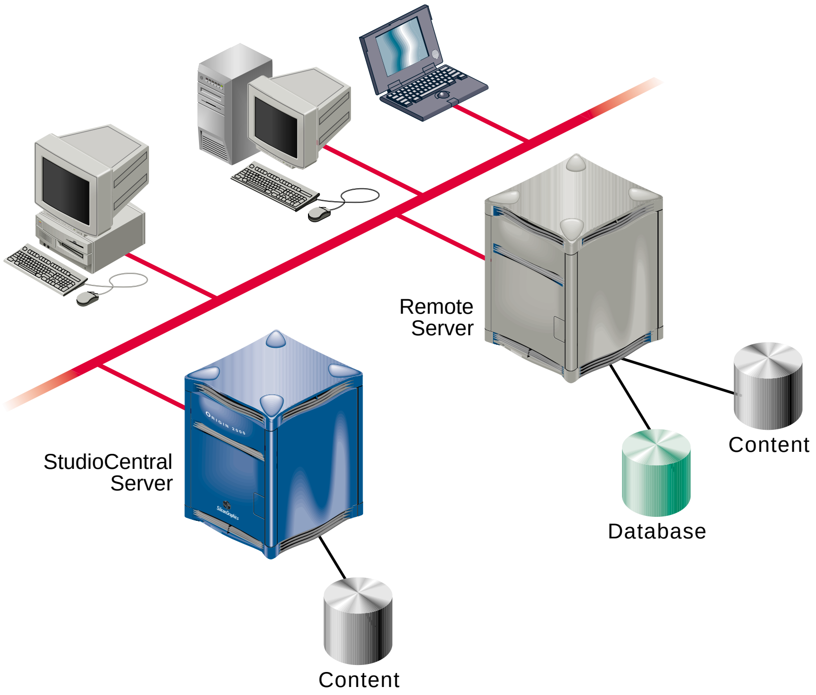


Figure 4-4 Configuration with a Remote Database and a Remote Content Server

StudioCentral Installation and Administration Tasks

This section lists the procedures required to install, configure, and administer StudioCentral. The section is organized by administrative function. In some environments, one individual may be responsible for all functions; in others, more than one individual may be responsible. When more than one individual is involved, it is important to coordinate the administrative tasks for the successful installation and operation of StudioCentral.

StudioCentral Administrator

The StudioCentral administrator is responsible for the overall installation and operation of the StudioCentral servers and clients. This individual performs the following tasks:

- installs, configures, and operates the StudioCentral servers (with help from the database administrator)
- installs and configures the StudioCentral clients (with help from the database administrator)
- creates and manages StudioCentral users and their security privileges
- creates IRIX users
- creates database users (with help from the database administrator)
- updates and maintains content server configuration files
- performs queries and data management on assets
- scans log files to troubleshoot problems
- starts and stops the StudioCentral services
- adds and manages StudioCentral datamodels

Database Administrator

The database administrator is responsible for the database management system that is used with StudioCentral. This individual is also responsible for helping the system administrator during the installation and configuration of StudioCentral. The following tasks are performed by the database administrator:

- helps install, configure, and maintain the Informix or Oracle database management system used by StudioCentral
- defines the initial database configuration
- uses a StudioCentral script to create the StudioCentral schema file
- manages database user IDs for the database(s) used in StudioCentral
- modifies and tunes the database for performance and reliability
- establishes and performs database backup and maintenance procedures for failure recovery of data

Glossary

Access Control

The AMP functionality that manages access to StudioCentral assets. StudioCentral provides two types of access control: repository-level and asset-level. For example, Access Control gives end users permission to access assets created by an administrator. See also *ACL*.

ACL

ACL stands for Access Control List. Many of the StudioCentral command-line utilities and AMP commands involved with repository and asset access control have the letters "ACL" as part of their name. See *access control*.

Ams Server

The StudioCentral core server process that performs and manages all database and content store operations, communicates with the clients, and communicates with the other StudioCentral server processes (Transfer Manager, Plugin Server, Search Server, Purge Daemon, Consistency Checker). A StudioCentral server includes a single Ams Server, but may be configured to have one or multiple repositories, each of which can have one or multiple content servers.

amsmgr

The "root" StudioCentral user. The user amsmgr is the admin-level StudioCentral user who performs configurations, is the owner of database tables, and is a member of all StudioCentral groups. amsmgr must exist as a database user before StudioCentral can be configured. Other users may be given StudioCentral admin-level privileges, but certain tasks can only be performed by amsmgr.

asset

Unit of storage in a *digital asset management system*. Each asset consists of descriptive information such as the title and duration; digital media data, if the asset has content; and an index, if required by the content format. For example, each movie, commercial, trailer, thumbnail, and so on, stored in an archive system is an asset. Assets may consist of only descriptive information if those assets are used to group other assets.

Asset Query Language (AQL)

Provides a way to search for assets using metadata attributes. The syntax for an AQL query resembles that of the database structured query language (SQL) select statement, except that AQL queries specify Datamodels and attributes instead of database tables and columns. See also *attribute* and *Datamodel*.

