

Oracle Content Management SDK: Concepts and Architecture

*An Oracle Technical White Paper
August 2005*

Oracle Content Management SDK: Concepts and Architecture

Overview.....	3
Oracle's Integrated Content Management Platform.....	4
Architecture Overview	5
Component Overview.....	5
Oracle CM SDK Domain.....	6
The Content Repository.....	6
Domain Controller.....	7
Nodes	7
Node Guardian.....	7
Node Manager	8
Services	8
Sessions.....	9
Servers	9
Protocol Servers	9
Agents	10
Managing Your Oracle CM SDK Instance	10
Oracle Enterprise Manager 10g Grid Control	10
Oracle Enterprise Manager 10g Application Server Control.....	10
Oracle CM SDK Manager.....	10
Other Management Utilities.....	11
Building Content Management Applications in Java	11
Connecting to the Repository	12
Creating Folders and Documents.....	13
Finding Your Content	14
Selector Search.....	15
Advanced Searches.....	15
Content-Based Searches using Oracle Text.....	15
Versioning Your Content.....	16
Securing Your Content.....	17
Authentication	17
Authorization.....	17
Driving your Content Processes with Oracle Workflow.....	18
Conclusion.....	18

Oracle Content Management SDK: Concepts and Architecture

Oracle Content Management SDK provides the run-time environment and tools to rapidly create applications that consolidate and manage your content in an Oracle database. It leverages the Oracle Application Server platform to provide the needed scalability, reliability and performance. This white paper examines the architecture and concepts of Oracle CM SDK.

OVERVIEW

Many companies have a need to provide content based services to their employees. For example, a company could provide a central document repository with collaboration capabilities to encourage document reuse for more efficient operation.

Alternatively, a company could build a custom application to handle various forms related to the operation of the business, such as a competitive car parts application for documenting all of the features of competitors' cars, or an application for handling insurance claims.

These applications have similar requirements: they all need to be developed rapidly, with a very high level of confidence that the project will be successful. The requirements are such that the application must provide the performance and reliability that the company needs to run mission-critical tasks. The projects must be based on a proven platform with best practice architecture, and they must use languages and tools that a large number of developers are familiar with.

Oracle CM SDK fulfills all of these requirements. By leveraging Oracle's proven technology, it provides the platform and tools to:

- More rapidly develop content-based applications with less risk than starting from scratch.
- Consolidate many file servers into one repository, potentially saving you money as the need for extra hardware and distributed management disappears.
- Provide uniform administration, security, and access control policies to ensure that the content is safe, accessible, and reusable. This is becoming even more important as regulations such as the Sarbanes-Oxley Act of 2002 and other

SEC regulations mandate various policies on the management and retention of data.

- Automate the business processes around your content to ensure efficient execution of your business tasks and improve employee productivity.

ORACLE'S INTEGRATED CONTENT MANAGEMENT PLATFORM

Oracle CM SDK provides a single point of integration for many of Oracle's products for content management. This makes it much easier to develop enterprise content management solutions since everything is now accessible from the Java APIs provided by Oracle CM SDK.

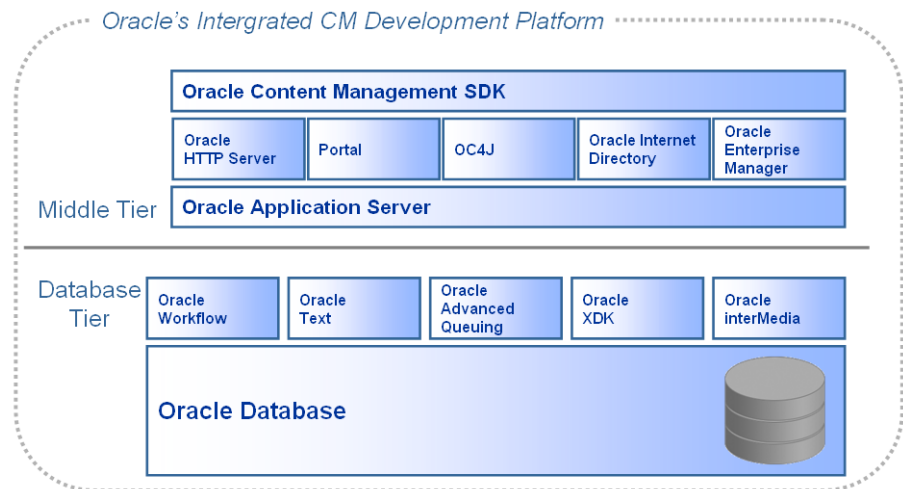


Figure 1: Oracle CM SDK provides a central point of integration for many of Oracle's content management products.

