Version 0.50

Content Management Interoperability Services

Part II –REST protocol binding

8/28/2008

CONTENTS

COPYRIGHT NOTICE

INTRODUCTION

The Content Management Interoperability Services (CMIS) standard will define a domain model and set of bindings, such as Web Service and REST/Atom 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 system's 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.

NOTATION (RFC 2119)

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, [RFC2119], as scoped to those conformance targets.

PART I OF THE CMIS SPECIFICATION

Part I of the CMIS specification can currently be found at this URL:

https://sharepoint.partners.extranet.microsoft.com/sites/CMIS/review/Specification/

Part I of the CMIS specification introduces the CMIS standard, discusses general design concepts, and describes the Data Model and services for CMIS that are common to all protocol bindings.

REPRESENTATIONAL STATE TRANSFER (REST)/ATOM PROTOCOL BINDING

REFERENCES

- Atom Publishing Protocol (APP) - http://www.ietf.org/rfc/rfc5023.txt
- Atom Syndication Format - http://www.ietf.org/rfc/rfc4287.txt
- Hypertext Transfer Protocol - HTTP/1.1 -- http://www.w3.org/Protocols/rfc2616/rfc2616.html
- RFC 5005 – Feed Paging

OVERVIEW

The REST binding has one mode: Pure-ATOM following naming conventions and additional information in ATOM documents

The client will request the service document at the URL provided by vendor. The client will then choose a collection, and then start accessing the repository.

The URI for different items are found off of the atom feed document in the link tags for operations Atom or APP does not natively support. The tags have special names that denote meaning with the CMIS: prefix. Optional parameters are passed in as HTTP headers.

Custom properties will be part of the CMIS namespace using generic property tags.

Special collections have been created that have semantic meaning beyond collection membership. These are:

- Unfiled – All documents added to this collection will be removed from all other collections
- Checkedout – All documents added to this collection will be checkedout

Authentication

Authentication will be handled by the transport protocol.

Response Formats

The client can specify in HTTP the Accept header which specifies which formats are acceptable to the client. With this mechanism the client can chose which response format the CMIS should respond with. CMIS compliant implementation MUST support at least this one response format:

- application/atom+xml

The CMIS repository will chose the response format based on the request format if the Accept header is not there. The repository will not be able to represent multi-request CMIS batch in an atom feed and will use HTTP semantics.

When the client sends an Accept header that applies to more than one format, the CMIS Atom format will be chosen.

The CMIS repository is free to support other formats such as:

- application/x-javascript (or text/javascript) for JSON
- text/html for an HTML interface to the API

Optional Arguments

The REST/ATOM binding supports the following mechanisms for supplying arguments:

- HTTP Headers
- URI arguments

The REST proposal, in particular the ATOM flavor of it, uses HTTP headers for optional arguments. Most language bindings provide access to HTTP headers. For some environments such as Flex, URI arguments must be used.

HTTP Headers will have a 'CMIS-' prefix in front of the argument name.

CMIS implementations MUST support arguments being specified in either mechanism.

Properties

Properties will be on the Atom Entry document in a cmis:properties element. Properties defined in the Data Model will be in the schema. Custom properties defined at the repository will leverage the xs:any tag and look the same as the properties in the Data Model, however, they won't be in the schema. An example of a custom property, keyword:

```
<cmis:keyword>foo</cmis:keyword>
```

If the repository finds any element in the cmis namespace in the cmis:properties element, that element must correspond to a property on the type.

Method Overrides

In case there is a proxy that is blocking PUT and DELETE, X-Method-Override header SHOULD be supported. This is not currently in an RFC, but started with Google Da–a - http://code.google.com/apis/gdata/basics.html#Updating-an-entry with support for X-HTTP-Method-Override

Since Google introduced this header, applications such as Apache, DOJO and Project.0 from IBM among others have added support for it.

MAPPING CMIS TO ATOM FORMATS

CMIS RESOURCES

In a REST based model, entities are mapped to resources that then have operations performed on them. This is the corner-stone of the REST model. In atom, these resources are exposed as links on feed and entry documents. The base CMIS domain model includes:

Resource (Document Type)	Description	HTTP Methods			
		GET	PUT	POST	DELETE
Repository (Service)	This is the repository logical object. It is represented by the atom service document. This is also exposed on entry as link 'cmis-repository'. Each repository is mapped to a Workspace in the Service document	getRepositories or getRepository-Info (Service)			
Root Folder Collection (Feed)	This is a collection described in the service document	getChildren (Feed)			
Query collection	This is a collection of persisted Queries	n/a (Feed)		Query (Query)	

Checked Out Collection (Feed)	This is a collection described in the service document that contains all the checkedout documents	getChecked-OutDocuments (Feed)		or checkout (POST doc to checkedout) (Entry)	
Unfiled Collection (Feed)	This is a collection of unfiled documents.	This will return an empty collection. (Feed)		remove-DocumentFrom-Folder or removeDocument-FromAllFolders (Entry)	
Types Children (Feed)	This is a collection described in the service document that contains all the types in the repository This is a feed.	getTypes (Feed)			
Types Descendants (Feed)	This is a collection described in the service document that contains all the types in the repository. This is a feed.	getTypes (Feed)			

Type (Entry)	This is the CMIS type of the object. This is exposed as link 'cmis-type', This is enclosed as an atom entry	getTypeDefinition (Entry)			
Document (Entry)	This is a CMIS Document instance. These are exposed in feeds or in an entry document. This can also be a private working copy (checkedout)	getProperties (Entry)	UpdateProperties (Entry)		Delete (N/A)
Document Private Working Copy (Entry)	This is a document in the private working copy of the document (checkedout version of document)	getProperties (Entry)	UpdateProperties Or checkin (with checkin param) (Entry)		Cancel-Checkout (N/A)
Folder (Entry)	This is a CMIS Folder instance. This is exposed as a feed. The properties of a folder map onto the feed tag.	getProperties (Entry)	UpdateProperties (Entry)		Delete (N/A)

Relationship (Entry)	This is a CMIS relationship instance. These objects are exposed via 'cmis-relationships'	getProperties (Entry)	UpdateProperties (Entry)		Delete (N/A)
Policy (Entry)	This is a CMIS policy instance. This is exposed via 'cmis-policies' as a feed	getProperties (Entry)	UpdateProperties (Entry)		Delete (N/A)
Content Stream (Non-XML)	This is the content stream portion of the document object. This is exposed via 'cmis-stream' on the entry	getContent-Stream (Any)	SetContent-Stream (Any)		DeleteContent-Stream (N/A)
Allowable Actions (Allowable-Actions)	This is the set of actions available to the current user on the current object. This is exposed as a link 'cmis-allowableactions'	getAllowable-Actions (AllowableActions)			
Relationships Collection (for a specific item) (Feed)	This is the set of relationships available (either source or target or both) from a specific item such as a document, folder or policy			createRelationship (POST to source folder) (Entry)	

Parents Collection (for a specific document or policy object) (Feed)	This is the set of parents for a specific document or policy object. This can also be the ancestor of folders if returnToRoot is selected	getDocument-Parents or getFolder-Parents with returnToRoot (Feed)			
Children (Feed)	This is a pseudo-object comprised of all the direct children of a particular folder. This is exposed as 'cmis-children'.	getChildren (Feed)		createDocument (POST to folder) createFolder (POST to folder) createPolicy or moveObject or addDocument-ToFolder (Entry)	
Descendants (Feed)	This is a pseudo-object comprised of all the direct and indirect children of a particular folder. This is exposed as 'cmis-descendants'	getDescendants (Feed)		createDocument (POST to folder) createFolder (POST to folder) createPolicy or moveObject or addDocument-ToFolder (Entry)	deleteTree (N/A)

AllVersions (Feed)	This is a pseudo-object made up of all document versions related to the current document version. This collection is also known as the version history	getAllVersions (Feed)			DeleteAll-Versions (N/A)

DOCUMENT TYPES

Document Type	Description	MimeType	Namespace	Starting Tag
Service	This is the APP Service Document	application/atomsvc+xml	APP	service
Feed	This is an Atom Feed	Application/atom+xml;type=feed	ATOM	feed
Entry	This is an Atom Entry	Application/atom+xml;type=entry	ATOM	entry
Query	This is a CMIS Query document	Application/cmisquery+xml	CMIS	query
AllowableActions	This is a CMIS AllowableActions document	Application/cmisallowableactions+xml	CMIS	allowableactions

CMIS LINK TYPES FOR ATOM

CMIS links are all lowercase and have the prefix 'cmis-' before the link name. The table below outlines the different link types in CMIS.

Also, Links can have the following attributes:

- (ATOM) href: Specifies the URI of the link
- (ATOM) rel: Specifies the relationship of the link
- (ATOM) type: Specifies the Atom Media type of link (Mime/Type)
- (ATOM) hreflang: Specifies the language of the link
- (ATOM) title: Specifies the title of the link
- (ATOM) length: Specifies the length of the link
- (CMIS) id: Specifies the CMIS ID of the resource pointed at the link. It is recommended to include this attribute.

CMIS Link	Description	Type	Document	Folder	Relationship	Actions	Policy	Feeds
parents	This is the collection of parents, specifically the folders a document is filed into Type: Feed	X (Type of P)	X For unfiled documents, link is not available	X (Parent Folder) Link is not available on root folder's or objects that do not have parents				
repository	This points back to the service document containing the workspace. The current repository will use relationship self on the workspace element Type: Service	X	X	X	X	X	X	X

children	This is the collection of children for a folder or type Type: Feed	X		X			X If Children, or Descendants, provide links to the relevant alternate listings
descendants	This is the collection of children for a folder or type Type: Feed (Tree)	X		X			X If Children, or Descendants, provide links to the relevant alternate listings
Allowable-actions	This points to the set of allowable actions for the respective objects Type: AllowableActions		X	X	X	X	
allversions	This points to the version history (set of versions) for the document Type: Feed		X				
latestversion	This always points to the latest version (URI stable)		X				

	Type: Entry							
relationships	This points to the set of relationship on the current object Type: Feed		X	X			X	
type	This points to the type definition for the current object Type: Type	X	X	X	X		X	
source	In a relationship, this points to the source object. In Feeds, this points to the entry object providing the feed Type: Source object type (Entry, Type)				X for Relationship entry (For Feeds, points to the atom entry generating feed)	X (Object Actions are against)	(For Feeds, points to the atom entry generating feed)	X (For Feeds, points to the atom entry generating feed)
target	In a relationship, this points to the target object				X			
stream	This points to the file for the document		X					

ATOM Link		Type	Document	Folder	Relationship	Policy
self		X	X	X	X	X
alternate			X			
edit			X	X	X	X
edit-media			X			

DOCUMENT FORMATS

ATOM SERVICE DOCUMENTS

How the client will get the ATOM (APP) service document is repository specific. Examples are via URI, or loading the service document from disk.

In the Atom (APP) Service Document, each workspace maps to a single repository. A workspace element for a CMIS repository will have a collection element for the following collections. Please see enumCollectionType for the values. :

- Root folder: Root folder of the Repository
    - 'root-children' for the children collection of the root folder
    - 'root-descendants' for the descendants collection of the root folder
- Unfiled folder: Folder for posting documents to be unfiled; read can be disabled
    - 'unfiled'
- Checkedout Folder: Folder containing all checked out documents user can see
    - 'checkedout'
- Types Folder: Folder containing all the types in the repository
    - 'types-children' for the children collection
    - 'types-descendants for the children collection
- Query collection: Collection for posting queries to be executed

The workspace element will have two CMIS attributes: id and repositoryRelationship. RepositoryRelationship specifies the relationship of the repository to others listed in the service document. The repository name will be exposed in the workspace element via the atom:title element.

