



Version 0.5

Content Management Interoperability Services

Part II – SOAP protocol binding

CONTENTS

Contents	2
Copyright Notice	5
Introduction.....	6
Status	Error! Bookmark not defined.
Notation (RFC 2119).....	6
Part I of the CMIS Specification.....	6
Simple Object Access Protocol (SOAP) Binding	7
Overview	7
Notational Conventions	7
Authentication.....	7
WS-I	7
Content Transfer	7
Reporting Errors.....	8
Object Representation Format	8
Properties	8
Examples.....	11
Repository Services	18
getRepositoryes	18
getRepositoryInfo	18
getTypes	19
getTypeDefinition	20
Navigation Services	20
getDescendants	20
getChildren.....	21
getFolderParent	22

getObjectParents	23
getCheckedoutDocuments	23
Object Services.....	24
createDocument	24
createFolder.....	25
createRelationship.....	25
createPolicy.....	26
getAllowableActions.....	27
getProperties	27
getContentStream	27
updateProperties	28
moveObject.....	29
deleteObject	29
deleteTree	30
setContentStream.....	30
deleteContentStream.....	31
Multi-Filing Services	32
addObjectToFolder	32
removeObjectFromFolder	32
Discovery Services.....	33
query	33
Versioning Services	33
checkOut.....	33
cancelCheckOut.....	34
checkIn.....	34
getPropertiesOfLatestVersion	35
getAllVersions.....	35

deleteAllVersions.....	35
Relationship Services.....	36
getRelationships.....	36
POLICY Services.....	36
applyPolicy.....	36
removePolicy.....	37
getAppliedPolicies.....	37

COPYRIGHT NOTICE

(c) 2006-2008 EMC Corporation (EMC), International Business Machines Corporation (IBM), and Microsoft Corporation. All rights reserved.

Upon adoption as a standard, permission to copy and display the "Content Management Interoperability Services" Specification, in any medium without fee or royalty is hereby granted, provided that you include the following on ALL copies of the "Content Management Interoperability Services" Specification, or portions thereof, that you make:

1. A link or URL to the "Content Management Interoperability Services" Specification at this location: <td>.
2. The copyright notice as shown in the "Content Management Interoperability Services" Specification.

Upon adoption as a standard, EMC Corporation (EMC), International Business Machines Corporation (IBM), and Microsoft Corporation (collectively, the "Authors") each agree to grant you a royalty-free license, under reasonable, non-discriminatory terms and conditions to their respective patents to the extent that the Authors that they deem necessary to implement the "Content Management Interoperability Services" Specification.

THE "CONTENT MANAGEMENT INTEROPERABILITY SERVICES" SPECIFICATION IS PROVIDED "AS IS," AND THE AUTHORS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE "CONTENT MANAGEMENT INTEROPERABILITY SERVICES" SPECIFICATION ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

THE AUTHORS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THE "CONTENT MANAGEMENT INTEROPERABILITY SERVICES" SPECIFICATION.

The name and trademarks of the Authors may NOT be used in any manner, including advertising or publicity pertaining to the "Content Management Interoperability Services" Specification or its contents without specific, written prior permission. Title to copyright in the "Content Management Interoperability Services" Specification will at all times remain with the Authors.

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.

SIMPLE OBJECT ACCESS PROTOCOL (SOAP) BINDING

OVERVIEW

All services and operations defined in part I of the CMIS specification are presented in this SOAP binding. The WSDL for these SOAP-based web services can be found in here:

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

The WSDL for these services reference two XSD documents. One defines elements for the primary data types of documents, folders, relationships and policies as well as collections of these types of objects. **Section 3.0** defines the mapping between all relevant pieces of the domain model as described in Part I of the CMIS specification. The second XSD defines the message formats for each of the CMIS services; the messages often refer to the data types defined in the first XSD schema. The WSDL presents exactly the abstract services defined in the services section of Part I of the CMIS specification.

The normative CMIS SOAP binding is defined by the WSDL and XSD as well as the details given here in this part of the CMIS specification except the examples.

Notational Conventions

The elements in both XSD schemas are defined in the CMIS namespace. The XML Namespace URI for CMIS is:

<http://www.cmis.org/ns/cmis1>

(Note: Namespace URI to be updated for final version.)

In each of the examples in this section the CMIS namespace is the default namespace, hence elements are shown without a namespace prefix.

Many of the services that are retrieving objects or sets of objects accept a filter that specifies the object properties that are returned. The examples below include only a limited set of properties for brevity. In most cases many more properties may be included in the XML that is returned. When examples are presented where only a limited set of properties are returned the accompanying text will clearly state that.

Authentication

A CMIS SOAP binding **MUST** support WS-Security 1.1 and Username Token Profile 1.1.

WS-I

A CMIS SOAP binding **SHALL** comply with WS-I Basic Profile 1.1 and Basic Security Profile 1.0.

Content Transfer

A CMIS SOAP binding **SHOULD** support MTOM content transfers. For examples please see sections 5.6 and 5.11.

Reporting Errors

Services SHALL report errors via SOAP faults. The `cmisMessageTypes.xsd` defines a basic fault structure that includes an error code and an error message and the WSDL for each service define specific messages that have the basic fault format.

OBJECT REPRESENTATION FORMAT

The Domain Model in Part I of the CMIS specification defines Document, Folder, Relationship, and Policy object types. This section defines a mapping between the pieces of that domain model and the elements of the CMIS SOAP binding XSD.

Properties

Properties are represented by generic `<cmis:propertyXX>` elements, where XX indicates a CMIS property type. There is one generic property element type for each CMIS property type:

- `<cmis:propertyBoolean>`
- `<cmis:propertyDateTime>`
- `<cmis:propertyDecimal>`
- `<cmis:propertyHtml>`
- `<cmis:propertyId>`
- `<cmis:propertyInteger>`
- `<cmis:propertyString>`
- `<cmis:propertyUri>`
- `<cmis:propertyXml>`

Each property is represented by a `<cmis:propertyXX>` element in the following form:

```
<cmis:propertyXX cmis:name="property_name" cmis:index="ordinal_number">  
  <cmis:value>property_value</cmis:value>  
</cmis:propertyXX>
```

The index attribute is used to indicate the ordinal position of each property value among the values of a multi-valued property. Specifically, the index attribute values for the set of sibling elements with the same name MUST contain a sequence of integer values from 0 (zero) to $n-1$ where n is the number of values that this multi-valued property has. The index attribute is not needed for single-valued properties.

