Reference Guide / CMIS Administrator

Chapter 1. Introduction

It is to introduce about basic knowledge of eXo CMIS, working principles and how to configure it.

1.1. CMIS Specification

Note

Content Management Interoperability Services (CMIS) Version 1.0 OASIS Standard 1 May 2010

[http://en.wikipedia.org/wiki/Content_Management_Interoperability_Services]

The Content Management Interoperability Services (CMIS) standard defines a domain model and Web Services and Restful AtomPub bindings that can be used by applications to work with one or more Content Management repositories/systems. The CMIS interface is designed to be layered on top of existing Content Management systems and their existing programmatic interfaces. It is not intended to prescribe how specific features should be implemented within those CM systems, nor to exhaustively expose all of the CM systems capabilities through the CMIS interfaces. Rather, it is intended to define a generic/universal set of capabilities provided by a CM system and a set of services for working with those capabilities.

The CMIS specification provides a Web services interface that:

- Is designed to work over existing repositories, enabling customers to build and leverage applications against multiple repositories.
- Decouples Web services and content from the content management repository, enabling customers to manage content independently.
- Provides common Web services and Web 2.0 interfaces to dramatically simplify application development.
- Is development platform and language agnostic.
- Supports the composite application development and mash-up by the business or IT analysts.

1.2. xCMIS project

The xCMIS project http://xcmis.org/, initially contributed to the Open Source community by http://www.exoplatform.com , is an implementation of the full stack of Java-based CMIS services. xCMIS also includes the client side frameworks for integrating content from different enterprise repositories, according to the http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cmis .

The project is distributed under the LGPL license. You can download sources at http://code.google.com/p/xcmis/source/checkout , or visit http://code.google.com/p/xcmis/w/list for more information.

1.3. eXo CMIS

eXo CMIS is one eXo Platform service exposing eXo Content via CMIS. eXo CMIS offers the way to access

the eXo ECMS content from the CMIS 1.0 compatible Web client using the REST Atom protocol.

Note

WS-SOAP/WSDL binding is also supported, but it is not delivered in the standard eXo Platform 3.0 bundle.

eXo CMIS is built on the top of xCMIS embedded in Platform to expose the WCM drives as the CMIS repositories. The CMIS features are implemented as a set of components deployed on the eXo Container using XML files to describe the service configuration.

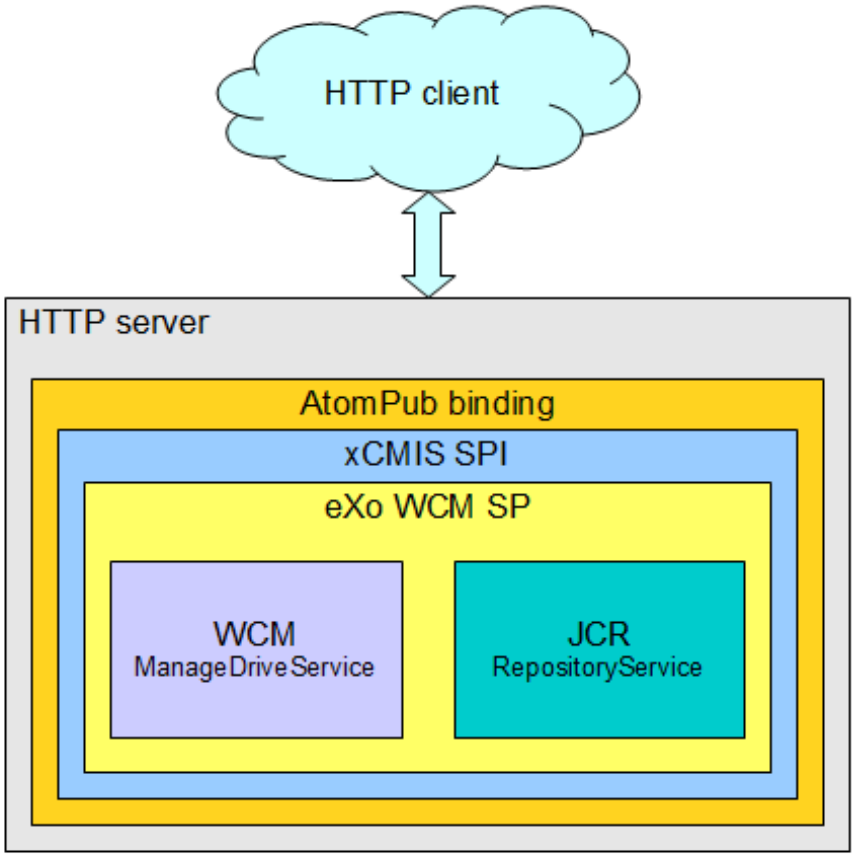


Figure: how eXo CMIS works

WCM drives exposure is implemented as WCM storage provider to xCMIS SPI. Storage provider uses mappings from WCM's *ManageDriveService* to actual JCR nodes. And *AtomPub* bindings makes WCM structure available via CMIS standard API.