This service document represents both getRepositories() and getRepositoryInfo(repid). Each repository will be a workspace in the service document.

ATOM FEED DOCUMENTS

At any point where an ATOM document of type Feed is sent or returned, it must be a valid atom Feed document and conform to the guidelines below:

a. Updated will be the latest time the folder or its contents was updated. If unknown by the underlying repository, it should be the current time.
b. Author/name will be CMIS:createdBy
c. Title will be CMIS:name
d. Id will be URI of the resource
e. App:edited will be CMIS:lastModifiedDate
f. Link will be generated to return the uri of the feed

Paging of feeds will follow RFC 5005 <http://www.ietf.org/rfc/rfc5005.txt>.

ATOM ENTRY DOCUMENTS

At any point where an ATOM document of type Entry is sent or returned, it must be a valid atom Entry document and conform to the guidelines below:

a. Atom:Title will be best efforts by the repository. The repository should chose a property closest to Title.
b. Atom:Id will be the URI of the resource
c. App:edited will be CMIS:lastModifiedDate
d. Link with relation self will be CMIS:uri
e. Published will be CMIS:createdDate
f. Atom:author will be CMIS:creator
g. For content tags
    i. Documents with content
        1. Leverage the src attribute to point to the same link as cmis-stream
        2. The repository SHOULD populate the summary tag with text that at best efforts represents the documents. For example, an HTML table containing the properties and their values for simple feed readers
    ii. Other (Content-less document, Folder, Relationship, Type, etc) – best efforts at generating HTML text that represents the object. That text would normally go into the summary tag, but since there is no content, goes in the content tag.
g. For summary tag,
    i. If content src is specified, the summary SHOULD contain a text or html representation of the object.
h. Links will be used to provide URIs to CMIS functionality
    ii. CMIS Links may be omitted if the function is not allowed and that function would not show up on getAllowableActions.
    iii. Links may be omitted if the repository does not support that capability
h. All CMIS properties will be exposed in CMIS properties tag even if they are duplicated in an atom element

When POSTing an Atom Document, the atom fields take precedence over the CMIS property field for writeable properties. For example, atom:title will overwrite cmis:name

ATOM CATEGORIES

```
<?xml version="1.0" ?>
<app:categories
    xmlns:app="http://www.w3.org/2007/app"
    xmlns:atom="http://www.w3.org/2005/Atom"
    fixed="yes" scheme="http://example.com/cats/bag3">
  <atom:category term="animal" />
  <atom:category term="vegetable" />
  <atom:category term="mineral" />
</app:categories>
```

Inside the entry document:

```
<atom:category scheme="http://example.com/cats/b”g3" term="mineral"/>
```

The categorization functionality of ATOM will be repository specific. Specifically the repository may:

1. Internalize categories via some scheme and expose
2. Ignore the category tags

CMIS SEARCH DOCUMENT FORMAT

A CMIS search request will use the document format below. It will have a mime/type of application/cmis+xml;type=searchrequest. To execute the search, the search request document must be posted to the CMIS query collection.

Example:

```
<query xmlns=”http://www.cmis.org/CMIS/1.0”>
            <statement>SELECT * FROM document</statement>
            <searchAllVersions>false</searchAllVersions>
            <pageSize>0</pageSize>
            <skipCount>0</skipCount>
</query>
```

Common Exceptions

The following table defines the HTTP status codes that repositories will return for the various common exceptions defined in Part I of the CMIS specification.

CMIS Services Exception	**HTTP Status Code**

InvalidArgumentException	400
ConstraintViolationException	409
ObjectNotFoundException	404
PermissionDeniedException	401
OperationNotSupportedException	405

UpdateConflictException	409
RuntimeException	500
Successful DELETE TREE	204

Other Exceptions

CMIS Services Exception	HTTP Status Code
TypeNotFoundException	404
FilterNotValidException	400
StreamNotSupportedException	403
StorageException	500
OffsetException	400
NotInFolderException	404
ContentAlreadyExistsException	409
VersioningException	409
ContentStreamNotProvided	400

SERVICES

REPOSITORY SERVICES

getRepositoryInfo

Description	This service is used to retrieve information about the CMIS repository and the capabilities it supports.
Arguments	Headers: CMIS-repositoryId

	HTTP Arguments: repositoryId

This service will be exposed by the return of an Atom service document, as described in the Overview section.

Notes:

- RepositoryURI is not applicable for REST binding, as the service document contains the relevant URI for the repository
- rootFolderId is returned as a URI to the root folder's children which contains the Id

getTypes

Description	Returns the list of all types in the repository.
Arguments	Headers: CMIS-type (typeId), CMIS-includePropertyDefinitions (Boolean), CMIS –maxItems (Integer), CMIS-skipCount (Integer) HTTP Arguments: type, includePropertyDefinitions, maxItems, skipCount

This method is modeled as an Atom call by following a collection of collectionType "types" collection from the workspace. The inputs will be HTTP header tags on the GET request.

```
GET /cmis/main?types HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-type: 1234567
CMIS-returnPropertyDefinitions: false
CMIS-maxItems: 100
CMIS-skipCount: 0
```

Response: The CMIS feed document containing the types as entries:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:cmis=
        "http://www.cmis.org/CMIS/1.0">
…
</feed>
```

getTypeDefinition

Description	Gets the definition for specified type
Arguments	Headers: CMIS-includePropertyDefinitions (Boolean) HTTP Arguments: includePropertyDefinitions

This method is modeled as an Atom call by following a link for the relevant entry. The inputs will be HTTP header tags on the GET request.

```
GET /cmis/typ1 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn

<!-- type example -->
```

NAVIGATION SERVICES

getDescendants

Description	Gets the list of document and folder objects contained at one or more levels below the specified folder. Only the selected properties associated with each object are returned. The content stream is not returned. This will return the version of the documents in the folder specified by the user filing the documents into the folder. Typically and by default this will be the latest version.
Arguments	Headers: CMIS-childTypes (enumTypesOfFileableObjects), CMIS-filter (String), CMIS-depth (Integer), CMIS-includeAllowableActions (Boolean), CMIS-includeRelationships (enumIncludeRelationships), CMIS-folderByPath (String) HTTP Arguments: childTypes, filter, depth, includeAllowableActions, folderByPath

This method is modeled as an Atom call on a collection (Folder). The URI can be found through 'cmis-descendants' link type. The inputs will be HTTP header tags on the GET request.

If folderByPath is specified, then the descendants to be retrieved will be the one specified by the argument. This argument is only valid on the root-children or root-descendants collection.

```
GET /cmis/main?getDescendants HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-types: Any
CMIS-filter: *
CMIS-depth: 2
CMIS-includeAllowableActions: True
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<!-- see descendants example -->
```

Note: For descendants, the atom:entry will have nesting (i.e. contain other atom:entry), up to the specified depth.

getChildren

Description	Gets the list of document and folder objects contained one level below the specified folder. Only the selected properties associated with each object are returned. The content streams of documents are not returned. This will return the version of the documents in the folder specified by the user filing the documents into the folder. Typically and by default this will be the latest version.
Arguments	Headers: CMIS-types (enumTypesOfFileableObjects), CMIS-filter (String), CMIS –maxItems (Integer), CMIS-skipCount (Integer), CMIS-includeAllowableActions (Boolean), CMIS-includeRelationships (enumIncludeRelationships), CMIS-folderByPath HTTP Arguments: types, filter, maxItems, skipCount , includeAllowableActions, folderByPath

This method is modeled as an Atom call on a collection (Folder). The URI can be discovered through the 'cmis-children' link type. The inputs will be HTTP header tags on the GET request.

If folderByPath is specified, then the children to be retrieved will be the one specified by the argument. This argument is only valid on the root-children or root-descendants collection.

```
GET /cmis/main?getchildren HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-types: Any
CMIS-filter: *
CMIS-maxItems: 100
CMIS-skipCount: 0
CMIS-includeAllowableActions: True
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<!-- see children example -->
```

getFolderParent

Description	Returns the parent folder object(s) above the specified object.

Arguments	Headers: CMIS-filter (String), CMIS-returnToRoot (Boolean), CMIS-includeAllowableActions (Boolean), includeRelationships (enumIncludeRelationships) HTTP Arguments: filter, returnToRoot, includeAllowableActions

This method is modeled as an Atom call by following a link of type "cmis-folderparent". The inputs will be HTTP header tags on the GET request.

```
GET /cmis/main/myfolder?getfolderparent HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-filter: *
CMIS-returnToRoot: false
CMIS-includeAllowableActions: True
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<!-- see folderparent example -->
```

If more than one level of parents is requested, then CMIS will return a flat feed list of the parents in order with the closest parent being the first entry in the feed.

getObjectParents

Description	Returns the parent folders for the specified document
Arguments	Headers: CMIS-filter (String), CMIS-returnToRoot (Boolean), CMIS-includeAllowableActions (Boolean), includeRelationships (enumIncludeRelationships) HTTP Arguments: filter, returnToRoot, includeAllowableActions

This method is modeled as an Atom call by following a link of type "cmis-parents". The inputs will be HTTP header tags on the GET request.

```
GET /cmis/main/mydoc?parents HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-filter: *
CMIS-returnToRoot: false
CMIS-includeAllowableActions: True
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<!-- see parents example -->
```

Notes:

1. Return order is repository-specific
2. When returnToRoot is true and the document is filed folder A/B and folder A/B/C, the caller will be unable to distinguish whether the document is filed in A/B (because both A/B and A/B/C) will be returned as entries in the feed.

getCheckedOutDocuments

Description	Gets the list of documents that are checked out that the user has access to. Most likely this will be the set of documents checked out by the user. The content stream is not returned.
Arguments	Headers: CMIS-filter (String), CMIS-maxItems (Integer), CMIS-skipCount (Integer), CMIS-folderId (String), CMIS-includeAllowableActions (Boolean), includeRelationships (enumIncludeRelationships) HTTP Arguments: filter, maxItems, skipCount, folderId, includeAllowableActions

This method is modeled as an Atom call by opening a collection of type 'checkedout'. The inputs will be HTTP header tags on the GET request.

```
GET /cmis/main?checkedout HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-filter: *
CMIS-maxItems: 100
CMIS-skipCount: 0
CMIS-folderId: foo1
CMIS-includeAllowableActions: True
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<!-- see get children example -->
```

OBJECT SERVICES

createDocument

Description	Creates the object of the specified type
Arguments	Headers: CMIS-versioningState (enumVersioningState) HTTP Arguments: versioningState

This method follows the Atom Publishing model where the entry document, atom or cmis, is posted to the root or specific folder URI. For unfiled documents, post the document to the unfiled collection. The document will be created in the folder posted. If the document is posted to the root collection and a folder property is specified, then the repository will create the document in the specified folder.

If the content stream is specified on create, it should be base64 encoded in the atom entry.

A content stream must be specified regardless of versioningState. If not provided, ContentStreamNotProvided exception will be thrown.