The property elements for an object are nested inside a `<cmis:properties>` element. They MAY appear in any order.

COMMON PROPERTIES

<i>CMIS Property Name</i>	<i>XML Element</i>
---------------------------	--------------------

ObjectId	<cmis:propertyId cmis:name="ObjectId">
Uri	<cmis:propertyUri cmis:name="Uri">
ObjectTypeId	<cmis:propertyId cmis:name="ObjectTypeId" >
CreatedBy	<cmis:propertyString cmis:name="CreatedBy">
CreationDate	<cmis:propertyDateTime cmis:name="CreationDate">
LastModifiedBy	<cmis:propertyString cmis:name="LastModifiedBy">
LastModificationDate	<cmis:propertyDateTime cmis:name="LastModificationDate">
ChangeToken	<cmis:propertyString cmis:name="ChangeToken">

DOCUMENT PROPERTIES

CMIS Property Name	XML Element
Name	<cmis:propertyString cmis:name="Name">
IsImmutable	<cmis:propertyBoolean cmis:name="IsImmutable">
IsLatestVersion	<cmis:propertyBoolean cmis:name="IsLatestVersion">
IsMajorVersion	<cmis:propertyBoolean cmis:name="IsMajorVersion">
IsLatestMajorVersion	<cmis:propertyBoolean cmis:name="IsLatestMajorVersion">
VersionLabel	<cmis:propertyString cmis:name="VersionLabel">
VersionSeriesId	<cmis:propertyId cmis:name="VersionSeriesId">
IsVersionSeriesCheckedOut	<cmis:propertyBoolean cmis:name="IsVersionSeriesCheckedOut">
VersionSeriesCheckedOutBy	<cmis:propertyString cmis:name="VersionSeriesCheckedOutBy">
VersionSeriesCheckedOutId	<cmis:propertyId cmis:name="VersionSeriesCheckedOutId">
CheckinComment	<cmis:propertyString cmis:name="CheckinComment">
ContentStreamAllowed	<cmis:propertyString cmis:name="ContentStreamAllowed">
ContentStreamLength	<cmis:propertyInteger cmis:name="ContentStreamLength">

ContentStreamMimeType	<cmis:propertyString cmis:name="ContentStreamMimeType">
ContentStreamFilename	<cmis:propertyString cmis:name="ContentStreamFilename">
ContentStreamUri	<cmis:propertyUri cmis:name="ContentStreamUri">

FOLDER PROPERTIES

<i>CMIS Property Name</i>	<i>XML Element</i>
Name	<cmis:propertyString cmis:name="Name">
ParentId	<cmis:propertyId cmis:name="ParentId">
AllowedChildObjectTypes	<cmis:propertyId cmis:name="AllowedChildObjectTypes" cmis:index="">

RELATIONSHIP PROPERTIES

<i>CMIS Property Name</i>	<i>XML Element</i>
SourceId	<cmis:propertyId cmis:name="SourceId">
TargetId	<cmis:propertyId cmis:name="TargetId">

POLICY PROPERTIES

<i>CMIS Property Name</i>	<i>XML Element</i>
PolicyName	<cmis:propertyString cmis:name="PolicyName">
PolicyText	<cmis:propertyString cmis:name="PolicyText">

Examples

The examples in this Part of the specification are not normative. They assume a folder and document hierarchy as follows:

- Folder X
 - Document A
- Folder Y
 - Document B
 - Document C

The following would be returned from a *getChildren()* call against Folder X. It consists of a list of <object> elements.

```
<cmis:getChildrenResponse>
  <cmis:object>
    <cmis:properties>
      <cmis:propertyBoolean cmis:name="IsImmutable">
        <cmis:value>>false</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsLatestVersion">
        <cmis:value>>true</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsMajorVersion">
        <cmis:value>>true</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsLatestMajorVersion">
        <cmis:value>>true</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsVersionSeriesCheckedOut">
        <cmis:value>>false</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyDateTime cmis:name="CreationDate">
        <cmis:value>2007-06-30T12:29:29Z</cmis:value>
      </cmis:propertyDateTime>
      <cmis:propertyDateTime cmis:name="LastModificationDate">
        <cmis:value>2007-06-30T12:29:29Z</cmis:value>
      </cmis:propertyDateTime>
      <cmis:propertyId cmis:name="ObjectId">
        <cmis:value>DocumentAId</cmis:value>
      </cmis:propertyId>
      <cmis:propertyId cmis:name="ObjectTypeId">
        <cmis:value>myDocumentTypeId</cmis:value>
      </cmis:propertyId>
      <cmis:propertyId cmis:name="VersionSeriesId">
        <cmis:value>DocumentASeriesId</cmis:value>
      </cmis:propertyId>
      <cmis:propertyInteger cmis:name="ContentStreamLength">
        <cmis:value>534</cmis:value>
      </cmis:propertyInteger>
    </cmis:properties>
  </cmis:object>
</cmis:getChildrenResponse>
```

```
<cmis:propertyString cmis:name="CreatedBy">
  <cmis:value>Cornelia Davis</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="LastModifiedBy">
  <cmis:value>Cornelia Davis</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ChangeToken">
  <cmis:value>asfgwihogwgqo</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="Name">
  <cmis:value>My document</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="VersionLabel">
  <cmis:value>1.0</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="CheckinComment">
  <cmis:value>First review version</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamAllowed">
  <cmis:value>allowed</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamMimeType">
  <cmis:value>application/msword</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamFilename">
  <cmis:value>myDocument.doc</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="Keyword" cmis:index="0">
  <cmis:value>XML</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="Keyword" cmis:index="1">
  <cmis:value>standards</cmis:value>
</cmis:propertyString>
<cmis:propertyUri cmis:name="Uri">
  <cmis:value>http://www.acme.com/001</cmis:value>
</cmis:propertyUri>
<cmis:propertyUri cmis:name="ContentStreamUri">
  <cmis:value>http://www.acme.com/001/stream</cmis:value>
</cmis:propertyUri>
</cmis:properties>
</cmis:object>
<cmis:object>
  <cmis:properties>
    <cmis:propertyDateTime cmis:name="CreationDate">
      <cmis:value>2007-07-15T10:31:20Z</cmis:value>
    </cmis:propertyDateTime>
    <cmis:propertyDateTime cmis:name="LastModificationDate">
      <cmis:value>2007-09-02T12:12:34Z</cmis:value>
    </cmis:propertyDateTime>
    <cmis:propertyId cmis:name="ObjectId">
```

```

        <cmis:value>FolderYId</cmis:value>
      </cmis:propertyId>
    <cmis:propertyId cmis:name="ObjectTypeId">
      <cmis:value>myFolderTypeId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyId cmis:name="ParentId">
      <cmis:value>FolderXId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyString cmis:name="CreatedBy">
      <cmis:value>Al Brown</cmis:value>
    </cmis:propertyString>
    <cmis:propertyString cmis:name="LastModifiedBy">
      <cmis:value>Al Brown</cmis:value>
    </cmis:propertyString>
    <cmis:propertyString cmis:name="ChangeToken">
      <cmis:value>sdflljpouqegldf</cmis:value>
    </cmis:propertyString>
    <cmis:propertyString cmis:name="Name">
      <cmis:value>My folder</cmis:value>
    </cmis:propertyString>
    <cmis:propertyUri cmis:name="Uri">
      <cmis:value>http://www.acme.com/002</cmis:value>
    </cmis:propertyUri>
  </cmis:properties>
</cmis:object>
  <hasMoreItems>>false</hasMoreItems>
</cmis:getChildrenResponse>

