CDMI Reference Implementation – Developer's Guide

Version 1.0b

"Publication of this Working Draft for review and comment has been approved by the Cloud TWG. This draft represents a "best effort" attempt by the Cloud TWG to reach preliminary consensus, and it may be updated, replaced, or made obsolete at any time. This document should not be used as reference material or cited as other than a "work in progress." Suggestion for revision should be directed to the SNIA Technical Council Managing Director at tcmd@snia.org."

WORKING DRAFT

June 8, 2010

Revision History

Version	Date	Originator	Sections	Comments
1.0a	5/18/10	M. McMinn	All	1st draft
1.0b	6/8/10	M. Carlson	Ch 2, 3, 4	Added content.

Contents

Chapter 1: About this Guide

Purpose and Audience

The *CDMI Reference Implementation – Developer's Guide* is written for programmers and application developers who create custom applications for Cloud Storage Systems. This document, along with the *Cloud Data Management Interface Version 1.0*, provides the information that developers need to create custom applications for a Cloud Storage System.

Contents

The contents of this document are described as follows:

- Chapter 1, "About this Guide" describes the audience and purpose, contents of this guide, additional references and web sites, and typographical conventions.
- Chapter 2, "Introduction" lists the terms of use, software requirements, and supported operating systems.
- Chapter 3, "CDMI Architecture" describes the components of the reference implementation and the theory of operations.
- Chapter 4, "CDMI Capabilities" describes the capabilities of the reference implementation.
- Chapter 5, "CDMI Test Architecture" describes how to perform typical tests and how to add your own tests.
- Chapter 6, "CDMI Error Code Mapping" maps the error codes in the reference implementation to JavaExceptions.

References

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Comment 1: TBD.

Additional Information

The SNIA web site provides additional information about the SNIA Cloud Storage Initiative at **http://www.snia.org/cloud.**

Conventions

Table 1 describes the typographic conventions that are used in this document.

Table 1 – Typographic Conventions

Convention	Description
`Fixed-width text`	The names of commands, files and directories, and on-screen computer output
`Bold, fixed-width text`	What you type, contrasted with on-screen computer output
Italicized text	Variables, field names, and book titles
Note:	Additional or useful informative text.
WARNING:	Indicates that you should pay careful attention to the probable action, so that you may avoid system failure or harm.

SNIA Welcomes Your Comments

SNIA is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by sending an e-mail to **tcmd@snia.org.**

Chapter 2: Introduction

This chapter provides an introduction to the SNIA *CDMI Reference Implementation - Developer's Guide*.

The following topics are discussed:

- Terms of Use
- Supported Operating Systems
- Software Requirements

Terms of Use

This developer's guide is released to you under the following copyright notice:

Supported Operating Systems

The reference implementation is written in Java and should work on any platform that supports Java version 1.6 and above.