```
POST /cmis/main HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Slug: My new story

<!-- atom entry -->
```

Response:

```
HTTP/1.1 201 Created
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
ETag: "abcdefghijkl1"
Location: http://example.org/cmis/atom04

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section for document. See "Atom Feed and Entry" section. -->
```

Notes:

1. The behavior is repository specific when a non-(atom/cmis)-entry document is posted to a folder URI . For example, the repository MAY auto-create a document with a specific type (document) the client could edit. The repository MAY also throw an exception.
2. Posting the content stream for a document is a separate operation, using the edit-media link in the response. Please See SetContentStream.
3. If an OID for an existing document is specified, then the document will be filed into the specified folder. See AddDocumentToFolder in the Multi-filing service.

createFolder

Description	Creates the object of the specified type

This method follows the Atom Publishing model where the entry document is posted to the feed.

```
POST /cmis/main HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Slug: CMIS Folder

<?xml version="1.0" encoding="utf-8"?>
<!-- ATOM entry section for document. See Section XYZF -->
```

Response:

```
HTTP/1.1 201 Created
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
ETag: "abcdefghijkl1"
Location: http://example.org/folders/cmis

<?xml version="1.0" encoding="utf-8"?>
<!-- ATOM entry section for document. See "Atom Feed and Entry" section -->
```

Note:

1. If the OID for an existing folder is specified, the action will be to Move the existing folder, NOT to Create a new folder. See moveObject.

createRelationship

Description	Creates the object of the specified type

This method follows the Atom Publishing model where the entry document is posted to the root or any other CMIS collection.

```
POST /cmis/main HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Slug: CMIS Relationship1

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section for relationship. See “Atom Feed and Entry” section -->
```

Response:

```
HTTP/1.1 201 Created
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
ETag: “abcdefghijkl1”
Location: http://example.org/folders/cmis

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section for relationship. See “Atom Feed and Entry” section -->
```

getAllowableActions

Description	Returns the list of allowable actions for a document, folder, or relationship object based on the current user’s context.
Arguments	

The client will follow a link of type ‘cmis-allowableactions’.

```
GET /cmis/atom03?getactions HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section for actions. See “Atom Feed and Entry” section -->
```

getProperties

Description	Returns the properties of object
Arguments	Headers: CMIS-filter, CMIS-returnVersion (enumReturnVersion) HTTP Arguments: filter, returnVersion Enum returnVersion: This, Latest, Major

This method is modeled as an Atom call by following a link in an entry element. The inputs will be HTTP header tags on the GET request.

```
GET /cmis/main/myfolder HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-filter: *
CMIS-returnVersion: true
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section. See “Atom Feed and Entry” section -->
```

getContentStream

Description	Returns the content stream for the document
Arguments	Headers: CMIS-returnVersion (enumReturnVersion), CMIS-offset (Integer), CMIS-length (Integer) HTTP Arguments: returnVersion, offset, length

This method is modeled as an Atom call by following the 'edit-media', or 'cmis-stream' link in an entry element. The inputs will be HTTP header tags on the GET request.

```
GET /media/atom03 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-returnVersion: true
```

At this point the repository will return the current content stream using HTTP mechanisms, including respecting the HTTP Content-Range parameters.

updateProperties

Description	Updates properties of the specified object.

This method follows the Atom Publishing model where the entry document is PUT at the metadata URL

```
PUT /atom03 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Slug: Atom-Powered Robots Run Amok -updated

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section. See "Atom Feed and Entry" section -->
```

If there, CMIS will ignore the system properties. Duplicated properties must be the same (entry/title and cmis:object/cmis:title). If duplicated properties in ATOM and CMIS are different, it is repository specific how to handle it.

Response:

```
HTTP/1.1 201 Created
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
ETag: "abcdefghijkl1"
Location: http://example.org/cmis/atom04

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section. See "Atom Feed and Entry" section -->
```

Note:

1. For all user-defined properties not specified in the request except system properties, they must be same after the update.

moveObject

Description	Moves specified folder or document to new location
Arguments	Headers: CMIS-removeFrom (String) HTTP Arguments: removeFrom

Post an entry doc to the new collection location.

Header CMIS-removeFrom: folderId.

Note: For repositories that do not support multi-filing, the item will always be removed from the previous folder, even if the header is not specified.

deleteObject

Description	Deletes specified object

The client must DELETE the URI of the resource

```
DELETE /atom03 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: xxxx/yyyy
Content-Length: nnn
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn
Location: http://example.org/cmis/main
```

Note:

1. If a document, this will delete the specified version only. To delete all documents in a version series, delete the URI specified by 'cmis-allversions'.

deleteTree

Description	Deletes specified folder and the entire sub-tree
Arguments	Headers: CMIS-continueOnFailure (Boolean), CMIS-unfileMultiFiledDocuments (enumUnfileNonfolderObjects) HTTP Arguments: continueOnFailure, unfileMultiFileDocuments

The client MUST delete the URI specified by 'cmis-descendants'.

```
DELETE /folder1 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: xxxx/yyyy
Content-Length: nnn
CMIS-continueOnFailure: false
CMIS-unfileMultiFiledDocuments: true
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn
Location: http://example.org/cmis/main
```

If something fails, then the content returned will be a feed of the items that failed to be deleted. Otherwise it will be empty.

setContentStream

Description	Sets (creates or replaces) the content stream for the version specified of a document object.

This method follows the Atom Publishing model where the media (content stream) is PUT at the edit-media or stream link.

```
PUT /edit-media/atom03 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: xxxx/yyyy
Content-Length: nnn
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
ETag: "abcdefghijkl1"
Location: http://example.org/edit/atom03
```

Because repositories MAY automatically create new Document Versions on a user's behalf, the Location returned may not match the URL of the PUT request.

deleteContentStream

Description	Deletes the content stream of the specified document Id. This does not delete properties. If there are other versions this does not affect them, their properties or content. This does not change the ID of the document.

The client should send DELETE for the edit-media or stream link.

```
DELETE /edit-media/atom03 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Location: http://example.org/edit/atom03
```

MULTI-FILING SERVICES

addDocumentToFolder

Description	Adds an existing document object to a folder. This may fail based on repository rules such as multi-filing not being supported, documents only being allowed to be filed once in any folder, etc.
Arguments	Headers: CMIS-removeFrom (String), CMIS-thisVersion (Boolean) HTTP Arguments: removeFrom, thisVersion

As described in createDocument, this is just a POST to the folder collection to which the document is being added, passing in the document's OID.

```
POST /cmis/folder1 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Slug: My story
CMIS-removeFrom: folder2

<?xml version="1.0" encoding="utf-8"?>
<!—Atom entry document including OID -->
```

Response:

```
HTTP/1.1 201 Created
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
ETag: "abcdefghijkl1"
Location: http://example.org/cmis/atom04

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section for document. See "Atom Feed and Entry" section -
->
```

Note: As described in createDocument and moveObject, use the optional "CMIS-removeFrom" property to move a document.

removeDocumentFromFolder

Description	Removes document from a folder. This does not delete the document and does not change the ID of the document.

To remove the document from all folders: post it to the Unfiled collection.

To remove the document from a particular folder: post an entry for the document to a folder in which you want the document to be filed, including the http header "CMIS-removeFrom" with the value set to the folder from which you want to unfile the document. (If the document is already in the folder to which you are posting, the document will be removed from the specified folder, but NOT doubly filed into the target folder.)

DISCOVERY SERVICES

query

Description	Queries the repository for folders and content based on properties or an optional full text string. Query returns version that matches the constraints of a content object and does not search relationship objects. The content stream is not returned as part of query.

The client must POST a CMIS Search Request document to the CMIS Query collection like the example below:

```
POST /cmis/main HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/cmisrequest+xml;type=query
Content-Length: nnn
Accept: application/atom+xml;type=feed

<?xml version="1.0"?>
<query xmlns=”http://www.cmis.org/CMIS/1.0”>
            <statement>object_id1</statement >
            <searchAllVersions>false</searchAllVersions>
            <pageSize>0</pageSize>
            <skipCount>0</skipCount>
</query>
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn
Location: http://www.cmis.org/CMIS/search-abcd123123

<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:cmis=
        “http://www.cmis.org/CMIS/1.0”>
      <title>Unfiled documents</title>
      <link href="http://example.org/cmis/main?unfiled"/>
      <updated>2003-12-13T18:30:02Z</updated>
      <author>
            <name>John Doe</name>
      </author>
<id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>
<!—if more items -->
      <link rel=”next” href="http://example.org/cmis/main?unfiled&n=100"/>
  <entry>
      <title>Atom-Powered Robots Run Amok</title>
            <link href="http://example.org/atom03"/>
            <id>id1</id>
```

```
            <updated>2003-12-13T18:30:02Z</updated>
            <link rel="cmis:getdocumentparents"
             href="http://example.org/atom03?getdocumentparents">
            <link rel="edit-media"
             href="http://example.org/media/atom03">
            <!– cmis:object for documents will be included here.
            Please see "Atom Feed and Entry Documents" section -->
      </entry>
</feed>
```

Note:

1. If the repository can persist the search, a Location header must be returned with a URI to the search results.
2. The search document can be posted to any CMIS collection, with the same search results. (To scope the search query to a specific folder / folder tree, specify an appropriate query statement.)

VERSIONING SERVICES

Checkout

Description	Create a private working copy of the object, copies the metadata and optionally content. It is up to the repository to determine if updates to the current version (not PWC) and prior versions are allowed if checked-out..
Output	Headers: CMIS-contentCopied (Boolean)

To perform a checkout of a document, the client must POST the desired document to the checkedOut collection. The repository will return the private working copy of the document if the checkout succeeds.

```
POST /atom03? checkedout HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==

<?xml version="1.0" encoding="utf-8"?><!—Atom entry document including OID --
>
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Location: http://www.cmis.org/atom05pwc

<?xml version="1.0" encoding="utf-8"?>
<!—Atom entry document will be returned -->
```

cancelCheckOut

Description	Removes the private working copy of the document object, allowing other documents in the version series to be checked out again..

The client must send a DELETE request on the private working copy URI.

```
DELETE /atom03?cancelcheckout HTTP/1.1
Host: example.org
```

```
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: xxxx/yyyy
Content-Length: nnn
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Location: http://example.org/atom03
```

checkIn

Description	Makes the working (private) copy the current version of the content.
Arguments	Headers: CMIS-checkinComment (String), CMIS-major (Boolean), CMIS-checkin (Boolean) HTTP Arguments: checkinComment, major, checkin

The client must PUT a new copy of the PWC with CMIS headers specifying checkin equal to 'true'. As such, checkin is treated like UPDATE with special semantics.

```
PUT /atom03?checkin HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: xxxx/yyyy
Content-Length: nnn
CMIS-checkinComment: Updating story
CMIS-major: true
CMIS-checkin: true
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn
Location: http://example.org/atom03
```

Note:

1. Content Streams must be specified before the checkin and not with the service.
2. To set a content stream, see SetContentStream.

getPropertiesOfLatestVersion

Description	Returns the properties of the latest version, or the latest major version, of the specified version series
Arguments	Headers: CMIS-filter (Boolean), CMIS- majorVersion (Boolean) HTTP Arguments: filter, majorVersion

This method is accessed by following the link of type 'cmis-latestversion' on any entry. The inputs will be HTTP header tags on the GET request.