```

The following would be returned from a *getDescendants()* call against Folder X with depth of 2 (or more). Compared to the first example, it includes a <child> element for Document B and a <child> element for Document C nested inside the <object> element for Folder Y (shown below in *italics*). There is no <hasMoreItems> element since paging is not supported.

```

<cmis:getDescendantsResponse>
  <cmis:object>
    <cmis:properties>
      <cmis:propertyBoolean cmis:name="IsImmutable">
        <cmis:value>>false</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsLatestVersion">
        <cmis:value>>true</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsMajorVersion">
        <cmis:value>>true</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsLatestMajorVersion">
        <cmis:value>>true</cmis:value>
      </cmis:propertyBoolean>
      <cmis:propertyBoolean cmis:name="IsVersionSeriesCheckedOut">
        <cmis:value>>false</cmis:value>

```

```
</cmis:propertyBoolean>
<cmis:propertyDateTime cmis:name="CreationDate">
  <cmis:value>2007-06-30T12:29:29Z</cmis:value>
</cmis:propertyDateTime>
<cmis:propertyDateTime cmis:name="LastModificationDate">
  <cmis:value>2007-06-30T12:29:29Z</cmis:value>
</cmis:propertyDateTime>
<cmis:propertyId cmis:name="ObjectId">
  <cmis:value>DocumentAId</cmis:value>
</cmis:propertyId>
<cmis:propertyId cmis:name="ObjectTypeId">
  <cmis:value>myDocumentTypeId</cmis:value>
</cmis:propertyId>
<cmis:propertyId cmis:name="VersionSeriesId">
  <cmis:value>DocumentASeriesId</cmis:value>
</cmis:propertyId>
<cmis:propertyInteger cmis:name="ContentStreamLength">
  <cmis:value>534</cmis:value>
</cmis:propertyInteger>
<cmis:propertyString cmis:name="CreatedBy">
  <cmis:value>Cornelia Davis</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="LastModifiedBy">
  <cmis:value>Cornelia Davis</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ChangeToken">
  <cmis:value>asfgwihogwgqo</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="Name">
  <cmis:value>My document</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="VersionLabel">
  <cmis:value>1.0</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="CheckinComment">
  <cmis:value>First review version</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamAllowed">
  <cmis:value>allowed</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamMimeType">
  <cmis:value>application/msword</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamFilename">
  <cmis:value>myDocument.doc</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="Keyword" cmis:index="0">
  <cmis:value>XML</cmis:value>
</cmis:propertyString>
```

```
<cmis:propertyString cmis:name="Keyword" cmis:index="1">
  <cmis:value>standards</cmis:value>
</cmis:propertyString>
<cmis:propertyUri cmis:name="Uri">
  <cmis:value>http://www.acme.com/001</cmis:value>
</cmis:propertyUri>
<cmis:propertyUri cmis:name="ContentStreamUri">
  <cmis:value>http://www.acme.com/001/stream</cmis:value>
</cmis:propertyUri>
</cmis:properties>
</cmis:object>
<cmis:object>
  <cmis:properties>
    <cmis:propertyDateTime cmis:name="CreationDate">
      <cmis:value>2007-07-15T10:31:20Z</cmis:value>
    </cmis:propertyDateTime>
    <cmis:propertyDateTime cmis:name="LastModificationDate">
      <cmis:value>2007-09-02T12:12:34Z</cmis:value>
    </cmis:propertyDateTime>
    <cmis:propertyId cmis:name="ObjectId">
      <cmis:value>FolderYId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyId cmis:name="ObjectTypeId">
      <cmis:value>myFolderTypeId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyId cmis:name="ParentId">
      <cmis:value>FolderXId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyString cmis:name="CreatedBy">
      <cmis:value>Al Brown</cmis:value>
    </cmis:propertyString>
    <cmis:propertyString cmis:name="LastModifiedBy">
      <cmis:value>Al Brown</cmis:value>
    </cmis:propertyString>
    <cmis:propertyString cmis:name="ChangeToken">
      <cmis:value>sdflljpouqegldf</cmis:value>
    </cmis:propertyString>
    <cmis:propertyString cmis:name="Name">
      <cmis:value>My folder</cmis:value>
    </cmis:propertyString>
    <cmis:propertyUri cmis:name="Uri">
      <cmis:value>http://www.acme.com/002</cmis:value>
    </cmis:propertyUri>
  </cmis:properties>

  <cmis:child>
    <cmis:properties>
      <cmis:propertyBoolean cmis:name="IsImmutable">
        <cmis:value>true</cmis:value>
      </cmis:propertyBoolean>
    </cmis:properties>
  </cmis:child>
</cmis:object>
```

```
<cmis:propertyBoolean cmis:name="IsLatestVersion">
  <cmis:value>true</cmis:value>
</cmis:propertyBoolean>
<cmis:propertyBoolean cmis:name="IsMajorVersion">
  <cmis:value>>false</cmis:value>
</cmis:propertyBoolean>
<cmis:propertyBoolean cmis:name="IsLatestMajorVersion">
  <cmis:value>>false</cmis:value>
</cmis:propertyBoolean>
<cmis:propertyBoolean cmis:name="IsVersionSeriesCheckedOut">
  <cmis:value>>false</cmis:value>
</cmis:propertyBoolean>
<cmis:propertyDateTime cmis:name="CreationDate">
  <cmis:value>2007-05-25T10:12:32Z</cmis:value>
</cmis:propertyDateTime>
<cmis:propertyDateTime cmis:name="LastModificationDate">
  <cmis:value>2007-06-29T12:36:48Z</cmis:value>
</cmis:propertyDateTime>
<cmis:propertyId cmis:name="ObjectId">
  <cmis:value>DocumentBId</cmis:value>
</cmis:propertyId>
<cmis:propertyId cmis:name="ObjectTypeId">
  <cmis:value>docTypeId</cmis:value>
</cmis:propertyId>
<cmis:propertyId cmis:name="VersionSeriesId">
  <cmis:value>DocumentBSeriesId</cmis:value>
</cmis:propertyId>
<cmis:propertyInteger cmis:name="ContentStreamLength">
  <cmis:value>781</cmis:value>
</cmis:propertyInteger>
<cmis:propertyString cmis:name="CreatedBy">
  <cmis:value>David Choy</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="LastModifiedBy">
  <cmis:value>David Choy</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ChangeToken">
  <cmis:value>sdflaksrtlngg</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="Name">
  <cmis:value>Release Notes</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="VersionLabel">
  <cmis:value>Draft</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamAllowed">
  <cmis:value>allowed</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamMimeType">
  <cmis:value>application/pdf</cmis:value>
</cmis:propertyString>
<cmis:propertyString cmis:name="ContentStreamFilename">
  <cmis:value>Version4ReleaseNotes.pdf</cmis:value>
</cmis:propertyString>
```

```
<cmis:propertyUri cmis:name="Uri">
  <cmis:value>http://www.acme.com/003</cmis:value>
</cmis:propertyUri>
<cmis:propertyUri cmis:name="ContentStreamUri">
  <cmis:value>http://www.acme.com/002/stream</cmis:value>
</cmis:propertyUri>
</cmis:properties>
</cmis:child>
<cmis:child>
  <cmis:properties>
    <cmis:propertyId cmis:name="ObjectId">
      <cmis:value>DocumentCId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyId cmis:name="ObjectTypeId">
      <cmis:value>anotherDocTypeId</cmis:value>
    </cmis:propertyId>
    ...
    <cmis:propertyUri cmis:name="Uri">
      <cmis:value>http://www.acme.com/004</cmis:value>
    </cmis:propertyUri>
  </cmis:properties>
</cmis:child>

</cmis:object>
</cmis:getDescendantsResponse>
```