Oracle CM SDK is integrated with the following database products:

- Oracle Workflow for managing business processes around your content.
- Oracle Text for content searches.
- Oracle *interMedia* for rich media annotation.
- Oracle Advanced Queuing for integrating to other applications.

Oracle CM SDK is also integrated with the following middle-tier products that form part of the Oracle Application Server platform, including:

- Oracle Enterprise Manager 10g Application Server Control
- Oracle Internet Directory for LDAP-based directory services and Single Sign-On capability

- OracleAS Portal

To further aid to rapid development of your content management applications, you can easily build these applications with Oracle JDeveloper, an award winning integrated development environment.

ARCHITECTURE OVERVIEW

Oracle CM SDK is designed with a three-tier approach to provide excellent performance, scalability, and reliability. The database is used extensively to store the content and metadata, taking advantage of its benefits and features to ensure that the content and associated metadata is safe, secure, and easily retrievable.

The protocol servers and application business logic is provided by the middle tier. This enables Oracle CM SDK to provide better scalability by allowing you to add middle-tier machines to cope with increased demand. The multiple middle-tier machines can be load balanced for more effective handling of loads and provide redundancy in the event that one machine fails. The following sections briefly discuss the components of Oracle CM SDK and demonstrate how the product features can be used to build the content-based application that meets your organization's needs.

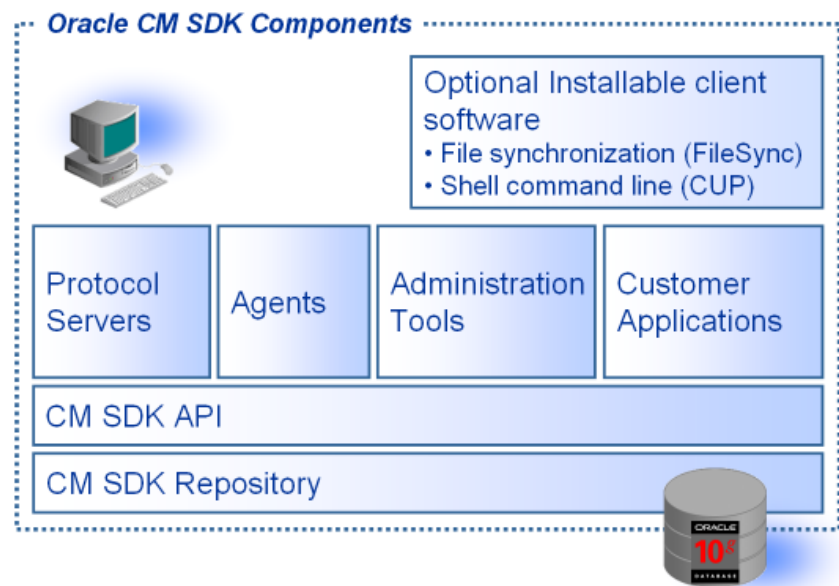


Figure 2: Oracle CM SDK Components

Component Overview

Oracle CM SDK comprises the following components:

- The repository, consisting of the Oracle Database and the objects for managing the persistence of the content.

- A 100% Java API for developing content-centric applications.
- Servers, which provide Agent-based background processing and protocol access.
- Administrative tools for managing your instance of Oracle CM SDK.
- Optional client software for synchronizing the content between your local machine and the content repository.