Chapter 2. Configuration

2.1. CMIS Configuration

To expose WCM drives to the CMIS repositories, you must make a special extension of *CmisRegistry*.

To make a typical component *org.exoplatform.ecms.xcmis.sp.jcr.exo.DriveCmisRegistry*, do as follows:

```
<component>
   <type>org.exoplatform.ecms.xcmis.sp.jcr.exo.DriveCmisRegistry</type>
   <init-params>
      <!-- Disabled by default. Uncomment if you need query support in CMIS. -->
         <!-- value-param>
            <name>indexDir</name>
            <value>${gatein.jcr.index.data.dir}/cmis-index${container.name.suffix}</value>
         </value-param-->
      <value-param>
         <name>exo.cmis.renditions.persistent</name>
         <value>true</value>
      </value-param>
      <values-param>
         <name>renditionProviders</name>
         <description>Redition providers classes.</description>
         <!-- <value>org.xcmis.renditions.impl.PDFDocumentRenditionProvider</value> -->
         <value>org.xcmis.renditions.impl.ImageRenditionProvider</value>
      </values-param>
   </init-params>
</component>
```

Where configuration parameters include:

- indexDir - directory where the lucene index will be placed. It is disabled by default.
- exo.cmis.renditions.persistent - Indicates if a rendition of the document (thumbnails) should be persisted in the JCR. The allowed value are *true* or *false*.
- renditionProviders - set of FQN of classes which can be used as Rendition Providers. Classes which implement the org.xcmis.spi.RenditionProvider interface are used to preview the CMIS objects (thumbnails).

Note

In most cases, it is not required to change anything in the xCIMIS configuration. In case of any change of the indexer storage location, do not comment the indexDir value parameter and point it to the actual location.

2.2. Required nodetypes and namespaces in JCR

The following configuration is mandatory for JCR to work correctly:

```
<external-component-plugins>
   <target-component>org.exoplatform.services.jcr.RepositoryService</target-component>
   <component-plugin>
      <name>add.namespaces</name>
      <set-method>addPlugin</set-method>
      <type>org.exoplatform.services.jcr.impl.AddNamespacesPlugin</type>
      <init-params>
```

```
         <properties-param>
            <name>namespaces</name>
            <property name="cmis" value="http://www.exoplatform.com/jcr/cmis/1.0" />
            <property name="xcmis" value="http://www.exoplatform.com/jcr/xcmis/1.0" />
         </properties-param>
      </init-params>
   </component-plugin>
   <component-plugin>
      <name>add.nodeType</name>
      <set-method>addPlugin</set-method>
      <type>org.exoplatform.services.jcr.impl.AddNodeTypePlugin</type>
      <init-params>
        <values-param>
            <name>autoCreatedInNewRepository</name>
            <description>Node types configuration file</description>
            <value>jar:/conf/cmis-nodetypes-config.xml</value>
         </values-param>
      </init-params>
   </component-plugin>
</external-component-plugins>
```

2.3. Authenticator and organization service configuration

An Authenticator is responsible for creating Identity Security Service Authenticator

The eXo CMIS service is based on:

- The authentication mechanism provided by the eXo organization service.
- The JAAS configuration of eXo Platform. For example,

```
gatein-domain {
  org.exoplatform.web.security.PortalLoginModule required;
  org.exoplatform.services.security.jaas.SharedStateLoginModule required;
  org.exoplatform.services.security.j2ee.TomcatLoginModule required;
};
```

2.4. CMIS search and index

The CMIS standard defines a query language based on on a subset of the SQL-92 grammar (ISO/IEC 9075: 1992 -- Database Language SQL), with a few extensions to enhance its filtering capability for the CMIS data model, such as existential quantification for multi-valued property, full-text search, and folder membership.

Warning

CMIS search is disabled by default in eXo CMIS. Uncomment indexDir parameter if you need query support in CMIS. To discover search capability check capabilityQuery property (see table below).