For other examples below, the default namespace is assumed to be "cmis".

REPOSITORY SERVICES

The RepositoryService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

getRepositories

Description
Returns a list of available repositories for this CMIS service endpoint.

The output message for this service contains a list of repositories in the following form:

```
<getRepositoriesResponse>
  <repository>
    <repositoryID>5001</repositoryID>
    <repositoryName>My Repository</repositoryName>
    <repositoryURI>http://www.acme.com/myRepository</repositoryURI>
  </repository>
  <repository>
    <repositoryID>5008</repositoryID>
    <repositoryName>My Archive Repository</repositoryName>
    <repositoryURI>http://www.acme.com/myArchive</repositoryURI>
  </repository>
</getRepositoriesResponse>
```

getRepositoryInfo

Description
Gets information about a CMIS repository

This service takes a repository ID and produces a message containing a <repositoryInfo> element of the following form:

```
<getRepositoryInfoResponse>
  <repositoryId>5001</repositoryId>
  <repositoryName>My Repository</repositoryName>
  <repositoryRelationship>self</repositoryRelationship>
  <repositoryDescription>Repository Descripton</repositoryDescription>
  <vendorName>XYZ Corporation</vendorName>
  <productName>XYZ ECM Suite</productName>
  <productVersion>1.1.0.2</productVersion>
  <rootFolderId>01234</rootFolderId>
  <capabilities>
    <capabilityMultifiling>true</capabilityMultifiling>
    <capabilityUnfiling>true</capabilityUnfiling>
  </capabilities>
</getRepositoryInfoResponse>
```

```

    <capabilityVersionSpecificFiling>false
      </capabilityVersionSpecificFiling>
    <capabilityPWCUpdateable>true</capabilityPWCUpdateable>
    <capabilityAllVersionsSearchable>true
      </capabilityAllVersionsSearchable>
    <capabilityPWCSearchable>false</capabilityPWCSearchable>
    <capabilityJoin>inneronly</capabilityJoin>
    <capabilityFullText>fulltextandstructured
      </capabilityFullText>
  </capabilities>
  <cmisVersionsSupported>1.0</cmisVersionsSupported>
  <repositorySpecificInformation>...</repositorySpecificInformation>
  <relatedRepository>
    <repositoryID>5008</repositoryID>
    <repositoryName>My Archive Repository</repositoryName>
    <repositoryRelationship>archive</repositoryRelationship>
  </relatedRepository>
</getRepositoryInfoResponse>

```

getTypes

Description

Returns the list of all types in the repository.

This service produces an output message containing a <types> element which contains <type> children. If the service is called with returnPropertyDefinitions set to false, only type attributes will be returned (without property definition), as follows:

```

<getTypesResponse>
  <types>
    <type>
      <typeID>TypeAId</typeID>
      <queryName>TypeNameA</queryName>
      <displayName>Type A Display Name</displayName>
      <baseType>folder</baseType>
      <baseTypeQueryName>Folder</baseTypeQueryName>
      <parentID>ParentTypeId</parentID>
      <description>Description</description>
      <creatable>true</creatable>
      <fileable>true</fileable>
      <queryable>true</queryable>
      <controllable>false</controllable>
      <includedInSupertypeQuery>true</includedInSupertypeQuery>
      <versionable>false</versionable>
      <contentStreamAllowed>notAllowed</contentStreamAllowed>
      <allowedSourceTypes></allowedSourceTypes>
      <allowedTargetTypes></allowedTargetTypes>
    </type>
    <type>
      <typeID>TypeBId</typeID>
      <queryName>TypeNameB</queryName>
      <displayName>Type B Display Name</displayName>
      ...
  </types>
</getTypesResponse>

```

```
</type>
  <hasMoreItems>false</hasMoreItems>
</getTypesResponse>
```

getTypeDefinition

Description	Gets the definition for specified type
--------------------	--

The service produces an output message containing a <type> element, which has the same form as a <type> element returned by the getTypes() service call with returnPropertyDefinitions set to true.