Oracle CM SDK Domain

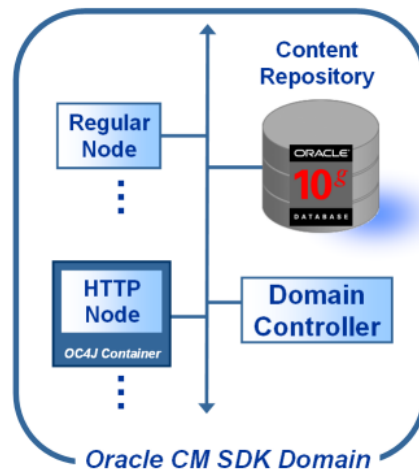


Figure 3: The Oracle CM SDK domain is initially configured with one HTTP node and one regular node

When discussing the Oracle Content Management SDK we use the term *domain* to mean one Oracle CM SDK instance.

The domain consists of the domain controller, regular and HTTP nodes, and the Oracle CM SDK repository.

The Content Repository

Oracle Content Management SDK stores all of the content and associated metadata in an Oracle database. When configuring Oracle CM SDK, users create a single Oracle CM SDK schema containing several data types:

- The primary data type for storing all of the information about the content itself
- The indexed and non-indexed media
- Oracle *interMedia* media for rich media annotations
- Oracle Text keymap, index, and data for content search information

The data types can be stored in different tablespaces for improved performance. There are several other schemas created, but these are internal to the application.

Domain Controller

The domain controller is responsible for starting and stopping the domain and subsequently monitors the domain to ensure that it is running properly.

Nodes

Nodes handle client requests, perform the requested operation and respond back to the client.

There are two types of nodes:

- *Regular Nodes* run in standalone Java Virtual Machines (JVMs).
- *HTTP Nodes* run in OC4J instances and can run JSP, and Servlets.

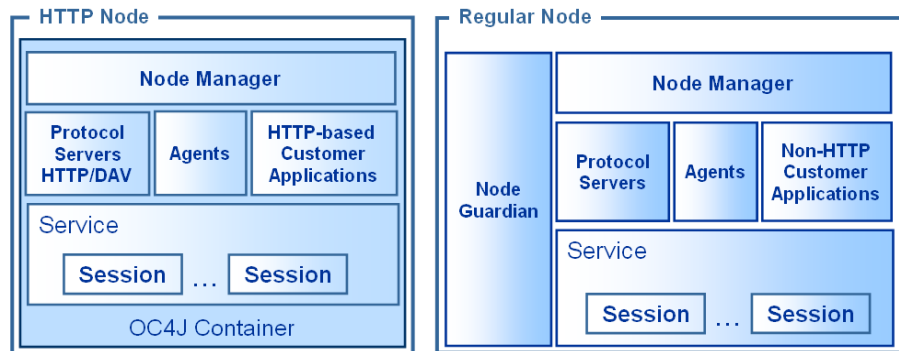


Figure 4: The two kinds of Oracle CM SDK nodes and their components

Nodes can be located all on one host, or spread across a set of hosts for greater system scalability.

Within a node there are several components:

- The node guardian
- The node manager
- Protocol servers and agents
- Services
- Sessions

Node Guardian

The node guardian is a process with a dual purpose; to start the node by creating a new node manager process, and then to monitor the node in case of failure. If the node fails, the guardian will restart the node manager, effectively restarting the node.

In HTTP nodes, the node guardian runs in the same process as the other components (along with the node manager) and its monitoring capabilities are somewhat diminished. The monitoring of the node is therefore mostly provided by the Oracle Application Server Container for J2EE (OC4J) process.

Node Manager

The node manager is the process responsible for starting the appropriate services and servers, and for setting up node logging. It works in conjunction with the node guardian to provide the environment in which the services and servers operate.

Services

Services provide “windows” to the repository. Users establish a connection to Oracle CM SDK by authenticating against a service. The service will create and return a session for that user that provides a connection context.

Services also provide caching of objects to improve performance. Each service has a “data cache” that holds the values of objects’ attributes. The cache is shared across the sessions of that service.

Each service also has an Access Control List (ACL) resolution cache. ACLs are resolved in-memory to a table of “user or group X has permission to do Y.” Services also contain mechanisms for maintaining cache synchronization with other nodes. This is done through the exchanging of events via the database.

Services also manage the connections to the database. A service provides two connection pools; one read-only connection pool and one writeable connection pool.