Asset Management Protocol (AMP)

A language-independent, lightweight, protocol that can be used to quickly develop thin client applications. AMP is a TCP-based interface for storing and retrieving the metadata and content of assets stored in StudioCentral repositories and covers the full range of StudioCentral core and asset services. To facilitate communication between StudioCentral and plugin applications, AMP command output is available in XML format.

asset typing

The process by which an asset's type is determined, its content file is read, and the following are automatically generated: metadata attributes based on the asset type and, if appropriate, a *thumbnail*. For example, for a video asset, the duration, bit rate, and size of its image are extracted from the content file and automatically stored in the asset's metadata when it is checked into StudioCentral. Asset typing is part of cataloging.

attribute

A single piece of an asset's metadata information. Metadata consists of system attributes, which are defined by StudioCentral and exist in all assets, and Datamodel attributes, which extend the attributes associated with an asset. For example, there is a system attribute that contains an asset's name. Attributes are stored in the database in default or explicit mapping tables. See also *Datamodel, default mapping, explicit mapping*.

cataloging

The process by which the information for an asset is determined and stored with the asset as metadata attributes. Cataloging occurs when an asset is first checked in or an existing asset is updated. Asset typing is part of cataloging at initial checkin of an asset.

check in

To add a new asset to StudioCentral, you must check it in. At checkin time, StudioCentral automatically assigns the asset a unique identifier (*OID*) in order to locate it in the repository, and secure it in the database. Assets may be created in StudioCentral as versioned or non-versioned assets. For all assets, each time you check it back in, StudioCentral increments the version number. In the StudioCentral desktop clients, assets are reserved (locked) during asset cataloging and check in. See also *check out*, *repository, retrieve*, *version*, *zeroth version*.

check out

To modify an asset, you must either retrieve or check it out of the StudioCentral repository. Both retrieve and check out give you a “working copy” of the asset. Check out also locks the “real” file to prohibit changes by other users until you check in your modifications. Locking the real file is called “making a reservation.” Check out is available to StudioCentral developers (with AMP commands), but the StudioCentral desktop clients do not implement check out. Instead they implement asset locking during check in. See also *retrieve*.

composite asset

An asset that points to, or contains, other assets, some of which may also be composite assets. Composite assets are stored in StudioCentral using references to point to related assets.

Consistency Checker

A StudioCentral tool that checks the data integrity of a repository and its associated content servers.

content

The data portion of an asset (for example, the image, scene, sound sequence, or song). StudioCentral stores an asset’s content separately from its metadata. See also *asset*, *metadata*.

content server key

The software component that enables a content store to be used with StudioCentral. For example, the key for the Filesystem Content Server enables content files to be stored in XFS filesystems.

database key

The software component that enables a database management system to be used with StudioCentral. For example, StudioCentral has a database key for the Informix Online Dynamic Server.

Datamodel

A data structure that defines metadata attributes; it is used for modeling data. An asset can have more than one Datamodel associated with it. If you want to extend the metadata attributes of an asset, you can create a new Datamodel for it.

Datamodel template

StudioCentral's "stylesheet" for a Datamodel. It contains the definitions of all attributes within that Datamodel.

default mapping

The standard method of database storage for StudioCentral attributes. Datamodel attributes are stored in (mapped to) database tables based on the attribute data type (string, float, char, etc.). The default mapping tables can get quite large and cannot provide optimized query performance. To optimize query performance, use explicit mapping tables instead. See also *explicit mapping*.

developer

Software engineer who writes language-independent plugins or applications for StudioCentral using the AMP protocol. See also *end user*.

digital asset management system

A system for storing, managing, and distributing digital multimedia data.

direct mapping tables

See *explicit mapping*.

end user

End user of StudioCentral application programs.

explicit mapping

A method of database structuring to enhance StudioCentral query performance. Datamodel attributes that are frequently used in queries are stored in (mapped to) user-specified database columns and tables. By default, StudioCentral Datamodel attributes are not explicitly mapped, but are stored in default mapping tables based on the attribute data type (string, float, char, etc.). The use and management of database tables and columns to facilitate better StudioCentral performance for queries and other tasks is called database tuning. See *default mapping*.

file typing

See, *asset typing*.

Kasenna MediaBase

Media serving software for delivering high-quality MPEG-1, MPEG-2, and H.263 video and audio streams to Web clients using IP or native ATM networks.

metadata

A series of attributes that describe an asset. Metadata is stored in a database and is used to search for (or query) assets that meet specified criteria. For example, if you have an asset that contains one scene in a movie and its metadata specifies that the scene uses special sound effects, you find this asset when you query the database for all movie scenes that use special sound effects. To speed queries, StudioCentral stores metadata separately from content. See also *asset*, *attribute*, *content*, *query*.

multiformat

A type of asset that groups other assets, each of which has the same digital content but is encoded in a different digital format.

non-versioned

When you first check in an asset, you can check it in as a non-versioned, or unversioned, asset rather than a *versioned* asset. StudioCentral stores only the latest version of a non-versioned asset. At each checkin, it overwrites the previous version and increments the version number. You only have the choice between versioned and non-versioned when you first check in an asset. See also *version*, *zeroth version*.