```
GET /atom03?getPropertiesOfLatestVersion unfiled HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-filter: *
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=entry
Content-Length: nnn

<?xml version="1.0" encoding="utf-8"?>
<!—ATOM entry section. See "Atom Feed and Entry" section -->
```

getAllVersions

Description	Returns the list of all members of the version series for the specified document, sorted by CREATION_DATE descending.
Arguments	Headers: CMIS-filter (String) HTTP Arguments: filter

This method is accessed by following the link of type 'cmis-allversions' on the entry document. The inputs will be HTTP header tags on the GET request.

```
GET /atom03?getallversions unfiled HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
```

```
CMIS-filter: *
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<!-- feed example -->
```

deleteAllVersions

Description	Deletes all documents in the version series specified by a member's ObjectId.

This method will be invoked by calling DELETE on link 'cmis-allversions' resource.

```
DELETE /atom03 HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
```

At that point CMIS will return standard HTTP response.

RELATIONSHIP SERVICES

getRelationships

Description	Returns a list of relationships associated with the object.
Arguments	Headers: CMIS-relationshipType (String), CMIS-includeSubRelationshipTypes (Boolean), CMIS-filter (String), CMIS-maxItems (Integer), CMIS-skipCount (Integer), CMIS-direction (enumRelationshipDirection), CMIS-includeAllowableActions (Boolean) HTTP Arguments: relationshipType, includeSubRelationshipTypes, filter, maxResults, skipCount, direction (target, source, **all**) , includeAllowableActions

This method is accessed by following a link called 'cmis- relationships' on the entry element. The inputs will be HTTP header tags on the GET request.

```
GET /cmis/atom3/relationships-source HTTP/1.1
Host: example.org
User-Agent: Thingio/1.0
Authorization: Basic ZGFmZnk6c2VjZXJldA==
CMIS-relationshipType: relationship
CMIS-includeSubRelationshipTypes: false
CMIS-filter: *
CMIS-maxResults: 0
CMIS-skipCount: 0
CMIS-direction: target
CMIS-includeAllowableActions: True
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 12 July 2007 17:17:17 GMT
Content-Type: application/atom+xml;type=feed
Content-Length: nnn

<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:cmis=
    "http://www.cmis.org/CMIS/1.0">
  <title>Unfiled documents</title>
  <link href="http://example.org/atom03?source-relationships"/>
  <updated>2003-12-13T18:30:02Z</updated>
  <author>
         <name>John Doe</name>
  </author>
   <id>feed_id</id>
   <!—a set of atom entry documents for relationships -->
</feed>
```

SCHEMA DEFINITIONS

APP

```
<?xml version="1.0" encoding="UTF-8"?>
      <!--
            -*- rnc -*- RELAX NG Compact Syntax Grammar for the Atom Format
            Specification Version 11
      -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      elementFormDefault="qualified"
targetNamespace="http://www.w3.org/2007/app"
      xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:app="http://www.w3.org/2007/app"
      xmlns:cmis="http://www.cmis.org/2008/05"
      version="0.5">
      <xs:import namespace="http://www.w3.org/2005/Atom"
            schemaLocation="ATOM4CMIS.xsd" />
      <xs:import namespace="http://www.cmis.org/2008/05"
            schemaLocation="CMIS.xsd" />
      <xs:element name="service" type="app:appServiceType"></xs:element>
      <xs:complexType name="appServiceType">
            <xs:sequence>
                  <xs:element ref="atom:author" minOccurs="0"
maxOccurs="1"></xs:element>
                  <xs:element ref="app:workspace" minOccurs="1"
maxOccurs="unbounded"></xs:element>
                  <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                        namespace="##other" />
            </xs:sequence>
      </xs:complexType>
      <xs:element name="workspace" type="app:appWorkspaceType"></xs:element>
      <xs:complexType name="appWorkspaceType">
            <xs:sequence>
                  <xs:element ref="atom:title"></xs:element>
                  <xs:element ref="cmis:repositoryInfo" minOccurs="0"
                        maxOccurs="1"></xs:element>
                  <xs:element ref="app:collection" minOccurs="0"
maxOccurs="unbounded"></xs:element>
            </xs:sequence>
            <xs:attribute ref="cmis:id"></xs:attribute>
            <xs:attribute ref="cmis:repositoryRelationship"></xs:attribute>
      </xs:complexType>
      <xs:element name="collection"
type="app:appCollectionType"></xs:element>
      <xs:complexType name="appCollectionType">
            <xs:sequence>
                  <xs:element ref="atom:title"></xs:element>
                  <xs:element name="accept" type="xs:string" minOccurs="0"
                        maxOccurs="unbounded" />
                  <xs:element name="categories" type="app:appCategoriesType"
                        minOccurs="0" maxOccurs="unbounded" />
                  <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                        namespace="##other" />
```

```
            </xs:sequence>
            <xs:attribute ref="cmis:collectionType"></xs:attribute>
            <xs:attribute name="href" type="xs:anyURI"></xs:attribute>
      </xs:complexType>
      <xs:complexType name="appCategoriesType">
            <xs:sequence>
                  <xs:element name="category" type="app:appCategoryType"
                        minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="fixed" type="xs:boolean" />
      </xs:complexType>
      <xs:complexType name="appCategoryType">
            <xs:sequence>
            </xs:sequence>
            <xs:attribute name="scheme" type="xs:anyURI" />
            <xs:attribute name="term" type="xs:string" />
            <xs:attribute name="label" type="xs:string" />
      </xs:complexType>
</xs:schema>

<!-- EOF -->
```

ATOM

```
<?xml version="1.0" encoding="UTF-8"?>
      <!--
            -*- rnc -*- RELAX NG Compact Syntax Grammar for the Atom Format
            Specification Version 11
      -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      elementFormDefault="qualified"
targetNamespace="http://www.w3.org/2005/Atom"
      xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:xhtml="http://www.w3.org/1999/xhtml"
      xmlns:cmis="http://www.cmis.org/2008/05"
xmlns:xml="http://www.w3.org/XML/1998/namespace"
      xmlns:jaxb="http://java.sun.com/xml/ns/jaxb"
xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
      jaxb:extensionBindingPrefixes="xjc" jaxb:version="2.1" version="0.5">
      <xs:import namespace="http://www.w3.org/XML/1998/namespace"
            schemaLocation="xml.xsd" />
      <xs:import namespace="http://www.cmis.org/2008/05"
            schemaLocation="CMIS.xsd" />


      <!-- Common attributes -->
      <xs:attributeGroup name="atomCommonAttributes">
            <xs:attribute ref="xml:base" />
            <xs:attribute ref="xml:lang" />
            <xs:attributeGroup ref="atom:undefinedAttribute" />
      </xs:attributeGroup>
      <!-- Text Constructs -->
      <xs:attributeGroup name="atomPlainTextConstruct">
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
            <xs:attribute name="type">
                  <xs:simpleType>
                        <xs:restriction base="xs:token">
```

```
                        <xs:enumeration value="text" />
                        <xs:enumeration value="html" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:attributeGroup>
        <xs:group name="atomXHTMLTextConstruct">
            <xs:sequence>
                <xs:any minOccurs="0" maxOccurs="unbounded"
namespace="http://www.w3.org/1999/xhtml" />
            </xs:sequence>
        </xs:group>
        <xs:attributeGroup name="atomXHTMLTextConstruct">
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
            <xs:attribute name="type" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:token">
                        <xs:enumeration value="xhtml" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:attributeGroup>
        <xs:complexType name="atomTextConstruct" mixed="true">
            <xs:group minOccurs="0" ref="atom:atomXHTMLTextConstruct" />
            <xs:attribute name="type">
                <xs:simpleType>
                    <xs:restriction base="xs:token">
                        <xs:enumeration value="text" />
                        <xs:enumeration value="html" />
                        <xs:enumeration value="xhtml" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
        </xs:complexType>
        <!-- Person Construct -->
        <xs:complexType name="atomPersonConstruct">
            <xs:sequence>
                <xs:element ref="atom:name" minOccurs="0" maxOccurs="1" />
                <xs:element ref="atom:uri" minOccurs="0" maxOccurs="1" />
                <xs:element ref="atom:email" minOccurs="0" maxOccurs="1" />
                <xs:group ref="atom:extensionElement" minOccurs="0"
                    maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
        </xs:complexType>
        <xs:element name="name" type="xs:string" />
        <xs:element name="uri" type="xs:string" />
        <xs:element name="email" type="atom:atomEmailAddress" />
        <!-- Date Construct -->
        <xs:complexType name="atomDateConstruct">
            <xs:simpleContent>
                <xs:extension base="xs:dateTime">
                    <xs:attributeGroup ref="atom:atomCommonAttributes" />
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
```

```
	<!-- atom:feed -->
	<xs:element name="feed" type="atom:feedType"></xs:element>
	<xs:complexType name="feedType">
		<xs:sequence>
			<xs:element ref="atom:author" minOccurs="0"
maxOccurs="unbounded" />
			<xs:element ref="atom:category" minOccurs="0"
maxOccurs="unbounded" />
			<xs:element ref="atom:contributor" minOccurs="0"
				maxOccurs="1" />
			<xs:element ref="atom:generator" minOccurs="0"
maxOccurs="unbounded" />
			<xs:element ref="atom:icon" minOccurs="0"
maxOccurs="unbounded" />
			<xs:element ref="atom:id" minOccurs="1" maxOccurs="1" />
			<xs:element ref="atom:link" minOccurs="0"
maxOccurs="unbounded" />
			<xs:element ref="atom:logo" minOccurs="0" maxOccurs="1" />
			<xs:element ref="atom:rights" minOccurs="0" maxOccurs="1"
/>
			<xs:element ref="atom:subtitle" minOccurs="0" maxOccurs="1"
/>
			<xs:element ref="atom:title" minOccurs="1" maxOccurs="1" />
			<xs:element ref="atom:updated" minOccurs="1" maxOccurs="1"
/>
			<xs:element minOccurs="0" maxOccurs="unbounded"
ref="atom:entry" />

			<!-- Start Atom's extension here -->
			<xs:element ref="cmis:hasMoreItems" minOccurs="1"
				maxOccurs="1" />

			<!--  original atom extension element -->
			<xs:group ref="atom:extensionElement" />
		</xs:sequence>
		<xs:attributeGroup ref="atom:atomCommonAttributes" />
	</xs:complexType>
	<!-- atom:entry -->
	<xs:element name="entry" type="atom:entryType">
	</xs:element>
	<xs:complexType name="entryType">
		<xs:sequence>
			<xs:sequence>
				<xs:element ref="atom:author" minOccurs="0"
maxOccurs="unbounded" />
				<xs:element ref="atom:category" minOccurs="0"
maxOccurs="unbounded" />
				<xs:element ref="atom:content" minOccurs="0"
maxOccurs="1" />
				<xs:element ref="atom:contributor" minOccurs="0"
					maxOccurs="1" />
				<xs:element ref="atom:id" minOccurs="1" maxOccurs="1"
/>
				<xs:element ref="atom:link" minOccurs="0"
maxOccurs="unbounded" />
				<xs:element ref="atom:published" minOccurs="0"
maxOccurs="1" />
```

```
                        <xs:element ref="atom:rights" minOccurs="0"
maxOccurs="1" />
                        <xs:element ref="atom:source" minOccurs="0"
maxOccurs="1" />
                        <xs:element ref="atom:summary" minOccurs="0"
maxOccurs="1" />
                        <xs:element ref="atom:title" minOccurs="1"
maxOccurs="1" />
                        <xs:element ref="atom:updated" minOccurs="1"
maxOccurs="1" />