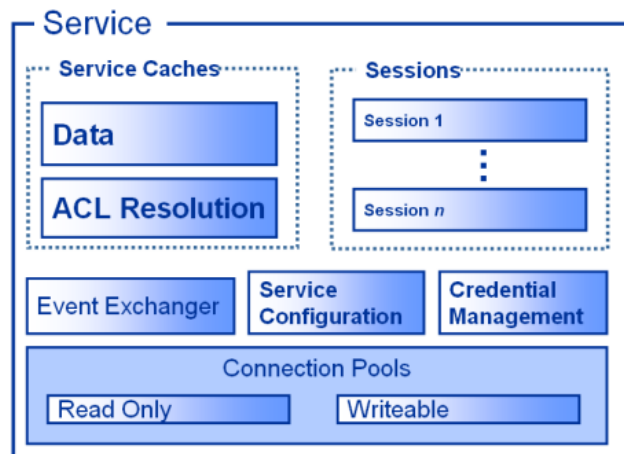


Figure 5: Oracle CM SDK services support the connection of applications to the database and caches content for improved performance.

Sessions

Sessions provide a mapping between users and the repository. From an API point of view, most API method calls require a valid session object. Sessions cache objects which have not been committed to the database.

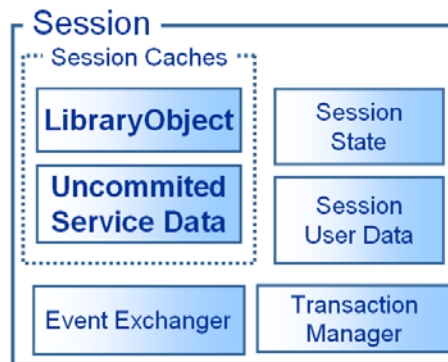


Figure 6: The session caches the content not yet committed to the database and provides the mapping between specific users and the repository.

Sessions also provide caching. Each session has an “object cache” that holds the Java flyweights – lightweight Java objects that represent objects in the services data cache. This is how sessions can share cached objects. Some flyweights also hold session-specific state.

Each session also has an “uncommitted data cache” that holds the attribute values of objects changed within the session’s current transaction. The cache contents are made available to the other sessions through the Service data cache once the transaction is committed.

Servers

Much of the work of Oracle CM SDK is provided by servers. These are objects that run in the node manager’s process. Servers follow a pattern that allows them to run in, and be managed by, a node.

There are two basic types of servers in Oracle CM SDK: protocol servers and agents.

Protocol Servers

Protocol servers provide out-of-the-box access to the repository via industry standard file system protocols like SMB/NTFS, AFP, or NFS. There are also protocol servers for Internet-based access to the repository, namely FTP, HTTP, and WebDAV. Additionally you can connect to the repository by the standard email protocols IMAP and SMTP.

These protocol servers allow you to use your existing productivity tools, such as word processors, spreadsheet programs, your favorite email clients or a Web-based user interface, to store and manage your content in the Oracle CM SDK repository.

Agents

Another type of server in an Oracle CM SDK node, called an *Agent*, provides asynchronous (background) processing capabilities. Agents can be triggered based on time or by an event. Many management tasks are performed by Agents to improve usability and to provide housekeeping services to keep the repository running efficiently. For example, there are Agents for garbage collection, for aging content, for provisioning users from Oracle Internet Directory to Oracle CM SDK, and for pre-initializing service caches so that they start up more quickly.

MANAGING YOUR ORACLE CM SDK INSTANCE

Oracle CM SDK provides several levels of management, all designed to ensure that your system runs effectively and efficiently.

Oracle Enterprise Manager 10g Grid Control

The Oracle Enterprise Manager 10g Grid Control is a Web-based tool that provides centralized management for multiple middle tiers, including Oracle Application Server middle tiers, OracleAS Infrastructure tiers, and Oracle Database Server hosts.

Oracle Enterprise Manager 10g Application Server Control

Oracle CM SDK is fully integrated with the Oracle Enterprise Manager 10g Application Server Control, providing extensive management of Oracle CM SDK processes. Some of the management tasks that can be performed from the Application Server Control include:

- Starting and stopping Oracle CM SDK processes from a single interface and editing the configuration information for Oracle CM SDK nodes, servers, and services.
- Automatically monitoring status, usage and performance data from the Oracle CM SDK processes and view aggregate performance.
- Viewing network ports and system log files from a central location.