NAVIGATION SERVICES

The NavigationService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

getDescendants

Description	Gets the list of 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.
--------------------	---

The set of properties returned for each object is specified in the filter that is input. Below is an example if the filter selects the *ObjectID*, *ObjectTypeID*, and *Uri* properties.

```
<cmis:getDescendantsResponse>
  <cmis:object>
    <cmis:properties>
      <cmis:propertyId cmis:name="ObjectID">
        <cmis:value>DocumentAId</cmis:value>
      </cmis:propertyId>
      <cmis:propertyId cmis:name="ObjectTypeID">
        <cmis:value>myDocumentTypeId</cmis:value>
      </cmis:propertyId>
      <cmis:propertyUri cmis:name="Uri">
        <cmis:value>http://www.acme.com/001</cmis:value>
      </cmis:propertyUri>
    </cmis:properties>
  </cmis:object>
  <cmis:object>
    <cmis:properties>
      <cmis:propertyId cmis:name="ObjectID">
        <cmis:value>FolderYId</cmis:value>
      </cmis:propertyId>
    </cmis:properties>
  </cmis:object>
</cmis:getDescendantsResponse>
```

```

        </cmis:propertyId>
        <cmis:propertyId cmis:name="ObjectTypeId">
            <cmis:value>myFolderTypeId</cmis:value>
        </cmis:propertyId>
        <cmis:propertyUri cmis:name="Uri">
            <cmis:value>http://www.acme.com/002</cmis:value>
        </cmis:propertyUri>
    </cmis:properties>
    <cmis:child>
        <cmis:properties>
            <cmis:propertyId cmis:name="ObjectId">
                <cmis:value>DocumentBId</cmis:value>
            </cmis:propertyId>
            <cmis:propertyId cmis:name="ObjectTypeId">
                <cmis:value>docTypeId</cmis:value>
            </cmis:propertyId>
            <cmis:propertyUri cmis:name="Uri">
                <cmis:value>http://www.acme.com/003</cmis:value>
            </cmis:propertyUri>
        </cmis:properties>
    </cmis:child>
    <cmis:child>
        <cmis:properties>
            <cmis:propertyId cmis:name="ObjectId">
                <cmis:value>DocumentCId</cmis:value>
            </cmis:propertyId>
            <cmis:propertyId cmis:name="ObjectTypeId">
                <cmis:value>anotherDocTypeId</cmis:value>
            </cmis:propertyId>
            <cmis:propertyUri cmis:name="Uri">
                <cmis:value>http://www.acme.com/004</cmis:value>
            </cmis:propertyUri>
        </cmis:properties>
    </cmis:child>
</cmis:object>
</cmis:getDescendantsResponse>

```

getChildren

Description

Gets the list of objects contained at one level below the specified folder. Only the selected properties associated with each object are returned. The content stream is not returned.

The set of properties returned for each object is specified in the filter that is input. Below is an example if the filter selects the *ObjectID*, *ObjectTypeID*, and *Uri* properties.

```
<cmis:getChildrenResponse>
```

```

<cmis:object>
  <cmis:properties>
    <cmis:propertyId cmis:name="ObjectId">
      <cmis:value>DocumentAId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyId cmis:name="ObjectTypeId">
      <cmis:value>myDocumentTypeId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyUri cmis:name="Uri">
      <cmis:value>http://www.acme.com/001</cmis:value>
    </cmis:propertyUri>
  </cmis:properties>
</cmis:object>
<cmis:object>
  <cmis:properties>
    <cmis:propertyId cmis:name="ObjectId">
      <cmis:value>FolderYId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyId cmis:name="ObjectTypeId">
      <cmis:value>myFolderTypeId</cmis:value>
    </cmis:propertyId>
    <cmis:propertyUri cmis:name="Uri">
      <cmis:value>http://www.acme.com/002</cmis:value>
    </cmis:propertyUri>
  </cmis:properties>
</cmis:object>
<hasMoreItems>false</hasMoreItems>
</cmis:getChildrenResponse>

```

getFolderParent

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

Either only the immediate parent of the specified folder is returned or all folders on the path to the root are returned. The output message contains a list of one or more <object> elements. The order in which these <object> elements are returned is repository-specific. Repositories SHOULD always return the *ObjectId* and *ParentId* properties for each folder to allow construction of the path. The format of each <object> element is the same as that in the output message for the getChildren() service, including filter-selected properties.

In the example below, Folder X is the root folder. The *ParentId* property of the root folder contains the root folder's own object ID.

```

<getFolderParentResponse>
  <object>
    <properties>
      <propertyId name="ObjectId"><value>FolderYId</value></propertyId>

```

```

        <propertyId name="ParentId"><value>FolderXId</value></propertyId>
    </properties>
</object>
<object>
    <properties>
        <propertyId name="ObjectId"><value>FolderXId</value></propertyId>
        <propertyId name="ParentId"><value>FolderXId</value></propertyId>
    </properties>
</object>
</getFolderParentResponse>

```

getObjectParents

Description
Returns the parent folders for the specified non-folder fileable object.

Only the immediate parent(s) of the specified object are returned. The output message format is the same as that of the `getFolderParent()` service, containing a list of `<object>` elements. The order in which these `<object>` elements are returned is repository-specific.

```

<getObjectParentsResponse>
  <object>
    <properties>
      <propertyId name="ObjectId"><value>Folder1Id</value></propertyId>
      <propertyString name="Name"><value>Folder 1</value>
    </properties>
  </object>
  <object>
    <properties>
      <propertyId name="ObjectId"><value>Folder2Id</value></propertyId>
      <propertyString name="Name"><value>Folder 2</value>
    </properties>
  </object>
  <object>
    <properties>
      <propertyId name="ObjectId"><value>Folder3Id</value></propertyId>
      <propertyString name="Name"><value>Folder 3</value>
    </properties>
  </object>
</getObjectParentsResponse>