2.4.1. CMIS Relational View

The relational view of a CMIS repository consists of a collection of virtual tables that are defined on top of the CMIS data model. A virtual table exists for every queryable object type (content type if you prefer) in the repository. Each row in these virtual tables correspond to an instance of the corresponding object type (or of

one of its subtypes). A column exists for every property that the object type has.

2.4.2. Query Capabilities

Table 2.1.

Capability	Value
capabilityQuery	`bothcombined` (if indexDir configured, otherwise `none`)
capabilityJoin	`none`
capabilityPWCSearchable	`false`
capabilityAllVersionsSearchable	`false`

2.4.3. Configuration

To be able to provide full-text search capabilities xCMIS uses its own index. There is configuration parameter:

Table 2.2.

Parameter	Default	Description
indexDir	none	The location of the index directory. This parameter is mandatory for default implementation

All these parameters can be passed through the xml configuration.

How to set up the index directory.

```
<component>
   <type>org.exoplatform.ecms.xcmis.sp.DriveCmisRegistry</type>
   <init-params>
         <!-- Disabled by default. Uncomment if you need query support in CMIS. -->
         <!-- value-param>
            <name>indexDir</name>
            <value>${gatein.jcr.index.data.dir}/cmis-index${container.name.suffix}</value>
         </value-param-->
         ..............
   </init-params>
</component>
```

2.4.4. Index atomicity and durability

- Write-ahead logging

To be able to provide index consistency and recoverability in the case of unexpected crashes or damages, XCMIS uses write-ahead logging (WAL) technique. Write-Ahead Logging is a standard approach to

transaction logging. Briefly, WAL's central concept is that changes to data files (indexes) must be written only after those changes have been logged, that is, when log records describing the changes have been flushed to permanent storage. If we follow this procedure, we do not need to flush data pages to disk on every transaction commit, because we know that in the event of a crash we will be able to recover the index using the log: any changes that have not been applied to the data pages can be redone from the log records. (This is roll-forward recovery, also known as REDO.)

A major benefit of using WAL is a significantly reduced number of disk writes, because only the log file needs to be flushed to disk at the time of transaction commit, rather than every data file changed by the transaction.

- Recover uncommited transaction

When you start Indexer, it will check uncommitted transaction logs. If there are at least one log exists - recover process will be started. Indexer will read all logs and extract added, updated and removed uuids into set. Then it will walk throw this set and check objects according to UUID. If object exist - indexer will put into the added documents list, in other case uuid will be added to removed documents list. After according to the list of added and removed documents changes will be applyed to the index.

- Initial index population

When you run the indexer checks the number of documents in the index. If there is no documents in the index or previous re-indexation wasn't successful then re-indexation of all content will be started. First step is cleaning old index data. Uncommited transaction logs and old persistent data is removed. This data is useless, because re-indexation of all content will be started. Then indexer walks throw all objects and make lucene document for each one. Then batches with less then 100 elements will be saved to the index. After re-indexation, all logs (WAL) will be removed, all data mentioned on this changes logs - already indexed.

Note

If administrator get exception with message "Can't remove reindex flag." it means that restore index was finished but file-flag was not removed (see index directory, file named as "reindexProcessing"). You can manualy remove this file-flag, and avoid new reindex of repository on jcr start.

Chapter 3. Service JARs

Note

JCRs listed for informational purpose only. E.g. if the customer application will need to use same third-parties used by eXo CMIS but of another version. All files are delivered in Platform 3.0 distribution.

eXo CMIS consists of JAR files:

xCMIS project files (actual for the xCMIS 1.2.x versions):

- xcmis-renditions-X.X.X.jar
- xcmis-restatom-X.X.X.jar
- xcmis-search-model-X.X.X.jar
- xcmis-search-parser-cmis-X.X.X.jar
- xcmis-search-service-X.X.X.jar
- xcmis-spi-X.X.X.jar

eXo WCM Storage Provider for xCMIS SPI (uses the same version as WCM/ECMS system):

- exo-ecms-ext-xcmis-sp-Y.Y.Y.jar

Third-party dependencies (of xCMIS 1.2.x):