Oracle CM SDK Manager

Oracle CM SDK Manager is a Web-based administration tool that enables management of the repository. Features include the ability to:

- Manage Access Control Lists and permissions.

- Manage users and groups.
- Manage mount points, printers, and queues.
- Extend the properties of Oracle CM SDK by extending existing class objects or creating new ones, including the ability to create new categories for custom metadata.

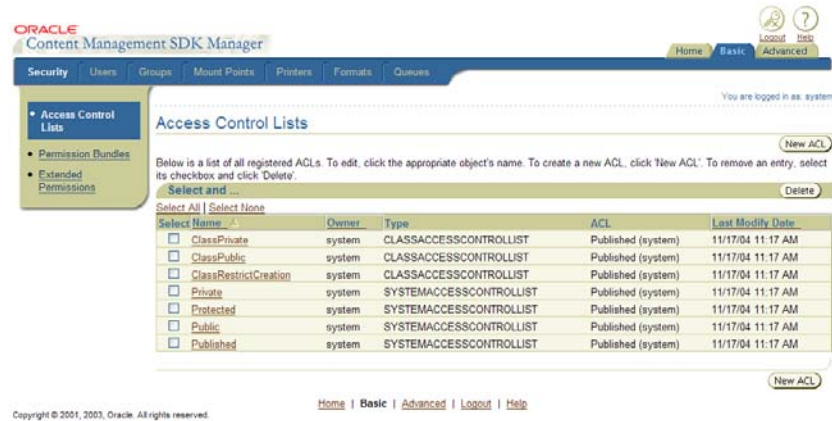


Figure 7: Oracle CM SDK Manager provides Web-based management and the ability to extend the functionality of Oracle CM SDK by extending existing classes or creating custom class objects and attributes.

Other Management Utilities

The Oracle CM SDK Object Inspector enables management of repository objects through a simple and intuitive graphical application that allows you to:

- Display the folder hierarchy and apply an ACL to an object.
- Display the properties, categories, and relationships of selected objects.
- Search for objects using attributes, relationships, and categories associated with an object.

Several configuration scripts and wizards, as well as a Command-line Utilities Protocol (CUP) server, are also provided with Oracle CM SDK to facilitate administration tasks.

BUILDING CONTENT MANAGEMENT APPLICATIONS IN JAVA

The Oracle CM SDK Java API supports all of the features and functionality required to build sophisticated content management applications such as

- Folders
- Security and access control

- Check-in/Check-out and document locking
- Versioning
- Searching
- Extensible metadata
- Other standard content management operations

The API supports an object-oriented approach to developing content management applications and, coupled with the repository, provides object-to-relational mapping of the data. Developers build their applications in Java and the objects are made persistent by Oracle CM SDK without developers having to develop in SQL or be aware of the underlying table structures in the database.

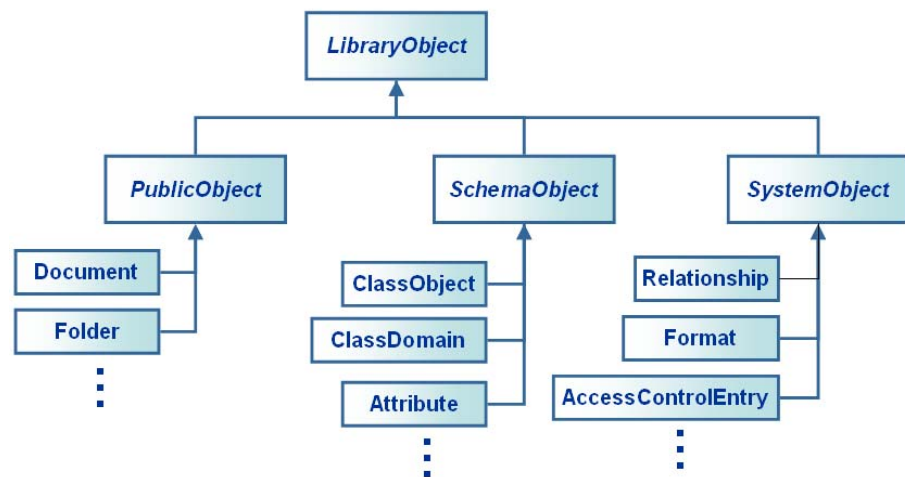


Figure 8: A portion of the Oracle CM SDK Java APIs class hierarchy showing the main abstract class families from which all other major classes are derived.