```

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 output message format is the same as that for the `getChildren()` service, consisting of a list of `<object>` elements followed by a `<hasMoreItems>` element.

```
<getCheckedoutDocsResponse>
  <object>
    <properties>
      <propertyId name="ObjectId"><value>DocumentAId</value></propertyId>
      <propertyId name="ObjectTypeId"><value>docTypeId</value>
      </propertyId>
    </properties>
  </object>
  <object>
    <properties>
      <propertyId name="ObjectId"><value>DocumentBId</value></propertyId>
      <propertyId name="ObjectTypeId"><value>docTypeId</value>
      </propertyId>
    </properties>
  </object>
  <hasMoreItems>true</hasMoreItems>
</getCheckedoutDocsResponse>
```

OBJECT SERVICES

The `ObjectService.wsdl`, `cmisMessageTypes.xsd` and `CMIS.xsd` files fully describe this service.

createDocument

Description	Creates a document of the specified type
--------------------	--

The input message for this service has the following form. If the set of properties includes an *ObjectTypeId* then the value MUST match the value of the *typeId* element of the message. The repository MAY enforce constraints on the object properties or the content and if any of those constraints are violated the object will not be created and the service will return an error. If present, some optional input properties such as *ObjectId*, *Uri*, *ContentStreamLength*, *ContentStreamUri* and *CreationDate* MAY be ignored by the repository. The content stream, if included in the `createDocument()` request, MUST be transferred via MTOM. For an example of MTOM based content transfers please see `setContentStream()` service.

```
<createDocument>
  <repositoryId>TravelRepositoryId</repositoryId>
  <typeId>ExpenseReportTypeId</typeId>
  <properties>
```

```

    <propertyDateTime name="TravelStartDate">
      <value>2008-03-02T08:00:00Z</value>
    </propertyDateTime>
    <propertyDateTime name="TravelEndDate">
      <value>2008-03-05T17:00:00Z</value>
    </propertyDateTime>
    <propertyId name="ObjectId">
      <value>probably ignored</value>
    </propertyId>
    <propertyId name="ObjectTypeId">
      <value>ExpenseReportTypeID</value>
    </propertyId>
  </properties>
  <folderId>parentFolderId</folderId>
  <versioningState>checkedout</versioningState>
</createDocument>

```

createFolder

Description	Creates a folder of the specified type
-------------	--

The input message for this service has the following form. If the set of properties includes an *ObjectTypeId* then the value MUST match the value of the *typeId* element of the message. The repository MAY enforce constraints on the object properties and if any of those constraints are violated the object will not be created and the service will return an error. Some input properties such as *ObjectId*, *Uri* and *CreationDate* MAY be ignored by the repository.

```

<createFolder>
  <repositoryId>MyRepositoryId</repositoryId>
  <typeId>myFolderTypeId</typeId>
  <properties>
    <propertyId name="ObjectId">
      <value>probably ignored</value>
    </propertyId>
    <propertyId name="ObjectTypeId">
      <value>myFolderTypeId</value>
    </propertyId>
    <propertyString name="Name">
      <value>myFolderName</value>
    </propertyString>
    <propertyString name="Project">
      <value>my project name</value>
    </propertyString>
  </properties>
  <folderId>parentFolderId</folderId>
</createFolder>

```

createRelationship

Description	Creates a relationship of the specified type
--------------------	--

The input message for this service has the following form. If the set of properties includes an *ObjectTypeId*, *SourceId* or *TargetId* then the values MUST match the values of the *typeId*, *sourceObjectId* and *targetObjectId* elements of the message, respectively. The repository MAY enforce constraints on the object properties and if any of those constraints are violated the object will not be created and the service will return an error. Some input properties such as *ObjectId*, *Uri* and *CreationDate* MAY be ignored by the repository.

```
<createRelationship>
  <repositoryId>MyRepositoryId</repositoryId>
  <typeId>myRelationshipTypeId</typeId>
  <properties>
    <propertyId name="ObjectId">
      <value>probably ignored</value>
    </propertyId>
    <propertyId name="ObjectTypeId">
      <value>myRelationshipTypeId</value>
    </propertyId>
    <propertyId name="SourceId">
      <value>objectIdA</value>
    </propertyId>
    <propertyId name="TargetId">
      <value>objectIdB</value>
    </propertyId>
  </properties>
  <sourceObjectId>objectIdA</sourceObjectId>
  <targetObjectId>objectIdB</targetObjectId>
</createRelationship>
```

createPolicy

Description	Creates a policy of the specified type
--------------------	--

The input message for this service has the following form. If the set of properties includes an *ObjectTypeId*, then the value MUST match the value of the *typeId* elements of the message. The repository MAY enforce constraints on the object properties and if any of those constraints are violated the object will not be created and the service will return an error. Some input properties such as *ObjectId*, *Uri* and *CreationDate* MAY be ignored by the repository.

```
<createPolicy>
  <repositoryId>MyRepositoryId</repositoryId>
  <typeId>myPolicyTypeId</typeId>
  <properties>
    <propertyId name="ObjectId">
      <value>probably ignored</value>
    </propertyId>
```

```
<propertyId name="ObjectTypeId">
  <value>myPolicyTypeId</value>
</propertyId>
<propertyString name="PolicyName">
  <value>My Default ACL</value>
</propertyString>
<propertyString name="PolicyText">
  <value>...</value>
</propertyString>
</properties>
<folderId>parentFolderId</folderId>
</createPolicy>
```

getAllowableActions

Description	Returns the list of allowable actions for an object based on the current user's context.
--------------------	--

The service produces an output message containing a <allowableActions> element with zero or more action permission settings.

```
<getAllowableActionsResponse>
  <allowableActions>
    <canGetProperties>true</canGetProperties>
    <canViewContent>true</canViewContent>
  </allowableActions>
</getAllowableActionsResponse>
```

getProperties

Description	Returns the properties of an object
--------------------	-------------------------------------

The output message contains a single <object> element, which has the same format as that in the output message for the getChildren() service.