                         <!-- CMIS type, optional if not CMIS  -->
                        <xs:choice minOccurs="0" maxOccurs="1">
                              <xs:choice minOccurs="0" maxOccurs="1">
                                    <xs:annotation>
                                          <xs:appinfo>
                                                <jaxb:property name="type" />
                                          </xs:appinfo>
                                    </xs:annotation>
                                    <xs:element ref="cmis:documentType" />
                                    <xs:element ref="cmis:folderType" />
                                    <xs:element ref="cmis:policyType" />
                                    <xs:element ref="cmis:relationshipType"
/>
                              </xs:choice>

                              <!-- CMIS object, optional if not CMIS  -->
                              <xs:element ref="cmis:object" minOccurs="0"
maxOccurs="1" />
                        </xs:choice>

                        <!-- This is necessary for nested entries such as
descendants -->
                        <xs:element ref="atom:entry" minOccurs="0"
maxOccurs="unbounded" />

                        <!-- syntactic sugar -->
                        <xs:element ref="cmis:terminator" minOccurs="1"
                              maxOccurs="1" />


                        <!-- Normal ATOM extension element -->
                        <xs:group ref="atom:extensionElement" />
                  </xs:sequence>
            </xs:sequence>
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
      </xs:complexType>

      <!-- atom:content -->
      <xs:attributeGroup name="atomInlineTextConstruct">
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
            <xs:attribute name="type">
                  <xs:simpleType>
                        <xs:restriction base="xs:token">
                              <xs:enumeration value="text" />
                              <xs:enumeration value="html" />
```

```
                        </xs:restriction>
                  </xs:simpleType>
            </xs:attribute>
      </xs:attributeGroup>
      <xs:group name="atomInlineOtherConstruct">
            <xs:sequence>
                  <xs:group minOccurs="0" maxOccurs="unbounded"
ref="atom:anyElement" />
            </xs:sequence>
      </xs:group>
      <xs:attributeGroup name="atomInlineOtherConstruct">
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
            <xs:attribute name="type">
                  <xs:simpleType>
                        <xs:union memberTypes="atom:atomMediaType">
                              <xs:simpleType>
                                    <xs:restriction base="xs:token">
                                          <xs:enumeration value="xhtml" />
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:union>
                  </xs:simpleType>
            </xs:attribute>
      </xs:attributeGroup>
      <xs:attributeGroup name="atomOutOfLineConstruct">
            <xs:attributeGroup ref="atom:atomCommonAttributes" />
            <xs:attribute name="type" type="atom:atomMediaType" />
            <xs:attribute name="src" use="required" />
      </xs:attributeGroup>
      <xs:element name="content">
            <xs:complexType mixed="true">
                  <xs:group minOccurs="0" ref="atom:atomInlineOtherConstruct"
/>
                  <xs:attribute name="type">
                        <xs:simpleType>
                              <xs:union memberTypes="atom:atomMediaType">
                                    <xs:simpleType>
                                          <xs:restriction base="xs:token">
                                                <xs:enumeration value="text"
/>
                                                <xs:enumeration value="html"
/>
                                          </xs:restriction>
                                    </xs:simpleType>
                                    <xs:simpleType>
                                          <xs:union
memberTypes="atom:atomMediaType">
                                                <xs:simpleType>
                                                      <xs:restriction
base="xs:token">
                                                            <xs:enumeration
value="xhtml" />
                                                      </xs:restriction>
                                                </xs:simpleType>
                                          </xs:union>
                                    </xs:simpleType>
                              </xs:union>
```

```
                        </xs:simpleType>
                  </xs:attribute>
                  <xs:attributeGroup ref="atom:atomCommonAttributes" />
                  <xs:attribute name="src" />
            </xs:complexType>
      </xs:element>
      <!-- atom:author -->
      <xs:element name="author" type="atom:atomPersonConstruct" />
      <!-- atom:category -->
      <xs:element name="category">
            <xs:complexType>
                  <xs:complexContent>
                        <xs:extension base="atom:undefinedContent">
                              <xs:attributeGroup
ref="atom:atomCommonAttributes" />
                              <xs:attribute name="term" use="required" />
                              <xs:attribute name="scheme" />
                              <xs:attribute name="label" />
                        </xs:extension>
                  </xs:complexContent>
            </xs:complexType>
      </xs:element>
      <!-- atom:contributor -->
      <xs:element name="contributor" type="atom:atomPersonConstruct" />
      <!-- atom:generator -->
      <xs:element name="generator">
            <xs:complexType mixed="true">
                  <xs:attributeGroup ref="atom:atomCommonAttributes" />
                  <xs:attribute name="uri" />
                  <xs:attribute name="version" />
            </xs:complexType>
      </xs:element>
      <!-- atom:icon -->
      <xs:element name="icon">
            <xs:complexType mixed="true">
                  <xs:attributeGroup ref="atom:atomCommonAttributes" />
            </xs:complexType>
      </xs:element>
      <!-- atom:id -->
      <xs:element name="id">
            <xs:complexType mixed="true">
                  <xs:attributeGroup ref="atom:atomCommonAttributes" />
            </xs:complexType>
      </xs:element>
      <!-- atom:logo -->
      <xs:element name="logo">
            <xs:complexType mixed="true">
                  <xs:attributeGroup ref="atom:atomCommonAttributes" />
            </xs:complexType>
      </xs:element>
      <!-- atom:link -->
      <xs:element name="link">
            <xs:annotation>
                  <xs:documentation>
                        The "atom:link" element defines a reference from an
                        entry or feed to a Web resource. This specification
                        assigns no meaning to the content (if any) of this
```

```
                        element.
                  </xs:documentation>
            </xs:annotation>

            <xs:complexType>
                  <xs:complexContent>
                        <xs:extension base="atom:undefinedContent">
                              <xs:attributeGroup
ref="atom:atomCommonAttributes" />
                              <xs:attribute name="href" use="required" />
                              <xs:attribute name="rel"></xs:attribute>
                              <xs:attribute name="type"
type="atom:atomMediaType" />
                              <xs:attribute name="hreflang"
type="atom:atomLanguageTag" />
                              <xs:attribute name="title" />
                              <xs:attribute name="length" />
                              <xs:attribute ref="cmis:id" use="optional" />
                        </xs:extension>
                  </xs:complexContent>
            </xs:complexType>
      </xs:element>
      <!-- atom:published -->
      <xs:element name="published" type="atom:atomDateConstruct" />
      <!-- atom:rights -->
      <xs:element name="rights" type="atom:atomTextConstruct" />
      <!-- atom:source -->
      <xs:element name="source">
            <xs:annotation>
                  <xs:documentation>
                        atom:source is used to preserve metadata of a feed
when
                        an entry is copied from a feed to another feed.
                  </xs:documentation>
            </xs:annotation>
            <xs:complexType>
                  <xs:choice minOccurs="0" maxOccurs="unbounded">
                        <xs:element ref="atom:author" />
                        <xs:element ref="atom:category" />
                        <xs:element ref="atom:contributor" />
                        <xs:element ref="atom:generator" />
                        <xs:element ref="atom:icon" />
                        <xs:element ref="atom:id" />
                        <xs:element ref="atom:link" />
                        <xs:element ref="atom:logo" />
                        <xs:element ref="atom:rights" />
                        <xs:element ref="atom:subtitle" />
                        <xs:element ref="atom:title" />
                        <xs:element ref="atom:updated" />
                        <xs:group ref="atom:extensionElement" />
                  </xs:choice>
                  <xs:attributeGroup ref="atom:atomCommonAttributes" />
            </xs:complexType>
      </xs:element>
      <!-- atom:subtitle -->
      <xs:element name="subtitle" type="atom:atomTextConstruct" />
      <!-- atom:summary -->
```

```
	<xs:element name="summary" type="atom:atomTextConstruct" />
	<!-- atom:title -->
	<xs:element name="title" type="atom:atomTextConstruct">
		<xs:annotation>
			<xs:documentation>
				The "atom:title" element is a Text construct that
				conveys a human- readable title for an entry or feed.
				atomTitle = element atom:title { atomTextConstruct }.
			</xs:documentation>
		</xs:annotation>
	</xs:element>
	<!-- atom:updated -->
	<xs:element name="updated" type="atom:atomDateConstruct">
		<xs:annotation>
			<xs:documentation>
				The "atom:updated" element is a Date construct
				indicating the most recent instant in time when an
entry
				or feed was modified in a way the publisher considers
				significant. Therefore, not all modifications
				necessarily result in a changed atom:updated value.
				atomUpdated = element atom:updated
{ atomDateConstruct
				}. Publishers MAY change the value of this element
over
				time.
			</xs:documentation>
		</xs:annotation>
	</xs:element>
	<!-- Low-level simple types -->
	<xs:simpleType name="atomNCName">
		<xs:restriction base="xs:string">
			<xs:minLength value="1" />
			<xs:pattern value="[^:]*" />
		</xs:restriction>
	</xs:simpleType>
	<!-- Whatever a media type is, it contains at least one slash -->
	<xs:simpleType name="atomMediaType">
		<xs:restriction base="xs:string">
			<xs:pattern value=".+/.+" />
		</xs:restriction>
	</xs:simpleType>
	<!-- As defined in RFC 3066 -->
	<xs:simpleType name="atomLanguageTag">
		<xs:restriction base="xs:string">
			<xs:pattern value="[A-Za-z]{1,8}(-[A-Za-z0-9]{1,8})*" />
		</xs:restriction>
	</xs:simpleType>
	<!--
		Unconstrained; it's not entirely clear how IRI fit into
xsd:anyURI so
		let's not try to constrain it here
	-->
	<!-- Whatever an email address is, it contains at least one @ -->
	<xs:simpleType name="atomEmailAddress">
		<xs:restriction base="xs:string">
			<xs:pattern value=".+@.+" />
```

```
            </xs:restriction>
      </xs:simpleType>
      <!-- Simple Extension -->
      <xs:group name="extensionElement">
            <xs:sequence>
                  <xs:any namespace="##other" processContents="lax"
minOccurs="0"
                        maxOccurs="unbounded">
                        <xs:annotation>
                              <xs:appinfo>
                                    <jaxb:property name='anyOther' />
                              </xs:appinfo>
                        </xs:annotation>
                  </xs:any>
                  <xs:any namespace="##local" processContents="lax"
minOccurs="0"
                        maxOccurs="unbounded">
                        <xs:annotation>
                              <xs:appinfo>
                                    <jaxb:property name='anyLocal' />
                              </xs:appinfo>
                        </xs:annotation>
                  </xs:any>
            </xs:sequence>
      </xs:group>
      <xs:attributeGroup name="undefinedAttribute">
            <xs:anyAttribute namespace="##other" processContents="lax" />
      </xs:attributeGroup>
      <xs:complexType name="undefinedContent" mixed="true">
            <xs:group minOccurs="0" maxOccurs="unbounded"
ref="atom:anyForeignElement" />
      </xs:complexType>
      <xs:group name="anyElement">
            <xs:sequence>
                  <xs:any processContents="lax" />
            </xs:sequence>
      </xs:group>
      <xs:group name="anyForeignElement">
            <xs:choice>
                  <xs:any namespace="##other" processContents="lax" />
                  <xs:any namespace="##local" processContents="lax" />
            </xs:choice>
      </xs:group>
      <!-- XHTML -->
      <xs:group name="anyXHTML">
            <xs:sequence>
                  <xs:any namespace="http://www.w3.org/1999/xhtml"
                        processContents="lax" />
            </xs:sequence>
      </xs:group>
</xs:schema>