There are several ways that developers can extend the functionality of Oracle CM SDK. They can:

- Create custom categories for defining arbitrary extensible metadata to better categorize and manage their content
- Write custom Agents
- Override functionality at certain pre-defined points in existing servers
- Subclass existing objects to extend their capabilities

Connecting to the Repository

As an example of the Oracle CM SDK API, we briefly look at connecting to the repository, creating a folder, and then uploading a document into that folder.

We make the assumption that the folder will be created in the home folder of the user connecting to the repository.

This is a simple example of directly connecting to the repository. Typically an application would register with a node by following the Server pattern and then associating that server with the node. This highlights the difference between managed servers and standalone application code.

First, we need to start a service. Services are described earlier in this document.

In Oracle CM SDK, you start a service as shown by the following code sample:

```
// create LibraryService
service = LibraryService.startService(
    // provide a name for this service
    serviceName,

    // The schema password was chosen during configuration of the
    // CM SDK
    schemaPassword,

    // Choose a configuration based on your requirements -
    // typically SmallServiceConfiguration.
    // MediumServiceConfiguration or LargeServiceConfiguration
    serviceConfiguration,

    // The domain name for this CM SDK domain
    domain
);
```

Once a service has been started, you connect to the service to obtain a session. This is the session that is used during subsequent API calls and that identifies this user to the repository (see document and folder creation samples).

```
// Generate a Credential, in this case use a simple text one
CleartextCredential credential =
    new CleartextCredential(username, password);

// Establish LibrarySession by supplying credential to LibraryService
session = service.connect(credential, null);
```

Once we have a session, we can proceed to create other objects in the repository.

When you have finished with a session you disconnect by using

```
session.disconnect();
```

Creating Folders and Documents

Creating documents and folders with Oracle CM SDK is very straightforward. The next set of examples show that you create a definition object for the item that you want created in the repository. You specify in this definition all of the attributes for the item and then finally create the item in the database. This ensures that there is only one round-trip to the database to update the folder or document.

Sample Code 1: Creating a folder in Oracle CM SDK is done by creating a folder definition and then creating the folder object in the database.

```
public Folder createFolder(LibrarySession session, String folderName)
throws IfsException {
    Folder userHomeFolder =
        session.getUser().getPrimaryUserProfile().getHomeFolder();

    // Create a new folder definition
    FolderDefinition folderDefinition =
```

```

    new FolderDefinition(session);

    // Give this folder a name
    folderDefinition.setAttribute(PublicObject.NAME_ATTRIBUTE,
        AttributeValue.newAttributeValue(folderName));

    // Provide a description for this folder
    folderDefinition.setAttribute(
        PublicObject.DESCRPTION_ATTRIBUTE,
        AttributeValue.newAttributeValue("Demonstrating creating
        folders"));

    // Place this folder in the Home folder of the user
    folderDefinition.setAddToFolderOption(userHomeFolder);

    // Finally create the actual folder object in the database
    Folder actualFolder =
        (Folder)session.createPublicObject(folderDefinition);

    // Return a reference to the newly created folder
    return actualFolder;
}

```

Once the folder has been created, you can upload a document directly into that folder as demonstrated by the next code sample.

Sample Code 2: Creating a document is similar to creating a folder. Create the document definition and then create the actual document object in the database

```

public void createDocument(LibrarySession session, Folder aFolder,
    String fileName) throws IfsException {

    // Create a document definition
    DocumentDefinition newDocumentDefinition =
        new DocumentDefinition(session);

    // Give this document a name
    newDocumentDefinition.setAttribute(PublicObject.NAME_ATTRIBUTE,
        AttributeValue.newAttributeValue(fileName));

    // Tell the system where to find this document's content
    // In this example we just point to a location on the local drive
    newDocumentDefinition.setContentPath("D:\\MathPapers\\Math.pdf");

    // Want to inherit the ACL from the parent folder
    newDocumentDefinition.setSecuringPublicObject(aFolder);

    // Specify in which folder to place this document
    newDocumentDefinition.setAddToFolderOption(aFolder);

    // Finally create this document in the database
    Document newDocument =
        (Document)session.createPublicObject(newDocumentDefinition);
}

```

It is worth noting that the document *content* is uploaded into the database by the method call:

```
newDocumentDefinition.setContentPath("D:\\MathPapers\\Math.pdf");
```

This is very different to many other content management systems that upload document metadata and just point to the content on an existing or obfuscated file system. We have in practice found that our approach is superior and provides many benefits.

