HydroShare Web Service Interface Design

Jeffery S. Horsburgh

11-20-2013

1. Introduction

HydroShare will expose Application Programmer Interfaces (APIs) that support interaction between client applications and the main HydroShare system and facilitate development of these client applications. HydroShare will also expose web services that conform to the DataONE web service specifications defined in the DataONE architectural documentation (DataONE, 2013) so that HydroShare can become a DataONE Member Node as described in the HydroShare Data Management Plan. In general, HydroShare web services will be implemented using a Representational State Transfer (REST) based approach using HTTP as the transport protocol and XML for encoding messages. The HydroShare web service APIs will expose the same functionality that can be accomplished through the HydroShare web interface.

2. Authentication and Authorization

Some API functions require authorization. The HydroShare web service APIs will use the same authorization functions and configuration as the web interface. If a user is authorized to do something in the web interface, they will be authorized to do it via the API as well.

NEED AN AUTHENTICATION MECHANISM FOR THE WEB SERVICE API. PERHAPS AN API KEY IN THE HEADER OF THE HTTP REQUEST? ALSO NEED THE UNDERLYING AUTHORIZATION SYSTEM THAT EVALUATES API REQUESTS AND DETERMINES IF A USER IS AUTHORIZED.

3. Web Service Interface Definitions

The following sections describe the HydroShare web service APIs. The APIs will be versioned, and users will be able to specify the version number in the URL of their REST requests.

3.1. HydroShare Resource Management API

The table below lists the REST URLs that will be implemented as part of the HydroShare Resource Management API.

Table ??. HydroShare Resource Management API URLs and methods.

Path	Method	Description
GET /	HydroShare.getCapabilities()	Returns a document

Source: DataONE		describing the capabilities of HydroShare.
GET /object/{pid} Source: DataONE, CKAN	HydroShare.get()	Retrieve an object identified by the pid from HydroShare. The response must contain the bytes of the indicated object, and the checksum of the bytes retrieved should match the checksum recorded in the system metadata for that object. If the object does not exist in HydroShare, then **Exceptions.NotFound** must be raised. Objects can be any unit of content within HydroShare that has been assigned a pid, including resources and resource science metadata documents. This method came from DataONE, which assigns identifiers to both data objects and science metadata objects. So, one GET method by identifier can retrieve either one. If HydroShare does not assign unique identifiers to science metadata, then we need a separate getScienceMetadata method that would take the pid of the data object as input.
GET /meta/{pid} Source: DataONE	HydroShare.getSystemMetadata()	Describes the object identified by the pid by returning the associated system metadata object. If the object does not exist, then **Exceptions.NotFound** must be raised.
GET /revisions/{pid}	HydroShare.getRevisions()	Returns a list of identifiers for Resources that are

Source: CKAN		revisions of the Resource identified by the specified pid.
GET /related/{pid} Source: CKAN	HydroShare.getRelated()	Returns a list of identifiers for Resources that are related to the Resource identified by the specified pid.
GET /checksum/{pid} [?checksumAlgorithm= {checksumAlgorithm}] Source: DataONE	HydroShare.getChecksum()	Returns a checksum for the specified object using an accepted algorithm. The result is used to determine if two instances referenced by a pid are identical.
POST /object/{pid} Source: DataONE, CKAN	HydroShare.create()	Called by a client to add a new object to HydroShare. The pid must not exist in HydroShare or should have been previously reserved using **HydroShare.generateIdentifier()**. The caller must have authorization to write content to HydroShare. Another option is to have this method work without a PID. The PID would be assigned by HydroShare upon inserting the Resources. In this context, this method would return the newly-assigned identifier. This would not be consistent with DataONE.
PUT /object/{pid} Source: DataONE, CKAN	HydroShare.update()	This method is called by clients to update objects in HydroShare. Updates an existing object by creating a new object identified by *newPid* in HydroShare that explicitly obsoletes the object identified by *pid* through appropriate changes to the

		SystemMetadata of *pid* and *newPid*. The *pid* of the object being obsoleted is passed in as a parameter, and HydroShare should record the update by storing the **SystemMetadata.obsoletes** and **SystemMetadata.obsoletedBy** fields for the respective objects in their system metadata. HydroShare MUST check or set the values of **SystemMetadata.obsoletes** and **SystemMetadata.obsoletedBy** so that they accurately represent the relationship between the new and old objects. If the client sets these values and they are incorrect, then an **InvalidSystemMetadata** MUST be raised. HydroShare MUST also set **SystemMetadata.dateSysMetadataModified**. The modified system metadata entries must then be available in **HydroShare.listObjects()** to ensure that any cataloging systems pick up the changes when filtering on **SystmeMetadata.dateSysMetadataModified**.
POST /generate Source: DataONE	HydroShare.generateIdentifier()	Generates an identifier (pid) that is unique for use with a new object to be submitted to HydroShare. Effectively reserves an identifier and prevents it from further use.

		This method would not be needed if we decide not to have the create() method require an Identifier. This would not be consistent with DataONE.
DELETE /object/{pid} Source: DataONE, CKAN	HydroShare.delete()	Deletes an object managed by HydroShare. The caller must be authorized to perform this function. The operations removes the object from further interaction with HydroShare services and interfaces. The implementation may delete the object bytes, and should do so since a delete operation may be in response to a problem with the object (e.g., it contains malicious content, is inappropriate, or is subject to a legal request). If the object does not exist, the **Exceptions.NotFound** exception is raised.