OID

The unique identifier that StudioCentral assigns to each version of an asset when you check it in. OIDs comprise three numbers: the repository ID, a number that uniquely identifies each version of an asset, and a number that uniquely identifies each version tree. For all assets in a given repository, the first portion of the OID is identical. OIDs define asset structures in which one asset points to, or contains, another asset. See also *check in*, *version*, *version tree*.

persistent

Protected from removal from the repository.

pinning assets

Preventing assets from being deleted. Pinning is used primarily in asset delivery applications. Pinning is performed on a particular version of an asset and does not prevent assets from being retrieved or new versions being checked in.Pinned assets must be unpinned before they can be deleted.

Plugin Server

An optional StudioCentral server process that allows the integration of, and manages traffic at known plugin stages between custom-written plugins and the Ams Server. A plugin stage is a fixed checkpoint (such as pre-write) at which a plugin can be invoked.

Policy Module

A StudioCentral feature used when a repository has more than one content server configured. The Policy Module uses the attribute values specified in the `cmgrMap` file to manage the location where content files are stored when assets are checked in. All of the content files for a particular asset are stored on the same content server.

preview

A smaller version of an asset's content. For example, the preview may be a short segment or a lower-resolution version of the asset.

query

The process whereby users search a StudioCentral repository for assets that satisfy given criteria. For example, a user might search for all video assets that contain footage of tornadoes. Users perform queries by specifying values for metadata attributes. Content search capability can be integrated into StudioCentral by writing a search key for a third-party search engine. The search key would then communicate with StudioCentral through the StudioCentral Search server.

queryable

A property of an *attribute* that lets it be used as a condition when searching for, or querying, assets. For example, if ***owner_name*** is a queryable attribute, you can search for assets that have an ***owner_name*** attribute whose value is "Smith." See also *query*.

query mapping tables

See *explicit mapping*.

reference

A metadata attribute that points to another asset. Composite assets include references.

repository

A storage unit for digital assets. Each repository consists of one database, for storing metadata, and one or more content servers, for storing the content. A StudioCentral server may include multiple repositories in either a single-server or multi-server configuration. The multiple repositories are managed by a single Ams Server process.

See also *asset*.

reserve

To lock an asset, so that only the specified user can check in the next version, you make a reservation or reserve the next version of the asset. When you reserve an asset, StudioCentral puts a lock on the next version of the asset. Other users are locked out from checking in the asset. For the StudioCentral desktop clients, assets are not reserved when they are retrieved; assets are only reserved during the time a user is cataloging or checking in the asset. See also *check out*.

retrieve

To modify an asset, you can retrieve it from or check it out of the StudioCentral repository. When you retrieve an asset, StudioCentral gives you a "working copy" of the asset. Other users are locked out from retrieving the asset only during the time you are cataloging or checking in the asset. See also *check out, reserve*.

root

The first version of a versioned asset.

Search Server

An optional StudioCentral server process that allows the integration of content search with the Ams Server by means of a third-party search engine and a custom-written StudioCentral content search key.

StudioCentral Server

A general term for the server-side software of a particular StudioCentral installation. A StudioCentral Server is comprised of a single Ams Server process, and one or more repositories containing assets. In a distributed environment, certain StudioCentral server processes may be run on distinct host machines (for example a remote content server). The database server may also run on a remote machine. From a hardware perspective, the term StudioCentral server is used to designate the SGI server where the Ams Server process runs.

StudioCentral system

A complete StudioCentral client-server configuration comprised of a single Ams server process and all of the associated repositories, processes and clients. A large StudioCentral system typically is installed using several servers (host machines) and many clients in a heterogeneous distributed environment. See also *StudioCentral server.*

thumbnail

A pictorial representation of the asset. For example, the thumbnail for a movie asset might be a still from the actual movie.

Transfer Manager

An optional StudioCentral server process that performs and manages the transfer of simple or complex assets from one StudioCentral system (or repository) to another.

trunk

All the versions that make up a versioned asset.

version

A particular stored instance of an asset. Most StudioCentral assets are created as versioned assets. This means that each time you modify the asset and check it in, a new version of the asset is given a unique OID and stored. See also *check in*, *check out*, *versioned*, *non-versioned*.

version tree

A schema for identifying all the versions of a particular versioned asset. As users continue to modify an asset (checking it in and out), StudioCentral "grows" its asset tree by creating a new version each time a changed version is checked in. Each version tree consists of a root and a trunk. See also *root*, *trunk*.

versioned

When you first check in an asset, by default you check it in as a versioned asset. This means that StudioCentral will track all subsequent revisions (*version*s) of the asset, storing each one as a separate asset with a unique OID. You only have the choice between versioned and non-versioned when you first check in an asset. See also *version*, *zeroth version*, *non-versioned*.

zeroth version

The latest version of an asset. See also *version*, *version tree*, *trunk*.

Index