FINDING YOUR CONTENT

Objects in the repository can be discovered in several ways:

- By unique object ID
- As the value of another object's attribute
- By folder path
- Through a collection (a memory-resident set) of LibraryObjects
- Through a search specified by a search specification object

Oracle CM SDK supports both simple and advanced searching.

Selector Search

The selector search is a simple search mechanism that allows you to find a specific object in the repository. You do this by creating a `selector` object and specifying the type of object that you are looking for, such as a document or a folder. You also specify a search criteria similar to the WHERE clause in a standard SQL query: for example to specify folders called NewFolder1 and NewFolder2 you would use the expression `"NAME like 'NewFolder%'"`. You can also specify a sort order and whether to search for subclasses. You can then navigate through the returned collection of objects.

Advanced Searches

The search API in Oracle CM SDK allows more advanced searching. You can search on the condition of attribute values of objects, or the existence of objects in the repository. You can also search based on the content of a document or documents. The classes in the API allow you to create an SQL-like query in Java. There are two types of search specifications that you can create: the attribute search specification and the context search specification. You build the search specification from several components:

- A search class specification which comprises the SELECT and FROM clauses.
- A search qualification which is the condition in the WHERE clause.
- A `SearchSortSpecification` object that allows you to specify an ordering for the returned objects. This maps to and QL ORDER BY clause.

With these objects, you can build sophisticated searches to more easily help you discover your content.

Content-Based Searches using Oracle Text

Content-based searches follow a similar pattern to the search described above. For content-based searches you build a context search specification for your search. This requires you to provide a context search qualification which, as we described above specifies the matching criteria for our searches. In this case, it corresponds to the CONTAINS clause used for Oracle Text content-based searches. The context search qualification allows you to build any valid search criteria supported by Oracle Text.

Because Oracle CM SDK is tightly integrated with Oracle Text, you can build extensive searching capabilities into your applications that take advantage of Oracle Text's advanced searching features. Oracle Text uses standard SQL to index, search, and analyze text and documents stored in the Oracle database, file servers and on the Web. Oracle Text can:

- Analyze document themes and gists
- Search text using variety of strategies including full-text: boolean, exact phrase, proximity, section searching, misspellings, stemming, wildcard, thesaurus, word equivalence, and scoring mixed: text index plus relational attributes, and thematic
- Search HTML and XML sections and tag values
- Render search results in various formats including unformatted text, HTML with automatic keyword highlighting, and original document format
- Analyze and index most document formats with over 150 document filters
- Oracle Text also supports multiple languages.

VERSIONING YOUR CONTENT

Oracle CM SDK provides a very powerful and flexible versioning model. The versions of a document form a family. This family allows you to manipulate all of the versions of a document as a single entity. The family contains a version series. The series contain a version description for each version of the document and can be used to support different versioning schemes. In figure 9 we can see the objects representing a Family for a versioned document.

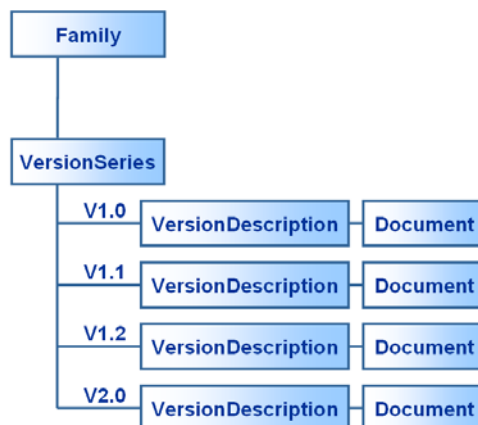


Figure 9: Oracle CM SDK versioned documents are represented by a family containing a version series.