getContentStream

Description	Returns the content stream for the document
--------------------	---

The service produces an HTTP response that is multi-part, the first part containing the SOAP envelope containing the <getContentStreamResponse> element which contains an <xop:Include> element with a value that is the ID of the associated second part to the response. The second part will contain the binary stream. The HTTP response will have the following form:

```

200 OK
Content-Type: Multipart/Related; type="application/xop+xml"; boundary="----
_Part_0_14615608.1167536350647"; start-info="text/xml"
Date: 23 Aug 2006 23:54:12 GMT
Connection: Close

-----=_Part_0_14615608.1167536350647
Content-Type: application/xop+xml; type="text/xml"; charset=utf-8

<?xml version="1.0" ?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ns1="http://catalog.mtom.company.com/book"
  xmlns:xop="http://www.w3.org/2004/08/xop/include">
  <soapenv:Body>
    <getContentStreamResponse>
      <stream>
        <xop:Include>
          xmlns:xop="http://www.w3.org/2004/08/xop/include"
          href="cid:1d0353b6-0ec7-4264-b4f4-
16c1e6a0812c@www.acme.com">
        </xop:Include>
      </stream>
    </getContentStreamResponse>
  </soapenv:Body>
</soapenv:Envelope>
-----=_Part_0_14615608.1167536350647
Content-Type: application/octet-stream
Content-ID: <1d0353b6-0ec7-4264-b4f4-16c1e6a0812c@www.acme.com>
Content-transfer-encoding: binary

.....(binary stream omitted)
-----=_Part_0_14615608.1167536350647

```

updateProperties

Description	Updates properties of the specified object
--------------------	--

The message input to the service includes a <properties> element containing a set of properties that are to be set for the specified item.

```

<updateProperties>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>2468</objectId>
  <changeToken>gsfhdhdhdfg</changeToken>

```

```
<properties>
  <propertyDateTime name="TravelEndDate">
    <value>2008-03-06T19:00:00Z</value>
  </propertyDateTime>
  <propertyString name="Name">
    <value>New Object Name</value>
  </propertyString>
</properties>
</updateProperties>
```

The output message for this service contains one element, <objectId>, the Id corresponding to the updated Document (which MAY be the same ID provided in the input message).

```
<updatePropertiesResponse>
  <objectId>Id</objectId>
</updatePropertiesResponse>
```

moveObject

Description	Moves specified fileable object from a source folder to a target folder
--------------------	---

If the repository does not support multi-filing then the input message takes the following form:

```
<moveObject>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>DocumentAId</objectId>
  <targetFolderId>newFolderId</targetFolderId>
</moveObject>
```

If the repository does support multi-filing then the input message may take the following form:

```
<moveObject>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>DocumentAId</objectId>
  <targetFolderId>newFolderId</targetFolderId>
  <sourceFolderId>oldFolderId</sourceFolderId>
</moveObject>
```

deleteObject

Description	Deletes the specified object
--------------------	------------------------------

The input message takes the following form:

```
<deleteObject>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>ToBeDeletedObjectId</objectId>
```

```
</deleteObject>
```

deleteTree

Description	Deletes specified folder and the entire sub-tree
--------------------	--

The input message contains a folder Id identifying the sub-tree to delete.

```
<deleteTree>
  <repositoryId>MyRepositoryId</repositoryId>
  <folderId>FolderId</folderId>
  <unfileNonfolderObjects>unfile</unfileNonfolderObjects>
  <continueOnFailure>true</continueOnFailure>
</deleteTree>
```

The output message includes a <failedToDelete> element. In the event that the service fails to delete the entire tree this element will contain a set of object Ids in the following form.

```
<deleteTreeResponse>
  <failedToDelete>
    <objectId>objectAId</objectId>
    <objectId>objectBId</objectId>
  </failedToDelete>
</deleteTreeResponse>
```

If all objects were deleted then the <failedToDelete> element will be empty.

```
<deleteTreeResponse>
  <failedToDelete/>
</deleteTreeResponse>
```

setContentStream

Description	Sets (creates or replaces) the content stream for the specified document object.
--------------------	--

The HTTP request is multi-part, the first part containing the SOAP envelope containing the <setContentStream> element which contains two parameters as well as an <xop:Include> element with a value that is the ID of the associated second part to the response. The second part will contain the binary stream. The HTTP request will have the following form:

```
POST /CMIS/SOAP HTTP/1.1
Content-Length: 8345
SOAPAction: ""
```

```

Content-Type: Multipart/Related; type="application/xop+xml"; boundary="-----_Part_0_14615608.1167536350647"; start-info="text/xml"
Accept: text/xml, application/xop+xml, text/html, image/gif, image/jpeg, *, q=.2, */*; q=.2
User-Agent: Java/1.5.0_09
Host: localhost:8989
Connection: keep-alive

-----_Part_0_14615608.1167536350647
Content-Type: application/xop+xml; type="text/xml"; charset=utf-8

<?xml version="1.0" ?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ns1="http://catalog.mtom.company.com/book"
  xmlns:xop="http://www.w3.org/2004/08/xop/include">
  <soapenv:Body>
    <setContentStream>
      <repositoryId>MyRepositoryID</repositoryId>
      <documentId>documentAId</documentId>
      <overwriteFlag>true</overwriteFlag>
      <contentStream>
        <length>234567</length>
        <mimeType>application/msword</mimeType>
        <filename>myExpenseReport.doc</filename>
        <stream>
          <xop:Include>
            xmlns:xop=http://www.w3.org/2004/08/xop/include
            href="cid:1d0353b6-0ec7-4264-b4f4-16c1e6a0812c@www.acme.com">
          </xop:Include>
        </stream>
      </contentStream>
    </setContentStream>
  </soapenv:Body>
</soapenv:Envelope>
-----_Part_0_14615608.1167536350647
Content-Type: application/octet-stream
Content-ID: <1d0353b6-0ec7-4264-b4f4-16c1e6a0812c@www.acme.com>
Content-transfer-encoding: binary

.....(binary stream omitted)
-----_Part_0_14615608.1167536350647

```

The output message for this service contains one elements, <documentId>, the Id corresponding to the updated Document (which MAY be the same ID provided in the input message).

```

<setContentStreamResponse>
  <documentId>ID</documentId>
</setContentStreamResponse>

```

deleteContentStream

Description	<p>Deletes the content stream of the specified document. This does not delete the document object or its properties.</p> <p>This does not change the ID of the document.</p>
--------------------	--

The input message for this service is as below. The output message will be empty.

```
<deleteContentStream>
  <repositoryId>MyRepositoryId</repositoryId>
  <documentId>documentAId</documentId>
</deleteContentStream>
```

MULTI-FILING SERVICES

The MultiFilingService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

addObjectToFolder

Description	<p>Adds an existing, fileable 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.</p>
--------------------	---

The input message for this service is as below. The output message will be empty

```
<addObjectToFolder>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>documentAId</objectId>
  <folderId>folderXId</folderId>
</addObjectToFolder>
```

removeObjectFromFolder

Description	<p>Removes a child object from a folder. This does not delete the object and does not change its ID.</p>
--------------------	--

The input message for this service contains at least an <objectId> element and optionally a <folderId> element. The output message will be empty.

```
<removeObjectFromFolder>
```

```
<repositoryId>MyRepositoryId</repositoryId>
<objectId>ObjectId</objectId>
<folderId>FolderId</folderId>
</removeObjectFromFolder>
```

DISCOVERY SERVICES

The DiscoveryService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

query

Description	
	Queries the repository for queryable objects based on properties and/or using full-text search. Query returns objects that matches the qualification. Content streams are not returned as part of query.

