# Data Registration from AWS S3 – Software Design Document

# API Design

HPC-DME provides an API to register a single data object, and an API to perform bulk registration, i.e. register multiple files via a single request. In both cases, physical file needs to be transferred to HPC-DME archive and the users can either attach it to the API request, provide a location of the file on a Globus endpoint, or request an upload URL – in that case, users will use the upload URL to upload the file following the completion of the registration API.

A new requirement is to provide ability to register and transfer files from AWS S3 bucket into HPC-DME archive. This document provides a proposal of design change to support the new requirements.

Note: HPC-DME supports 2 types of archives – Cleversafe and POSIX (file system). Registering from AWS S3 is to be supported by Cleversafe archive only.

## Single File Registration

The current API supports 3 types of uploading a single file – synchronized, async from Globus, and async via S3 upload URL.

The design proposal is to make an API change to support registering from AWS S3. The new API will be provided as a ‘v2’ of the API (i.e. v2 will be added to the API url). The existing API will be deprecated and continue to work for a while until support is removed.

The following sections will provide the details of the existing vs. the new (v2) proposed API for each type of single file registration.

### Synchronized Upload (Current)

The caller attaches a data file (it’s a multipart REST call), and a JSON file:

{

"callerObjectId" : "<user-defined-base-path-in-archive (optional) >",

"checksum" : "<data-checksum (optional) >",

"metadataEntries": [ <array-of-metadata> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy>

}

}

This API is supported by both Cleversafe and POSIX archives.

### Synchronized Upload (V2)

In V2 – the exact same API will be supported – no proposed change.

### Asynchronous Upload from Globus (Current)

The caller attaches just a JSON file to the request:

{

"source": {

"fileContainerId": "<Globus-Endpoint-ID>",

"fileId": "<Globus-Path>"

},

"callerObjectId" : "<user-defined-base-path-in-archive (optional) >",

“metadataEntries": [ <array-of-metadata> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

### Asynchronous Upload from Globus (V2)

The caller attaches just a JSON file to the request:

{

"globusUploadSource": {

"sourceLocation": {

"fileContainerId": "<Globus-Endpoint-ID>",

"fileId": "<Globus-Path>"

}

},

"callerObjectId" : "<user-defined-base-path-in-archive (optional) >",

“metadataEntries": [ <array-of-metadata> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

### Asynchronous Upload w/ an upload URL (Current)

The caller attaches just a JSON file to the request:

{

"generateUploadRequestURL": true

"callerObjectId" : "<user-defined-base-path-in-archive (optional) >",

“metadataEntries": [ <array-of-metadata> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

This API is supported for Cleversafe archive only. In a successful registration, the generated upload URL is returned to the caller. The caller will use that URL to upload the data directly into Cleversafe (via curl or any other way they wish).

### Asynchronous Upload w/ an upload URL (V2)

In V2 – the exact same API will be supported – no proposed change.

### Asynchronous Upload from AWS S3 (V2)

The caller attaches just a JSON file to the request:

{

"s3UploadSource": {

"sourceLocation": {

"fileContainerId": "<S3-bucket-name>",

"fileId": "<S3-object-key>"

},

"account" : {

"accessKey" : "<aws-access-key>",

"secretKey" : "<aws-secret-key>",

"region" :"<aws-region>"

}

},

"callerObjectId" : "<user-defined-base-path-in-archive (optional) >",

“metadataEntries": [ <array-of-metadata> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

## Bulk Data Object Registration

HPC-DME provides an API to perform bulk registration. The callers can provide a list of individual data objects to register, and/or a list of Globus endpoints to scan:

{

"uiURL" : "<base-url-to-HPC-ui (Optional)",

"dryRun" : "<if true, no actual registration is performed (Optional), false by default>",

"dataObjectRegistrationItems": [ <individual-file-registration-requests> ],

"directoryScanRegistrationItems": [ <individual-directory-scan-requests> ]

}

The design proposal is to make an API change to support bulk registration from AWS S3. The new API will be provided as a ‘v2’ of the API (i.e. v2 will be added to the API url). The existing API will be deprecated and continue to work for a while until support is removed.

In order to simplify the design proposal, the following sections will show the current and proposed design of a single file registration item and directory scan item. The API as shows above will continue to support having an array of both in a single request.

Please note that currently bulk data registration is only performed from Globus (no bulk registration using sync upload or S3 upload URL).