Software Requirements

Installing Dependencies

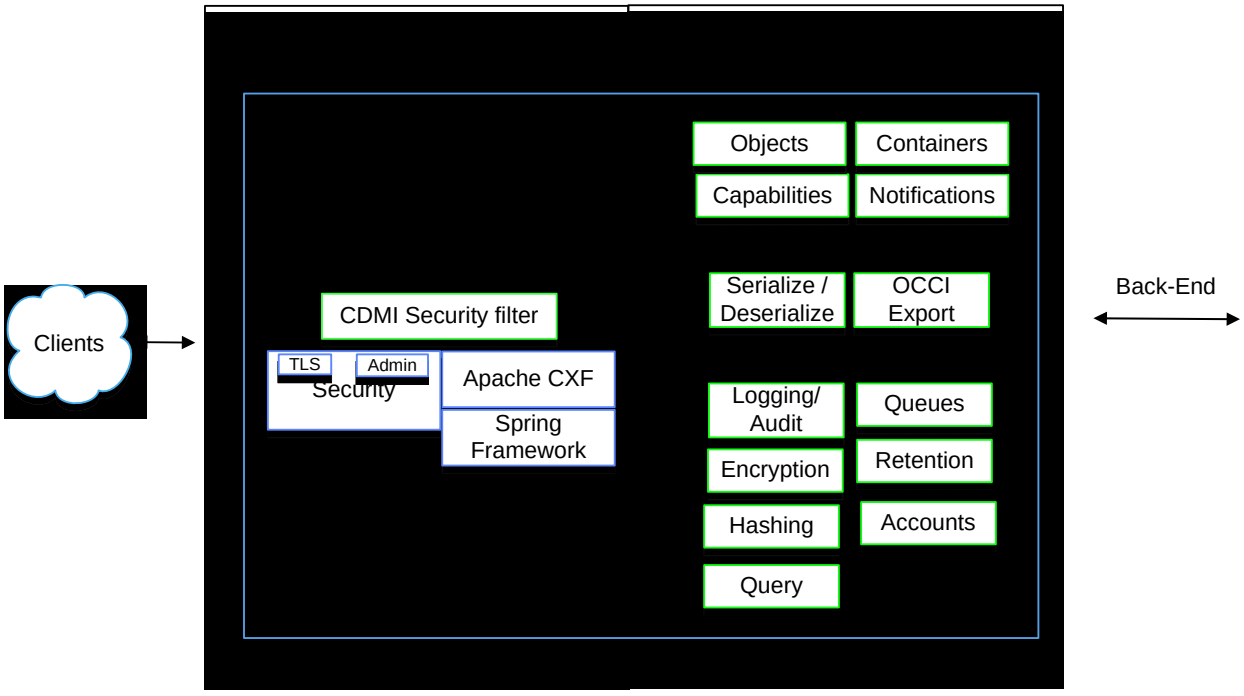
1. Install version 1.6 of Java.
2. Download and install version 2.0.9 of Maven. See http://maven.apache.org/ for details.
3. Install an IDE such as NetBeans or Eclipse if desired.

Chapter 3: CDMI Architecture

This chapter provides information about the architecture of a Cloud storage system.

Architecture Overview

Figure 1 and Figure 2 show the relationships of the front-end server, the mid-layer, and the back-end store of the CDMI architecture reference implementation.



Green: SNIA-developed code
Blue: 3rd-party code

Figure 1 – CDMI Architecture: Front-End Server and Mid-Layer

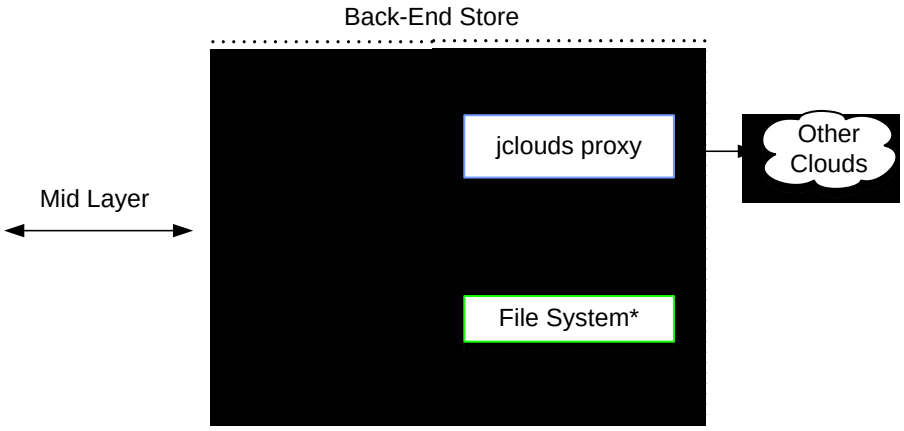


Figure 2 – CDMI Architecture: Mid-Layer, Back-End Store, and Other Clouds

File System Naming

The file system for the CDMI architecture contains the following naming conventions:

- Container Objects = Folders named with the container name
- Data objects = Files named with the object name, if one was given, else the Object ID
- Metadata = Files named with the same name as the corresponding object with an additional "." in front