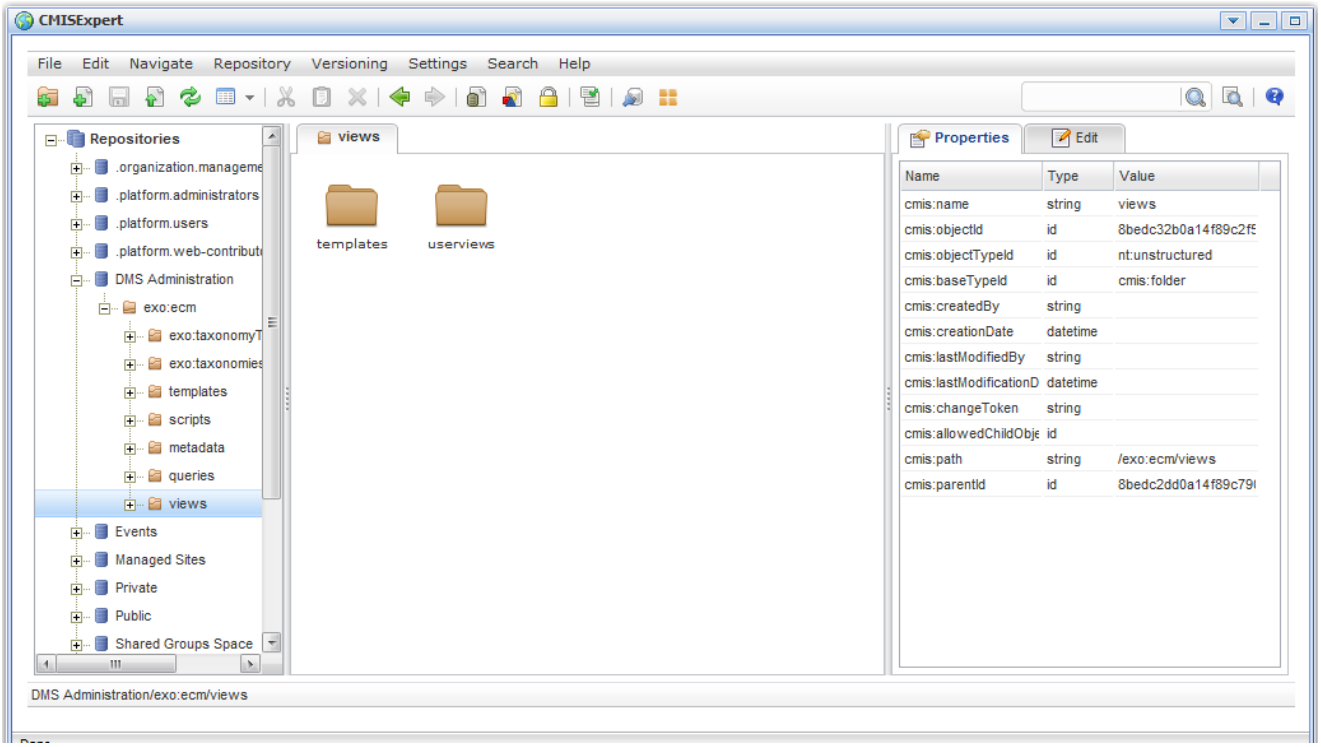
- *abdera-client-0.4.0-incubating.jar*
- *abdera-core-0.4.0-incubating.jar*
- *abdera-i18n-0.4.0-incubating.jar*
- *abdera-parser-0.4.0-incubating.jar*
- *abdera-server-0.4.0-incubating.jar*
- *antlr-runtime-3.1.3.jar*
- *axiom-api-1.2.5.jar*
- *axiom-impl-1.2.5.jar*
- *jaxen-1.1.1.jar*
- *lucene-regex-2.4.1.jar*

etc.

Chapter 4. Miscellaneous and Tips

Clients can be useful for CMIS repositories access:

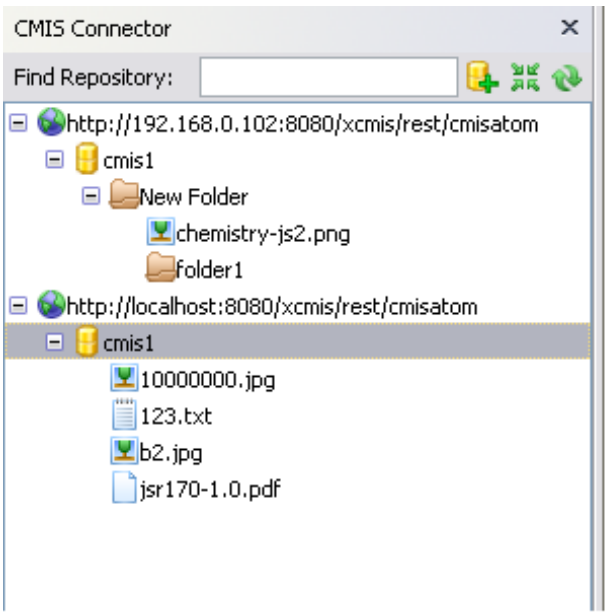
- CMIS Expert, available at http://www.xcmis.org as a Google gadget.