This service accepts an input message containing a query statement and produces an output message containing a list of <object> elements followed by a <hasMoreItems> element, which is the same format as that for the getChildren() service. The result set may contain objects of any queryable type.

The result set is always a flat list of <object> elements. The set of properties returned for each object is specified in the SELECT clause of the Statement that is input.

VERSIONING SERVICES

The VersioningService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

checkOut

Description	
	Create a private working copy of the document, 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 the document is checked-out.

The input message for this service is as below.

```
<checkOut>
  <repositoryId>MyRepositoryId</repositoryId>
  <documentId>DocumentAId</documentId>
</checkOut>
```

The output message for this service contains the document ID of the private working copy, and indicates whether content-stream is copied to the PWC.

```
<checkOutResponse>
  <documentId>PrivateWorkingCopyAId</documentId>
  <contentCopied>true</contentCopied>
</checkOutResponse>
```

cancelCheckOut

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

The input message for this service is as below. The output message will be empty.

```
<cancelCheckOut>
  <repositoryId>MyRepositoryId</repositoryId>
  <documentId>PrivateWorkingCopyAId</documentId>
</cancelCheckOut>
```

checkIn

Description	
	Makes the private working copy the current version of the content.

The input message for this service has the following form. The repository MAY enforce constraints on the object properties or the content and if any of those constraints are violated the object will not be checked in and the service will return an error. Some input properties, if provided, such as *ObjectId*, *Uri* and *CreationDate* MAY be ignored by the repository. The example shown here does not include a content stream. For an example how to include the content stream as an input, see the `<setContentStream>` method description.

```
<checkIn>
  <repositoryId>MyRepositoryId</repositoryId>
  <documentId>documentAId</documentId>
  <major>true</major>
  <properties>
    <propertyDateTime name="TravelEndDate">
      <value>2008-03-06T17:00:00Z</value>
    </propertyDateTime>
    <propertyId name="ObjectId">
      <value>probably ignored</value>
    </propertyId>
    <propertyId name="ObjectTypeId">
      <value>ExpenseReportTypeId</value>
    </propertyId>
  </properties>
</checkIn>
```

```
</properties>
  <checkinComment>Draft</checkinComment>
</checkin>
```

The output message contains a single <documentId> element.

```
<checkinResponse>
  <documentId>documentBId</documentId>
</checkinResponse>
```

getPropertiesOfLatestVersion

Description	Returns the properties of the latest version, or the latest major version, of the specified version series
--------------------	--

The input message for this service is as below.

```
<getPropertiesOfLatestVersion>
  <repositoryId>MyRepositoryId</repositoryId>
  <versionSeriesId>VersionSeriesAId</versionSeriesId>
  <majorVersion>true</majorVersion>
  <filter>...</filter>
</getPropertiesOfLatestVersion>
```

The output message is the same as that for the getProperties() service, containing a single <object> element.

getAllVersions

Description	Returns the list of all members of the version series for the specified document, sorted by CREATION_DATE descending.
--------------------	---

The format of the output message for this service is the same as that for getDescendants() service, containing a list of <object> elements. The order in which these <object> elements are returned is repository-specific.

deleteAllVersions

Description	Deletes all documents in the specified version series.
--------------------	--

The input message for this service is as below. The output message will be empty.

```
<deleteAllVersions>
  <repositoryId>MyRepositoryId</repositoryId>
  <versionSeriesId>VersionSeriesId</versionSeriesId>
</deleteAllVersions>
```

RELATIONSHIP SERVICES

The RelationshipService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

getRelationships

Description	Returns a list of relationships associated with the object.
-------------	---

The input message for this service has the following form.

```
<getRelationships>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>objectAId</objectId>
  <direction>source</direction>
  <typeId>relationshipTypeId</typeId>
  <includeSubRelationshipTypes>true</includeSubRelationshipTypes>
  <filter>ObjectId, Uri, ObjectTypeId, ...</filter>
</getRelationships>
```

The format of the output message for this service is the same as that for the getChildren() service, containing a list of <object> elements followed by a <hasMoreItems> element. The order in which the <object> elements are returned is repository-specific. The set of properties returned for each relationship object is specified in the filter that is input and the repository MAY return more properties than those requested.

POLICY SERVICES

The PolicyService.wsdl, cmisMessageTypes.xsd and CMIS.xsd files fully describe this service.

applyPolicy

Description	Applies a policy to a controllable object.
-------------	--

The input message for this service is as below. The output message will be empty

```
<applyPolicy>
  <repositoryId>MyRepositoryId</repositoryId>
```

```
<policyId>PolicyXId</policyId>
  <objectId>ObjectAId</objectId>
</applyPolicy>
```

removePolicy

Description	Removes a previously applied policy from an object. The policy object is not deleted, and may still be applied to other objects.
--------------------	--

The input message for this service is as below. The output message will be empty.

```
<removePolicy>
  <repositoryId>MyRepositoryId</repositoryId>
  <policyId>PolicyXId</policyId>
  <objectId>ObjectAId</objectId>
</removePolicy>
```

getAppliedPolicies

Description	Returns the list of policy objects currently applied to the specified object.
--------------------	---

The input message for this service is as below.

```
<getAppliedPolicies>
  <repositoryId>MyRepositoryId</repositoryId>
  <objectId>ObjectAId</objectId>
  <filter>ObjectId, ObjectTypeId, PolicyName</filter>
</getAppliedPolicies>
```

The format of the output message for this service is the same as that for the getDescendants() service, consisting of a list of <object> elements. The order in which these <object> elements are returned is repository-specific. The set of properties returned for each policy object is specified in the filter that is input and the repository MAY return more properties than those requested.