Examples

Examples of file names for containers, container metadata, and data object data and metadata include the following:

- Container: `/mnt/cdmi server/MyContainer`
- Container metadata: `/mnt/cdmi server/.MyContainer`
- Data object's data: `/mnt/cdmi server/MyContainer/MyDataObject.txt`
- Data object's metadata: `/mnt/cdmi server/MyContainer/.MyDataObject.txt`
- Data objects data: `/mnt/cdmi server/MyContainer/0000706D0010B84FAD185C425D8B537E`
- Data object's metadata: `/mnt/cdmi server/MyContainer/.0000706D0010B84FAD185C425D8B537E`

Theory of Operations

This reference implementation is based on the Apache CXF implementation of JAX-RS, which gets along well with Spring, so it also uses Spring

dependency injection design patterns for all of the application pieces. The various packages contain the following:

(1) `org.snia.cdmiserver.dao`

The DAO interfaces for the various CDMI data types.

(2) `org.snia.cdmiserver.dao.filesystem`

The implementations of the DAO interfaces, using an ordinary filesystem base directory to store things in, and modelling containers as directories and data objects as files. For each resource implementation, there is a corresponding bean in the Spring configuration file: (`applicationContext.xml`).

(3) `org.snia.cdmiserver.exception`

RuntimeException implementations that a JAX-RS resource method is allowed to throw, which are converted (by an exception mapper; see below) into an appropriate HTTP response with the right status code. For example, if a resource method throws NotFoundException, a 404 response will be generated.

If you add any additional exception classes here, you will also need to add a new exception mapper class into `org.snia.cdmiserver.provider` (and add it to the Spring configuration).

(4) `org.snia.cdmiserver.filter`

Originally contains two Jersey-specific filters, to enforce validation of the "X-CDMI-Specification-Version" header matching a version supported by this server, and a security filter to do authentication and authorization. You'll likely want to find CXF-compatible replacements for these two functions, since they need to apply to basically all of the resource methods.

(5) `org.snia.cdmiserver.model`

Simple JavaBean classes for the various CDMI data structures. For each one, you'll need a provider class to support conversion to/from JSON, following the exact rules in the spec. Only the Capability, Container, and DataObject classes have been fleshed out so far.

(6) `org.snia.cdmiserver.provider`

JAX-RS exception mappers for all the exception classes that you want to allow a resource method to throw (and then get converted into an appropriate HTTP status), and providers for converting CDMI data structures to/from JSON. As you add new exception mapper or provider implementations, don't forget to add configuration data in the Spring configuration file (`applicationContext.xml`).

(7) `org.snia.cdmiserver.resource`

JAX-RS resource classes for the container and data object related calls. Because of the open ended way the spec defines mapping URIs to methods, you'll end up (in PathResource) sometimes having to deal with the entire path (which is passed in as a method parameter).

Note: Resource classes should be declared with scope="prototype", meaning you'll get a new one for each request. Otherwise, the logic in these classes won't be threadsafe.

`(8) org.snia.cdmiserver.util`

A spot for generic utility classes.

Chapter 4: CDMI Capabilities

This chapter lists the following capabilities of the CDMI Reference Implementation:

- Cloud Storage System-Wide Capabilities
- Storage System Metadata Capabilities
- Data System Metadata Capabilities
- Data Object Capabilities
- Container Capabilities
- Domain Capabilities
- Queue Object Capabilities

Cloud Storage System-Wide Capabilities

Table 2 lists the system-wide capabilities that are supported in this reference implementation of a cloud storage system. These capabilities, which are found in the capabilities object, are referred to by the root URI (root capabilities).