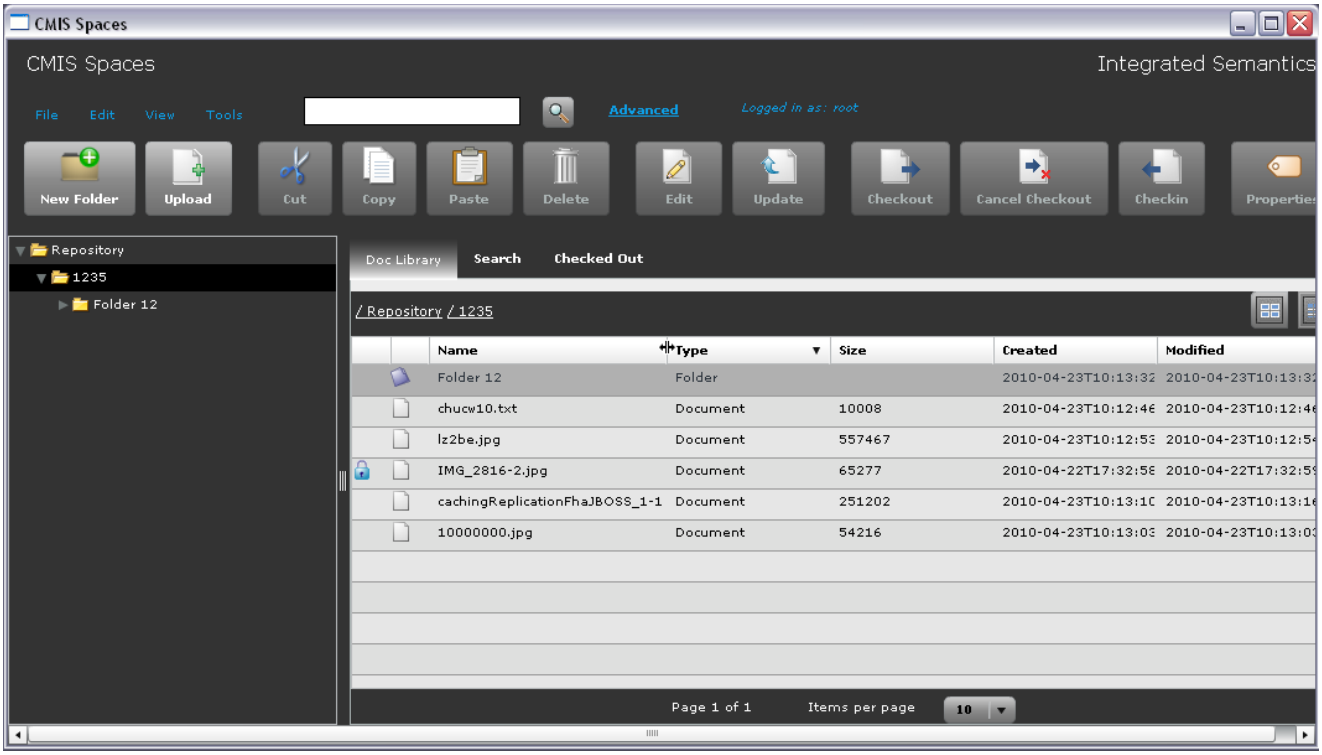
- IBM [CMIS Firefox Connector](#)

Warning

The IBM CMIS Firefox plugin is not compatible with Firefox 3.6.



- Flex+AIR and Flex+Browser CMIS clients, based on FlexSpaces framework. http://code.google.com/p/cmisspaces



Chapter 5. Links

eXo CMIS links

xCMIS project links:

- Community site http://xcmis.org/ with CMIS Expert demo
- Forum http://groups.google.com/group/xcmis
- xCMIS latest news http://gazarenkov.blogspot.com/
- eXo Platform blog http://blog.exoplatform.org/

CMIS related links:

- http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cmis
- TODO CMIS working group (list, mail, feeds etc)
- TODO CMIS external tutorials, guides, videos, blogs, news
- CMIS clients
  - CMIS Firefox Connector
  - Flex+AIR and Flex+Browser CMIS clients http://code.google.com/p/cmisspaces