<!-- EOF -->
```

CMIS

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
	<!--
		Common CMIS XSD for version 0.5
	-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
	elementFormDefault="qualified"
targetNamespace="http://www.cmis.org/2008/05"
	xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:xhtml="http://www.w3.org/1999/xhtml"
	xmlns:jaxb="http://java.sun.com/xml/ns/jaxb"
xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
	jaxb:extensionBindingPrefixes="xjc" jaxb:version="2.1"
	xmlns:cmis="http://www.cmis.org/2008/05" version="0.50">
	<xs:import namespace="http://www.w3.org/XML/1998/namespace"
		schemaLocation="xml.xsd" />

	<!--  enums -->
	<xs:simpleType name="enumDecimalPrecision">
		<xs:restriction base="xs:integer">
			<xs:enumeration value="32" />
			<xs:enumeration value="64" />
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="enumContentStreamAllowed">
		<xs:restriction base="xs:string">
			<xs:enumeration value="notallowed" />
			<xs:enumeration value="allowed" />
			<xs:enumeration value="required" />
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="enumCardinality">
		<xs:restriction base="xs:string">
			<xs:enumeration value="single" />
			<xs:enumeration value="multi" />
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="enumUpdateability">
		<xs:restriction base="xs:string">
			<xs:enumeration value="readonly" />
			<xs:enumeration value="readwrite" />
			<xs:enumeration value="whencheckedout" />
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="enumPropertyType">
		<xs:restriction base="xs:string">
			<xs:enumeration value="boolean" />
			<xs:enumeration value="id" />
			<xs:enumeration value="integer" />
			<xs:enumeration value="datetime" />
			<xs:enumeration value="decimal" />
			<xs:enumeration value="html" />
			<xs:enumeration value="string" />
			<xs:enumeration value="uri" />
			<xs:enumeration value="xml" />
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="enumCollectionType">
		<xs:restriction base="xs:string">
```

```
            <xs:enumeration value="root-children" />
            <xs:enumeration value="root-descendants" />
            <xs:enumeration value="unfiled" />
            <xs:enumeration value="checkedout" />
            <xs:enumeration value="types-children" />
            <xs:enumeration value="types-descendants" />
            <xs:enumeration value="query" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumObjectType">
      <xs:restriction base="xs:string">
            <xs:enumeration value="document" />
            <xs:enumeration value="folder" />
            <xs:enumeration value="relationship" />
            <xs:enumeration value="policy" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumCapabilityQuery">
      <xs:restriction base="xs:string">
            <xs:enumeration value="none" />
            <xs:enumeration value="metadataonly" />
            <xs:enumeration value="fulltextonly" />
            <xs:enumeration value="both" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumCapabilityJoin">
      <xs:restriction base="xs:string">
            <xs:enumeration value="nojoin" />
            <xs:enumeration value="inneronly" />
            <xs:enumeration value="innerandouter" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumCapabilityFullText">
      <xs:restriction base="xs:string">
            <xs:enumeration value="none" />
            <xs:enumeration value="fulltextonly" />
            <xs:enumeration value="fulltextandstructured" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumRepositoryRelationship">
      <xs:restriction base="xs:string">
            <xs:enumeration value="self" />
            <xs:enumeration value="replica" />
            <xs:enumeration value="peer" />
            <xs:enumeration value="parent" />
            <xs:enumeration value="child" />
            <xs:enumeration value="archive" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumTypesOfFileableObjects">
      <xs:restriction base="xs:string">
            <xs:enumeration value="documents" />
            <xs:enumeration value="folders" />
            <xs:enumeration value="policies" />
            <xs:enumeration value="any" />
      </xs:restriction>
</xs:simpleType>
```

```
<xs:simpleType name="enumVersioningState">
      <xs:restriction base="xs:string">
            <xs:enumeration value="checkedout" />
            <xs:enumeration value="minor" />
            <xs:enumeration value="major" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumReturnVersion">
      <xs:restriction base="xs:string">
            <xs:enumeration value="this" />
            <xs:enumeration value="latest" />
            <xs:enumeration value="latestmajor" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumUnfileNonfolderObjects">
      <xs:restriction base="xs:string">
            <xs:enumeration value="unfile" />
            <xs:enumeration value="deletesinglefiled" />
            <xs:enumeration value="delete" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumRelationshipDirection">
      <xs:restriction base="xs:string">
            <xs:enumeration value="source" />
            <xs:enumeration value="target" />
            <xs:enumeration value="both" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumIncludeRelationships">
      <xs:restriction base="xs:string">
            <xs:enumeration value="none" />
            <xs:enumeration value="source" />
            <xs:enumeration value="target" />
            <xs:enumeration value="both" />
      </xs:restriction>
</xs:simpleType>

<!--  properties in CMIS -->
<xs:simpleType name="enumPropertiesBase">
      <xs:restriction base="xs:string">
            <xs:enumeration value="ObjectId" />
            <xs:enumeration value="Uri" />
            <xs:enumeration value="ObjectTypeId" />
            <xs:enumeration value="CreatedBy" />
            <xs:enumeration value="CreationDate" />
            <xs:enumeration value="LastModifiedBy" />
            <xs:enumeration value="LastModificationDate" />
            <xs:enumeration value="ChangeToken" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumPropertiesDocument">
      <xs:restriction base="xs:string">
            <xs:enumeration value="ObjectId" />
            <xs:enumeration value="Uri" />
            <xs:enumeration value="ObjectTypeId" />
            <xs:enumeration value="CreatedBy" />
            <xs:enumeration value="CreationDate" />
```

```
            <xs:enumeration value="LastModifiedBy" />
            <xs:enumeration value="LastModificationDate" />
            <xs:enumeration value="ChangeToken" />
            <xs:enumeration value="IsImmutable" />
            <xs:enumeration value="isLatestVersion" />
            <xs:enumeration value="IsMajorVersion" />
            <xs:enumeration value="IsLatestMajorVersion" />
            <xs:enumeration value="VersionLabel" />
            <xs:enumeration value="VersionSeriesId" />
            <xs:enumeration value="IsVersionSeriesCheckedOut" />
            <xs:enumeration value="VersionSeriesCheckedOutBy" />
            <xs:enumeration value="VersionSeriesCheckedOutId" />
            <xs:enumeration value="CheckinComment" />
            <xs:enumeration value="ContentStreamAllowed" />
            <xs:enumeration value="ContentStreamLength" />
            <xs:enumeration value="ContentStreamMimeType" />
            <xs:enumeration value="ContentStreamFilename" />
            <xs:enumeration value="ContentStreamUri" />
            <xs:enumeration value="" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumPropertiesFolder">
      <xs:restriction base="xs:string">
            <xs:enumeration value="ObjectId" />
            <xs:enumeration value="Uri" />
            <xs:enumeration value="ObjectTypeId" />
            <xs:enumeration value="CreatedBy" />
            <xs:enumeration value="CreationDate" />
            <xs:enumeration value="LastModifiedBy" />
            <xs:enumeration value="LastModificationDate" />
            <xs:enumeration value="ChangeToken" />
            <xs:enumeration value="ParentId" />
            <xs:enumeration value="AllowedChildObjectTypeIds" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumPropertiesRelationship">
      <xs:restriction base="xs:string">
            <xs:enumeration value="ObjectId" />
            <xs:enumeration value="Uri" />
            <xs:enumeration value="ObjectTypeId" />
            <xs:enumeration value="CreatedBy" />
            <xs:enumeration value="CreationDate" />
            <xs:enumeration value="LastModifiedBy" />
            <xs:enumeration value="LastModificationDate" />
            <xs:enumeration value="ChangeToken" />
            <xs:enumeration value="SourceId" />
            <xs:enumeration value="TargetId" />
      </xs:restriction>
</xs:simpleType>
<xs:simpleType name="enumPropertiesPolicy">
      <xs:restriction base="xs:string">
            <xs:enumeration value="ObjectId" />
            <xs:enumeration value="Uri" />
            <xs:enumeration value="ObjectTypeId" />
            <xs:enumeration value="CreatedBy" />
            <xs:enumeration value="CreationDate" />
            <xs:enumeration value="LastModifiedBy" />
```

```
                <xs:enumeration value="LastModificationDate" />
                <xs:enumeration value="ChangeToken" />
                <xs:enumeration value="PolicyName" />
                <xs:enumeration value="PolicyText" />
          </xs:restriction>
    </xs:simpleType>

    <!--  CMIS Rest Arguments -->
    <xs:simpleType name="enumRestArguments">
          <xs:restriction base="xs:string">
                <xs:enumeration value="childTypes" />
                <xs:enumeration value="continueOnFailure" />
                <xs:enumeration value="depth" />
                <xs:enumeration value="direction" />
                <xs:enumeration value="filter" />
                <xs:enumeration value="folderByPath" />
                <xs:enumeration value="includeAllowableActions" />
                <xs:enumeration value="includePropertyDefinitions" />
                <xs:enumeration value="includeRelationships" />
                <xs:enumeration value="includeSubrelationshipTypes" />
                <xs:enumeration value="length" />
                <xs:enumeration value="majorVersion" />
                <xs:enumeration value="maxItems" />
                <xs:enumeration value="offset" />
                <xs:enumeration value="removeFrom" />
                <xs:enumeration value="relationshipType" />
                <xs:enumeration value="repositoryId" />
                <xs:enumeration value="returnToRoot" />
                <xs:enumeration value="returnVersion" />
                <xs:enumeration value="skipCount" />
                <xs:enumeration value="thisVersion" />
                <xs:enumeration value="typeId" />
                <xs:enumeration value="types" />
                <xs:enumeration value="unfileMultiFiledDocuments" />
                <xs:enumeration value="versioningState" />
          </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="enumRestOutputHeaders">
          <xs:restriction base="xs:string">
                <xs:enumeration value="contentCopied" />
          </xs:restriction>
    </xs:simpleType>
    <xs:attributeGroup name="cmisUndefinedAttribute">
          <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:attributeGroup>


<!--getRepositories Operation-->
    <xs:complexType name="cmisRepositoryEntryType">
          <xs:sequence>
                <xs:element name="repositoryID" type="xs:string"
                      minOccurs="1" maxOccurs="1" />
                <xs:element name="repositoryName" type="xs:string"
                      minOccurs="1" maxOccurs="1" />
                <xs:element name="repositoryURI" type="xs:anyURI"
                      minOccurs="1" maxOccurs="1" />
```

```
                  <xs:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
                        processContents="lax" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>

      <!--  Atom & APP -->
      <xs:attribute name="id" type="xs:string" />
      <xs:attribute name="href" type="xs:anyURI" />
      <xs:attribute name="repositoryRelationship"
type="cmis:enumRepositoryRelationship" />
      <xs:attribute name="collectionType" type="cmis:enumCollectionType" />
      <xs:element name="hasMoreItems" type="xs:boolean" />
      <xs:element name="repositoryInfo" type="cmis:cmisRepositoryInfoType" />

      <!--  main cmis object -->
      <xs:complexType name="cmisObjectType">
            <xs:sequence>
                  <xs:element name="properties"
type="cmis:cmisPropertiesType"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element ref="cmis:allowableActions" minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="relationship" type="cmis:cmisObjectType"
                        minOccurs="0" maxOccurs="unbounded" />
                  <xs:element name="child" type="cmis:cmisObjectType"
                        minOccurs="0" maxOccurs="unbounded" />
                  <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                        namespace="##other" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>
      <xs:complexType name="objectTreeCollectionType">
            <xs:sequence>
                  <xs:element name="object" type="cmis:cmisObjectType"
                        minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>

      <!--  anyother tag -->
      <xs:complexType name="cmisAnyXml">
            <xs:sequence>
                  <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                        namespace="##other" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>

      <!-- Entry tag for tree Type -->
      <xs:element name="object" type="cmis:cmisObjectType" />

      <!--  separator object -->
      <xs:element name="terminator" type="xs:string" nillable="true" />
```

```
	<!--  type and type sub group -->
	<xs:element name="type" type="cmis:cmisTypeDefinitionType">
		<!--
			<xs:annotation> <xs:appinfo> <jaxb:property
			generateElementProperty="false" /> </xs:appinfo>
</xs:annotation>
		-->
	</xs:element>
	<xs:element name="documentType"
type="cmis:cmisTypeDocumentDefinitionType"
		substitutionGroup="cmis:type" />
	<xs:element name="folderType" type="cmis:cmisTypeFolderDefinitionType"
		substitutionGroup="cmis:type" />
	<xs:element name="relationshipType"
type="cmis:cmisTypeRelationshipDefinitionType"
		substitutionGroup="cmis:type" />
	<xs:element name="policyType" type="cmis:cmisTypePolicyDefinitionType"
		substitutionGroup="cmis:type" />


	<!-- property bag -->
	<xs:attribute name="key" type="xs:string" />
	<xs:attribute name="index" type="xs:integer" />
	<xs:attribute name="name" type="xs:string" />
	<xs:attribute name="propertyType" type="cmis:enumPropertyType" />
	<xs:complexType name="cmisPropertiesType">
		<xs:sequence>
			<xs:choice minOccurs="0" maxOccurs="unbounded">
				<xs:annotation>
					<xs:appinfo>
						<jaxb:property name="property" />
					</xs:appinfo>
				</xs:annotation>
				<xs:element ref="cmis:propertyBoolean" />
				<xs:element ref="cmis:propertyId" />
				<xs:element ref="cmis:propertyInteger" />
				<xs:element ref="cmis:propertyDateTime" />
				<xs:element ref="cmis:propertyDecimal" />
				<xs:element ref="cmis:propertyHtml" />
				<xs:element ref="cmis:propertyString" />
				<xs:element ref="cmis:propertyUri" />
				<xs:element ref="cmis:propertyXml" />
			</xs:choice>
			<xs:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
				processContents="lax" />
		</xs:sequence>
		<xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
	</xs:complexType>

	<!--  sub group -->
	<xs:element name="property" type="cmis:cmisProperty"
		nillable="true" />
	<xs:element name="propertyBoolean" type="cmis:cmisPropertyBoolean"
		substitutionGroup="cmis:property" nillable="true" />
	<xs:element name="propertyId" type="cmis:cmisPropertyId"
```

```
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyInteger" type="cmis:cmisPropertyInteger"
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyDateTime" type="cmis:cmisPropertyDateTime"
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyDecimal" type="cmis:cmisPropertyDecimal"
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyHtml" type="cmis:cmisPropertyHtml"
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyString" type="cmis:cmisPropertyString"
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyUri" type="cmis:cmisPropertyUri"
            substitutionGroup="cmis:property" nillable="true" />
      <xs:element name="propertyXml" type="cmis:cmisPropertyXml"
            substitutionGroup="cmis:property" nillable="true" />

      <!-- start the prop definitions -->
      <xs:complexType name="cmisProperty">
            <xs:attribute ref="cmis:name" use="required" />
            <xs:attribute ref="cmis:index" use="optional" />
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>
      <xs:complexType name="cmisPropertyBoolean">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:boolean" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="boolean" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyId">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:string" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="id" />
                        <xs:attribute ref="cmis:href" use="optional" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyInteger">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:integer" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="integer" />
                  </xs:extension>
```

```
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyDateTime">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:dateTime" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="datetime" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyDecimal">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:decimal" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="decimal" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyHtml">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                                    namespace="##other" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="html" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyString">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:string" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="string" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyUri">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:anyURI" />
```

```
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="uri" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyXml">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisProperty">
                        <xs:sequence>
                              <xs:any minOccurs="1" maxOccurs="unbounded"
processContents="lax"
                                    namespace="##other" />
                        </xs:sequence>
                        <xs:attribute ref="cmis:propertyType" use="optional"
                              default="xml" />
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>


  <!--Content Stream-->
      <xs:complexType name="cmisContentStreamType">
            <xs:sequence>
                  <xs:element name="length" type="xs:integer" />