Table 2 – Cloud Storage System-Wide Capabilities

Capability	Comment
cdmi_domains	Not implemented in this release
cdmi_export_occi_iscsi	Not implemented in this release
cdmi_metadata_maxitems	
cdmi_metadata_maxsize	
cdmi_notification	Not implemented in this release
cdmi_query	Not implemented in this release
cdmi_queues	Not implemented in this release

Table 2 – Cloud Storage System-Wide Capabilities

Capability	Comment
cdmi_security_audit	Not implemented in this release
cdmi_security_data_integrity	Not implemented in this release
cdmi_security_encryption	Not implemented in this release
cdmi_security_https_transport	Not implemented in this release
cdmi_security_immutability	Not implemented in this release
cdmi_security_sanitization	Not implemented in this release
cdmi_serialization_json	Not implemented in this release

Storage System Metadata Capabilities

Table 3 lists the storage system metadata capabilities that are supported in this reference implementation of a cloud storage system. These capabilities are found in the capabilities objects for domains, data objects, containers, and queues.

Table 3 – Storage System Metadata Capabilities

Capability	Comment
cdmi_size	From filesystem
cdmi_billingsize	Not implemented in this release
cdmi_ctime	From filesystem
cdmi_mtime	From filesystem
cdmi_acount	Not implemented in this release
cdmi_mcount	Not implemented in this release
cdmi_hash	Not implemented in this release
cdmi_acl	Not implemented in this release

Data System Metadata Capabilities

Table 4 lists the capabilities for data system metadata that are supported in this reference implementation of a cloud storage system. These capabilities are found in the capabilities objects for domains, data objects, containers, and queues.

Table 4 – Data System Metadata Capabilities

Capability	Comment
cdmi_data_redundancy	Always set to 1
cdmi_infrastructure_redundancy	Always set to 1
cdmi_data_dispersion	Unset. Will not be implemented.

Table 4 – Data System Metadata Capabilities

Capability	Comment
cdmi_data_retention	Not implemented in this release
cdmi_data_autodelete	Not implemented in this release
cdmi_data_holds	Not implemented in this release
cdmi_encryption	Not implemented in this release
cdmi_value_hash	Not implemented in this release
cdmi_max_latency	Not implemented in this release
cdmi_max_throughput	Not implemented in this release
cdmi_sanitation_method	Not implemented in this release
cdmi_RPO	Unset. Will not be implemented.
cdmi_RTO	Unset. Will not be implemented.

Data Object Capabilities

Table 5 lists the capabilities for data objects that are supported in this reference implementation of a cloud storage system.

Table 5 – Data Object Capabilities

Capability	Comment
cdmi_read_value	Always true
cdmi_read_value_range	Not implemented in this release
cdmi_read_metadata	Always true
cdmi_modify_value	Always true
cdmi_modify_value_range	Not implemented in this release
cdmi_modify_metadata	Always true
cdmi_serialize_dataobject	Not implemented in this release
cdmi_deserialize_dataobject	Not implemented in this release
cdmi_delete_dataobject	Always true

Container Capabilities

Table 6 lists the capabilities for containers that are supported in this reference implementation of a cloud storage system.

Table 6 – Container Capabilities

Capability	Comment
cdmi_list_children	Always true
cdmi_list_children_range	Not implemented in this release

Table 6 – Container Capabilities

Capability	Comment
cdmi_read_metadata	Always true
cdmi_modify_metadata	Always true
cdmi_snapshot	Not implemented in this release
cdmi_serialize_container	Not implemented in this release
cdmi_create_dataobject	Always true
cdmi_post_dataobject	Always true
cdmi_create_container	Always true
cdmi_create_queue	Not implemented in this release
cdmi_create_reference	Not implemented in this release
cdmi_delete_container	Always true
cdmi_move_container	Not implemented in this release
cdmi_copy_container	Not implemented in this release

Domain Capabilities

Not implemented in this release

Queue Object Capabilities

Not implemented in this release

Chapter 5: CDMI Test Architecture

Comment 2: TBD.

Chapter 6: CDMI Error Code Mapping

The following Java exceptions should be mapped to HTTP status codes (see Table 7).

Table 7 – Error Code Mapping

Java Exception	HTTP Status Code
BadRequestException	400 - Bad Request
ConflictException	409 - Conflict
ForbiddenException	403 - Forbidden
NotFoundException	404 - Not Found
UnauthorizedException	401 - Unauthorized