3.2. HydroShare User Management and Authorization API

The table below lists the REST URLs that will be implemented as part of the HydroShare user management and authorization API.

Table ??. HydroShare user management and authorization API URLs and methods.

Path	**Method**	**Description**
GET /isAuthorized/{pid}?action={action} Source: DataONE	HydroShare.isAuthorized()	Test if the user identified by the provided session is authorized for operation on the specific object. A successful operation is indicated by a return HTTP status of 200. Failure is indicated by an exception such as **NotAuthorized** being returned.
PUT /owner/{pid}	HydroShare.setRightsHolder()	Changes ownership of the

Source: DataONE, CKAN		specified object to the subject specified by a userID.
PUT /accessRules/{pid} Source: DataONE, CKAN	HydroShare.setAccessPolicy()	Set the access permissions for an object identified by pid. Triggers a change in the system metadata. Successful completion of this operation in indicated by a HTTP response of 200. Unsuccessful completion of this operation must be indicated by returning an appropriate exception such as **NotAuthorized**. May need more methods for adding and removing users and groups.
POST /accounts Source: DataONE, CKAN	HydroShare.registerAccount()	Create a new user within the HydroShare system.
PUT /accounts/{subject} Source: DataONE, CKAN	HydroShare.updateAccount()	Update an existing subject within the HydroShare system. The target subject is given in the URL. The user calling this method must have write access to the account details.
GET /accounts/{subject} Source: DataONE	HydroShare.getSubjectInfo()	Get the information about a subject. For a user this would be their profile information, groups they belong to, etc. For a group this would be its description and membership list.
GET /accounts?query={query}[&status={status}&start={start}&count={count}] Source: DataONE	HydroShare.listSubjects()	List the subjects, including users, groups, and systems that match search criteria.
POST /groups Source: DataONE,	HydroShare.createGroup()	Create a group with a given name. Groups are lists of subjects that allow all

CKAN		members of the group to be referenced by listing soley the subject name of the group. Group names must be unique within HydroShare. Groups can only be modified by Subjects listed as rightsholders.
PUT /groups Source: DataONE, CKAN	HydroShare.updateGroup()	Add or remove members to/from the named group. Group members can be modified only by the original creator of the group, otherwise a NotAuthorized exception is throw. Group members are provided as a list of subjects that replace the group membership. Successful completion of this operation is indicated by a response status code of 200. Unsuccessful completion must be indicated by returning an appropriate exception.
GET /groupResources/{subject] Source: CKAN	HydroShare.getGroupResources()	Return a list of Resources that have been shared with a group.

3.3. HydroShare Data Discovery API

The table below lists the REST URLs that will be implemented as part of the HydroShare Data Discovery API.

Table ??. HydroShare data discovery API URLs and methods.

Path	**Method**	**Description**
GET /object [?fromDate={fromDate} &toDate={toDate} &formatId={formatId} &replicaStatus={replicaStatus} &start={start}&count={count}]	HydroShare.listObjects()	Retrieve the list of objects in HydroShare that match the calling parameters. This method is required to support cataloging of objects contained within HydroShare. At a minimum, this method

Source: DataONE		must be able to return a list of objects that match fromDate < SystemMetadata.dataSysMetadataModified but is expected to also support a date range.
GET /resourceTypes Source: DataONE	HydroShare.listResourceTypes()	Returns a list of all resource types registered in the HydroShare Resource Type Vocabulary.
GET /formats Source: DataONE	HydroShare.listFormats()	Returns a list of all object formats registered in the HydroShare Object Format Vocabulary
GET /search/{queryType}/{query} Source: DataONE	HydroShare.search()	Search the HydroShare metadata catalog and return a list of identifiers for resources that match the criteria. Search may be implemented using more than one type of search engine. The queryType parameter indicates which search engine should be targeted. The value and form of query is determined by the search engine.
GET /search Source: DataONE	HydroShare.listSearchEngines()	Returns a list of search engines supported by HydroShare – i.e., the supported values for the queryType parameter in the search method.

3.4. HydroShare Social API

The table below lists the REST URLs that will be implemented as part of the HydroShare Social API.

Table ??. HydroShare social API URLs and methods.

Path	Method	Description
POST /rating/{pid}	HydroShare.createRating()	Create a rating for a Resource in

Source: CKAN		HydroShare identified by pid.
POST /followSubject/{subject} Source: CKAN	HydroShare.followSubject()	Start following a user or group identified by subject.
POST /followResource/{pid} Source: CKAN	HydroShare.followResource()	Start following a Resource.
DELETE /followSubject/{subject}	HydroShare.followSubject()	Stop following a user or group identified by subject
DELETE /followResource/{pid}	HydroShare.followResource()	Stop following a Resource
POST /annotation/{pid}	HydroShare.annotateResource()	Create a comment about a resource

4.5. HydoShare DataONE API

The HydroShare Data management plan states that HydroShare will implement the DataONE Member Node APIs and become a DataONE Member Node. Most of the methods in the previous sections have equivalent methods in the DataONE Member Node APIs that could use the same internal methods. HydroShare will have to implement that additional methods described in the DataONE API documentation to become a fully compliant DataONE Member Node (see http://mule1.dataone.org/ArchitectureDocs-current/apis/MN_APIs.html#).

References

DataONE (2013). DataONE APIs. http://mule1.dataone.org/ArchitectureDocs-current/apis/index.html.

Open Knowledge Foundation (2013). The CKAN API. http://docs.ckan.org/en/latest/api.html