                  <xs:element name="mimeType" type="xs:string" minOccurs="0"
/>
                  <xs:element name="filename" type="xs:string" minOccurs="0"
/>
                  <xs:element name="uri" type="xs:anyURI" minOccurs="0" />
                  <xs:element name="stream" type="xs:base64Binary"
                        xmime:expectedContentTypes="application/octet-stream"
xmlns:xmime="http://www.w3.org/2005/05/xmlmime" />
                  <xs:any namespace="##other" processContents="lax"
minOccurs="0"
                        maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>


      <!--  allowable actions -->
      <xs:complexType name="cmisAllowableActionsType">
            <xs:sequence>
                  <xs:element name="parentId" type="xs:string" minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="parentUrl" type="xs:string" minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="canDelete" type="xs:boolean"
minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="canUpdateProperties" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetProperties" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetRelationships" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
```

```
                  <xs:element name="canGetParents" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetFolderParent" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetDescendants" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canMove" type="xs:boolean" minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="canDeleteVersion" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canDeleteContent" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCheckout" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCancelCheckout" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCheckin" type="xs:boolean"
minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="canSetContent" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetAllVersions" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canAddToFolder" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canRemoveFromFolder" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canViewContent" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canAddPolicy" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetAppliedPolicies" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canRemovePolicy" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canGetChildren" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCreateDocument" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCreateFolder" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCreateRelationship" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canCreatePolicy" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="canDeleteTree" type="xs:boolean"
                        minOccurs="0" maxOccurs="1" />
                  <xs:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
                        processContents="lax" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>

      <!--  main allowable actions element -->
      <xs:element name="allowableActions"
type="cmis:cmisAllowableActionsType" />
```

```
	<!-- subgroup -->
	<xs:element name="choice" type="cmis:cmisChoiceType" />
	<xs:element name="choiceBoolean" type="cmis:cmisChoiceBooleanType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceId" type="cmis:cmisChoiceIdType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceInteger" type="cmis:cmisChoiceIntegerType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceDateTime" type="cmis:cmisChoiceDateTimeType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceDecimal" type="cmis:cmisChoiceDecimalType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceHtml" type="cmis:cmisChoiceHtmlType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceString" type="cmis:cmisChoiceStringType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceUri" type="cmis:cmisChoiceUriType"
		substitutionGroup="cmis:choice" />
	<xs:element name="choiceXml" type="cmis:cmisChoiceXmlType"
		substitutionGroup="cmis:choice" />

	<!-- type for choices -->
	<xs:complexType name="cmisChoiceType" abstract="true">
		<xs:sequence>
			<xs:element ref="cmis:choice" minOccurs="0"
maxOccurs="unbounded" />
		</xs:sequence>
		<xs:attribute ref="cmis:index" use="optional" />
		<xs:attribute ref="cmis:key" use="optional" />
		<xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
	</xs:complexType>

	<!-- do the property type specific choice entry -->
	<xs:complexType name="cmisChoiceBooleanType">
		<xs:complexContent>
			<xs:extension base="cmis:cmisChoiceType">
				<xs:sequence>
					<xs:element minOccurs="0" name="value"
type="xs:boolean" />
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="cmisChoiceIdType">
		<xs:complexContent>
			<xs:extension base="cmis:cmisChoiceType">
				<xs:sequence>
					<xs:element minOccurs="0" name="value"
type="xs:string" />
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="cmisChoiceIntegerType">
		<xs:complexContent>
```

```
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:integer" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisChoiceDateTimeType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:dateTime" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisChoiceDecimalType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:decimal" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisChoiceHtmlType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                                    namespace="##other" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisChoiceStringType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:string" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisChoiceUriType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:element minOccurs="0" name="value"
type="xs:anyURI" />
                        </xs:sequence>
                  </xs:extension>
```

```
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisChoiceXmlType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisChoiceType">
                        <xs:sequence>
                              <xs:any minOccurs="0" maxOccurs="unbounded"
processContents="lax"
                                    namespace="##other" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>


  <!-- Property Attributes -->
      <xs:element name="propertyDefinition"
type="cmis:cmisPropertyDefinitionType" />
      <xs:element name="propertyBooleanDefinition"
type="cmis:cmisPropertyBooleanDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyDateTimeDefinition"
type="cmis:cmisPropertyDateTimeDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyDecimalDefinition"
type="cmis:cmisPropertyDecimalDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyIdDefinition"
type="cmis:cmisPropertyIdDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyIntegerDefinition"
type="cmis:cmisPropertyIntegerDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyHtmlDefinition"
type="cmis:cmisPropertyHtmlDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyStringDefinition"
type="cmis:cmisPropertyStringDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyXmlDefinition"
type="cmis:cmisPropertyXmlDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:element name="propertyUriDefinition"
type="cmis:cmisPropertyUriDefinitionType"
            substitutionGroup="cmis:propertyDefinition" />
      <xs:complexType name="cmisPropertyDefinitionType">
            <xs:sequence>
                  <xs:element name="name" type="xs:string" minOccurs="1"
                        maxOccurs="1" />
                  <xs:element name="id" type="xs:string" minOccurs="1"
                        maxOccurs="1" />
                  <xs:element name="displayName" type="xs:string"
minOccurs="1"
                        maxOccurs="1" />
                  <xs:element name="description" type="xs:string"
minOccurs="0"
                        maxOccurs="1" />
```

```
                  <xs:element name="propertyType"
type="cmis:enumPropertyType"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="cardinality" type="cmis:enumCardinality"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="updateability"
type="cmis:enumUpdateability"
                        minOccurs="1" maxOccurs="1" />

                  <!-- flags -->
                  <xs:element name="inherited" type="xs:boolean"
minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="required" type="xs:boolean" minOccurs="1"
                        maxOccurs="1" />
                  <xs:element name="queryable" type="xs:boolean"
minOccurs="1"
                        maxOccurs="1" />
                  <xs:element name="orderable" type="xs:boolean"
minOccurs="1"
                        maxOccurs="1" />

                  <!--  choices -->
                  <xs:choice minOccurs="0" maxOccurs="unbounded">
                        <xs:annotation>
                              <xs:appinfo>
                                    <jaxb:property name="choice" />
                              </xs:appinfo>
                        </xs:annotation>
                        <xs:element ref="cmis:choiceBoolean" />
                        <xs:element ref="cmis:choiceDateTime" />
                        <xs:element ref="cmis:choiceDecimal" />
                        <xs:element ref="cmis:choiceHtml" />
                        <xs:element ref="cmis:choiceId" />
                        <xs:element ref="cmis:choiceInteger" />
                        <xs:element ref="cmis:choiceString" />
                        <xs:element ref="cmis:choiceUri" />
                        <xs:element ref="cmis:choiceXml" />
                  </xs:choice>
                  <xs:element name="openChoice" type="xs:boolean"
minOccurs="0"
                        maxOccurs="1" />

                  <!--  extension -->
                  <xs:any processContents="lax" namespace="##other"
minOccurs="0"
                        maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>

 <!--  type specific definitions -->
      <xs:complexType name="cmisPropertyBooleanDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
```

```
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceBooleanType" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyIdDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceIdType" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyIntegerDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceIntegerType" />
                              <xs:element name="maxValue" type="xs:integer"
minOccurs="0"
                                    maxOccurs="1" />
                              <xs:element name="minValue" type="xs:integer"
minOccurs="0"
                                    maxOccurs="1" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyDateTimeDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceDateTimeType" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyDecimalDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceDecimalType" />
                              <xs:element name="precision"
type="cmis:enumDecimalPrecision"
                                    minOccurs="0" maxOccurs="1" />
                        </xs:sequence>
```

```
                </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyHtmlDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceHtmlType" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyStringDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceStringType" />
                              <xs:element name="maxLength" type="xs:integer"
                                    minOccurs="0" maxOccurs="1" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyUriDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceUriType" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisPropertyXmlDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisPropertyDefinitionType">
                        <xs:sequence>
                              <xs:element minOccurs="0" maxOccurs="unbounded"
name="defaultValue"
                                    type="cmis:cmisChoiceXmlType" />
                              <xs:element name="schemaURI" type="xs:anyURI"
minOccurs="0"
                                    maxOccurs="1" />
                              <xs:element name="encoding" type="xs:string"
minOccurs="0"
                                    maxOccurs="1" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>

  <!-- type definition -->
```

```
	<xs:complexType name="cmisTypeDefinitionType" abstract="false">
		<xs:sequence>
			<xs:element name="objectId" type="xs:string" minOccurs="1"
				maxOccurs="1" />
			<xs:element name="queryName" type="xs:string" minOccurs="1"
				maxOccurs="1" />
			<xs:element name="displayName" type="xs:string"
minOccurs="1"
				maxOccurs="1" />

			<!--  base type -->
			<xs:element name="baseType" type="cmis:enumObjectType"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="baseTypeQueryName" type="xs:string"
				minOccurs="1" maxOccurs="1" />

			<!-- parent -->
			<xs:element name="parentId" minOccurs="0" maxOccurs="1" />

			<!-- info -->
			<xs:element name="description" type="xs:string"
minOccurs="0"
				maxOccurs="1" />

			<!-- flags -->
			<xs:element name="creatable" type="xs:boolean"
minOccurs="1"
				maxOccurs="1" />
			<xs:element name="fileable" type="xs:boolean" minOccurs="1"
				maxOccurs="1" />
			<xs:element name="queryable" type="xs:boolean"
minOccurs="1"
				maxOccurs="1" />
			<xs:element name="controllable" type="xs:boolean"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="includedInSupertypeQuery"
type="xs:boolean"
				minOccurs="1" maxOccurs="1" default="true" />

			<!-- property definitions -->
			<xs:choice minOccurs="0" maxOccurs="unbounded">
				<xs:annotation>
					<xs:appinfo>
						<jaxb:property name="propertyDefinition"
/>
					</xs:appinfo>
				</xs:annotation>
				<xs:element ref="cmis:propertyBooleanDefinition" />
				<xs:element ref="cmis:propertyDateTimeDefinition" />
				<xs:element ref="cmis:propertyDecimalDefinition" />
				<xs:element ref="cmis:propertyHtmlDefinition" />
				<xs:element ref="cmis:propertyIdDefinition" />
				<xs:element ref="cmis:propertyIntegerDefinition" />
				<xs:element ref="cmis:propertyStringDefinition" />
				<xs:element ref="cmis:propertyUriDefinition" />
				<xs:element ref="cmis:propertyXmlDefinition" />
			</xs:choice>
```

```
            <!-- extension -->
                  <xs:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
                        processContents="lax" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>


      <!-- type specific typedefs -->
      <xs:complexType name="cmisTypeDocumentDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisTypeDefinitionType">
                        <xs:sequence>
                              <xs:element name="versionable"
type="xs:boolean"
                                    minOccurs="1" maxOccurs="1" />
                              <xs:element name="contentStreamAllowed"
type="cmis:enumContentStreamAllowed"
                                    minOccurs="1" maxOccurs="1" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisTypeFolderDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisTypeDefinitionType">
                        <xs:sequence>
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisTypeRelationshipDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisTypeDefinitionType">
                        <xs:sequence>
                              <xs:element name="allowedSourceTypes"
type="xs:string"
                                    minOccurs="0" maxOccurs="unbounded" />
                              <xs:element name="allowedTargetTypes"
type="xs:string"
                                    minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="cmisTypePolicyDefinitionType">
            <xs:complexContent>
                  <xs:extension base="cmis:cmisTypeDefinitionType">
                        <xs:sequence>
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
```

```
	<!--  query -->
	<xs:complexType name="cmisQueryType">
		<xs:sequence>
			<xs:element name="statement" type="xs:string" minOccurs="1"
				maxOccurs="1" />
			<xs:element name="searchAllVersions" type="xs:boolean"
				minOccurs="0" maxOccurs="1" />
			<xs:element name="pageSize" type="xs:integer" minOccurs="0"
				maxOccurs="1" />
			<xs:element name="skipCount" type="xs:integer"
minOccurs="0"
				maxOccurs="1" />
			<xs:element name="returnAllowableActions" type="xs:boolean"
				minOccurs="0" maxOccurs="1" />
			<xs:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
				processContents="lax" />
		</xs:sequence>
		<xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
	</xs:complexType>
	<xs:element name="query" type="cmis:cmisQueryType" />


	<!--  repository info -->
	<xs:complexType name="cmisRepositoryInfoType">
		<xs:sequence minOccurs="1">
			<xs:element name="repositoryId" type="xs:string"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="repositoryName" type="xs:string"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="repositoryRelationship" type="xs:string"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="repositoryDescription" type="xs:string"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="vendorName" type="xs:string"
minOccurs="1"
				maxOccurs="1" />
			<xs:element name="productName" type="xs:string"
minOccurs="1"
				maxOccurs="1" />
			<xs:element name="productVersion" type="xs:string"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="rootFolderId" type="xs:string" />
			<xs:element name="capabilities"
type="cmis:cmisRepositoryCapabilitiesType"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="cmisVersionsSupported" type="xs:string"
				minOccurs="1" maxOccurs="1" />
			<xs:element name="repositorySpecificInformation"
type="cmis:cmisAnyXml"
				maxOccurs="1" minOccurs="0" />
			<xs:any namespace="##other" processContents="lax"
minOccurs="0"
				maxOccurs="unbounded" />
		</xs:sequence>
		<xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
```

```
      </xs:complexType>
      <xs:complexType name="cmisRepositoryCapabilitiesType">
            <xs:sequence>
                  <xs:element name="capabilityMultifiling" type="xs:boolean"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityUnfiling" type="xs:boolean"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityVersionSpecificFiling"
type="xs:boolean"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityPWCUpdateable"
type="xs:boolean"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityAllVersionsSearchable"
type="xs:boolean"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityQuery"
type="cmis:enumCapabilityQuery"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityJoin"
type="cmis:enumCapabilityJoin"
                        minOccurs="1" maxOccurs="1" />
                  <xs:element name="capabilityFullText"
type="cmis:enumCapabilityFullText"
                        minOccurs="1" maxOccurs="1" />
                  <xs:any namespace="##other" processContents="skip"
minOccurs="0"
                        maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attributeGroup ref="cmis:cmisUndefinedAttribute" />
      </xs:complexType>
</xs:schema>
<!-- EOF -->
```