SECURING YOUR CONTENT

One of the benefits of developing a content management solution based on Oracle CM SDK is that once you consolidate all of your content into one location, the repository, you can more easily enforce access control and security to control access to your content. Oracle CM SDK provides a sophisticated API for allowing access to the content.

Authentication

Oracle CM SDK provides two credential managers:

- Oracle CM SDK Credential Management (`IfsCredentialManager`)
- Oracle Internet Directory (OID) Credential Management (`OidCredentialManager`)

Oracle CM SDK also allows you to create your own custom credential manager.

The native Oracle CM SDK credential manager stores credential information in the Oracle CM SDK schema.

Oracle Internet Directory stores and manages users and groups in a central LDAP server, with all of the administration benefits this provides. Oracle CM SDK allows authentication by either the users' Single Sign-On (SSO) password, or by an Oracle CM SDK-specific password for authentication against some protocol servers. Since some protocols send unencrypted passwords, authenticating with the Oracle CM SDK-specific password ensures that even if a password is intercepted, the SSO password is not compromised. The Oracle CM SDK-specific password is stored in OID, but separately from the SSO password.

Once Oracle CM SDK is configured to use Oracle Internet Directory, users and groups are provisioned to Oracle CM SDK using one of several mechanisms:

- Manually, an administrator can create users directly with the Oracle CM SDK Manager tool and indicate that the user should be authenticated using OID.
- Automatically, when a user is created in OID, an event is sent to Oracle CM SDK and the user is provisioned.
- Periodically, by an OID credential manager agent that provisions the users after a specified interval, for example, every 15 minutes or every 24 hours.

Users can also be migrated from the Oracle CM SDK credential manager to OID using migration tools. In this way, applications built using Oracle CM SDK can take advantage of the lower administration overhead, added convenience and improved security of managing all of their corporate users in one location.

Authorization

Access to the content is governed by Oracle CM SDK Access Control Lists (ACLs). These ordered lists contain Access Control Entries (ACEs), which provide a mapping between a user or group and the permissions that they have on a

particular piece of content. The permissions a user has to a document can be granted or revoked, and depending where this ACE is in the ACL, the entry can modify previously assigned permissions for that document.

Custom permissions can be created using the extended permissions capability of Oracle CM SDK. The business logic to handle these extended permissions must also be created.

Permissions can be grouped into permission bundles, which let users group multiple access levels into a customized bundle. The purpose of this is to provide convenience over having to use each access level separately. Permission bundles can also act as a “Role”, since when this bundle is updated, all ACLs automatically reflect the change.

Oracle CM SDK allows you to organize users into groups. Groups represent sets of users who have something in common. For example, you can create groups for the various roles that users may have (e.g., Manager, Developer, Writer). You can also create groups to represent different companies divisions or projects (e.g., Marketing, Sales, Development). Groups can include other groups, allowing you to create inclusive hierarchies of users.

When a permission is granted to a group, if a user is added to the group, then that user is automatically granted the permissions of the group. This is useful for reducing the number of ACLs, which improves performance and can simplify the management of ACLs.

DRIVING YOUR CONTENT PROCESSES WITH ORACLE WORKFLOW

One of the more powerful components integrated with Oracle CM SDK is Oracle Workflow. This integration allows you to automate many of the business processes around your content such as routing the content to the various approvers, handling the approvals, and managing all of the workflow steps around generating and managing your content. Even beyond this, the workflow engine can also be used to automate more general business processes, like the steps in handling insurance claims or the steps in filing legal documents. With the Oracle Workflow Builder, you can define your own workflows and then build the logic in Oracle CM SDK to take advantage of these automated processes.

CONCLUSION

Oracle Content Management SDK provides everything that developers need to rapidly build sophisticated enterprise content management solutions. Content can be loaded into the repository straight out of the box. With a small amount of Web-based development, you can build a highly functional user interface that supports many of your content management requirements, such as, check-in/check-out, security and access control, workflow-based approvals and document routing.

ORACLE FUSION MIDDLEWARE

Oracle Content Management SDK: Concepts and Architecture

August 2005

Author: Simon Azriel

Contributing Authors: David Pitfield, Sylvia Perez and Marla Azriel

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

oracle.com

Copyright © 2005, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission. Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.