APPENDIX

EXAMPLES

Each of the following resources have an example as defined below:

Resource (Root Tag)	Description	Example
Repository (Service)	This is the repository logical object. It is represented by the atom service document. This is also exposed on entry as link 'cmis-repository'	Please see Example-Service.xml

Root Folder Collection (Feed)	This is a collection described in the service document	Please see Example-FolderChildren.xml.
Checked Out Collection (Feed)	This is a collection described in the service document that contains all the checkedout documents	Please see Example-FolderChildren.xml. This will however, be a feed of private working copy document's only.
Unfiled Collection (Feed)	This is a collection of unfiled documents.	Please see Example-FolderChildren.xml. However there should be nothing returned.
Types Collection (Feed)	This is a collection described in the service document that contains all the types in the repository	
Type (Type)	This is the CMIS type of the object. This is exposed as link 'cmis-type',	Please see Example-Type.xml.
Document (Entry)	This is a CMIS Document instance. These are exposed in feeds or in an entry document. This can also be a private working copy (checkedout)	Please see Example-DocumentEntry.xml.
Document Private Working Copy (Entry)	This is a document in the private working copy of the document (checkedout version of document)	Please see Example-DocumentPWCEntry.xml.
Folder (Entry)	This is a CMIS Folder instance. This is exposed as a feed. The properties of a folder map onto the feed tag.	Please see Example-FolderEntry.xml.
Relationship	This is a CMIS relationship instance. These objects are exposed via 'cmis-	Please see Example-Relationship.xml.

(Entry)	relationships'	
Policy (Entry)	This is a CMIS policy instance. This is exposed via 'cmis-policies' as a feed	Please see Example-PolicyEntry.xml
Content Stream (Non-XML)	This is the content stream portion of the document object. This is exposed via 'cmis-stream' on the entry	No example. This is the original document format.
Allowable Actions (Allowable-Actions)	This is the set of actions available to the current user on the current object. This is exposed as a link 'cmis-allowableactions'	Please see Example-AllowableActions.xml
Relationships Collection (for a specific item) (Feed)	This is the set of relationships available (either source or target or both) from a specific item such as a document, folder or policy	Please see FolderChildren. However, the entries will be Relationships as in the RelationshipEntry example.
Parents Collection (for a specific document or policy object) (Feed)	This is the set of parents for a specific document or policy object. This can also be the ancestor of folders if returnToRoot is selected	Please see FolderChildren. However, the entries will be Folder as in the FolderEntry Example
Children (Feed)	This is a pseudo-object comprised of all the direct children of a particular folder. This is exposed as 'cmis-children'	Please see FolderChildren.
Descendants (Feed)	This is a pseudo-object comprised of all the direct and indirect children of a particular folder. This is exposed as 'cmis-descendants'	Please see FolderDescendants.
AllVersions	This is a pseudo-object made up of all document versions related to the current document version. This	Please see FolderChildren. However, the entries will be Documents as in DocumentEntry and

(Feed)	collection is also known as the version history	DocumentPWCEntry.

Please see the rest-schema.zip for Examples.

EXPRESSING MULTIPLE CONTENT STREAMS IN REST

CMIS does not currently support multiple content streams in the specification due to complexity and lack of support across a set of repositories. However, exposing the multiple content streams that already exist in a repository can be done quite readily with REST.

Inside the ATOM entry section:

```
<link rel="cmis-stream" foo:streamnumber=3
 href="http://example.org/media/atom03">
```

The same can be done in the cmis:object tag as well. The foo:streamnumber attribute tag exposes the stream id. Non-aware applications will see many links of relationship cmis-stream. If they are aware, they can chose which stream they want to retrieve.

For setting multiple content streams, this must be done outside of CMIS today.