### Data Object Registration Item from Globus (Current)

{

"path": <registration-path-in-irods>",

"source": {

"fileContainerId": "<Globus-Endpoint-ID>",

"fileId": "<Globus-Path>"

},

"callerObjectId": "<user-defined-base-path-in-archive (optional) >",

"dataObjectMetadataEntries": [ <registered-file-metadata-list> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

### Data Object Registration Item from Globus (V2)

{

"path": <registration-path-in-irods>",

"globusUploadSource": {

"sourceLocation": {

"fileContainerId": "<Globus-Endpoint-ID>",

"fileId": "<Globus-Path>"

}

},

"callerObjectId": ""<user-defined-base-path-in-archive (optional) >",

"dataObjectMetadataEntries": [ <registered-file-metadata-list> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

### Data Object Registration Item from AWS S3 (V2)

{

"path": <registration-path-in-irods>",

"s3UploadSource": {

"sourceLocation": {

"fileContainerId": "<S3-bucket-name>",

"fileId": "<S3-object-key>"

},

"account" : {

"accessKey" : "<aws-access-key>",

"secretKey" : "<aws-secret-key>",

"region" :"<aws-region>"

}

},

"callerObjectId": "<user-defined-base-path-in-archive (optional) >",

"dataObjectMetadataEntries": [ <registered-file-metadata-list> ],

"createParentCollections" : <true|false> (optional, false by default)>

"parentCollectionsBulkMetadataEntries" : {

<metadata-to-register-collections-up-the-hierarchy}

}

}

### Directory Scan Registration Item from Globus (Current)

{

"basePath": "<base-registration-path-in-irods”,

"scanDirectoryLocation": {

"fileContainerId": "<Globus-Endpoint-ID>",

"fileId": "<Globus-Directory-Path>"

},

"callerObjectId": ""<user-defined-base-path-in-archive (optional) >",

"includePatterns": [<scan-include-patterns>],

"excludePatterns": [<scan-exclude-patterns>"],

"patternType": [<SIMPLE|REGEX (Optional) SIMPLE by default>"],

"pathMap" : {

"fromPath": "<from-scanned-path>",

"toPath": "<to-registered-path>"

},

"bulkMetadataEntries": {

"pathsMetadataEntries": [<metadata-for-collection-or-files>],

"defaultCollectionMetadataEntries": [<override-system-default-metadata>]

}

}

### Directory Scan Registration Item from Globus (V2)

{

"basePath": "<base-registration-path-in-irods”,

"globusScanDirectory": {

"directoryLocation": {

"fileContainerId": "<Globus-Endpoint-ID>",

"fileId": "<Globus-Directory-Path>"

}

},

"callerObjectId": ""<user-defined-base-path-in-archive (optional) >",

"includePatterns": [<scan-include-patterns>],

"excludePatterns": [<scan-exclude-patterns>"],

"patternType": [<SIMPLE|REGEX (Optional) SIMPLE by default>"],

"pathMap" : {

"fromPath": "<from-scanned-path>",

"toPath": "<to-registered-path>"

},

"bulkMetadataEntries": {

"pathsMetadataEntries": [<metadata-for-collection-or-files>],

"defaultCollectionMetadataEntries": [<override-system-default-metadata>]

}

}

### Directory Scan Registration Item from AWS S3 (V2)

{

"basePath": "<base-registration-path-in-irods”,

"s3ScanDirectory": {

"directoryLocation": {

"fileContainerId": "<S3-bucket-name>",

"fileId": "<S3-Directory-key>"

},

"account" : {

"accessKey" : "<aws-access-key>",

"secretKey" : "<aws-secret-key>",

"region" :"<aws-region>"

}

},

"callerObjectId": ""<user-defined-base-path-in-archive (optional) >",

"includePatterns": [<scan-include-patterns>],

"excludePatterns": [<scan-exclude-patterns>"],

"patternType": [<SIMPLE|REGEX (Optional) SIMPLE by default>"],

"pathMap" : {

"fromPath": "<from-scanned-path>",

"toPath": "<to-registered-path>"

},

"bulkMetadataEntries": {

"pathsMetadataEntries": [<metadata-for-collection-or-files>],

"defaultCollectionMetadataEntries": [<override-system-default-metadata>]

}

}

# Detailed Design

This section provides the detailed design of the changes to code base to support the new functionality to register files from AWS S3.

## Rest API

* **hpc-dto/HpcDataManagement.v2.xsd**
  + The model (DTO) for the V2 Data Registration API needs to be added based on the API design in the section above.
* **hpc-ws-rs-api/HpcDataManagementRestService.java**
  + Add registerDataObject() method based on the API design above, and bind it to a /v2 endpoint. This method implements the single file registration API.
  + Add registerDataObjects() method based on the API design above, and bind it to a /v2 endpoint. This method implements the bulk registration API.
  + Add @Deprecate to the existing registration methods.
* **hpc-ws-rs-impl/HpcDataManagementRestServiceImpl.java**
  + Implement the 2 new API methods, using the V2 model.
  + Modify the existing API methods to convert the old model to the new V2 model and call the new methods.

## Business / Application Services

* **hpc-bus-service-api/HpcDataManagementBusService.java**
  + Change the registerDataObject() and registerDataObjects() methods to use the new V2 model classes.
* **hpc-bus-service-impl/HpcDataManagementBusServiceImpl.java**
  + Implement the registerDataObject() and registerDataObjects() methods to use the new V2 model classes.
* **hpc-domain-types/HpcDataTransferTypes.xsd**
  + Change the definition of HpcDataObjectUploadRequest to support Globus and S3 source locations
* **hpc-app-service-api/HpcDataTransferService.java**
  + Change the uploadDataObject() method to accept globus and s3 source locations.
* **hpc-app-service-impl/HpcDataTransferServiceImpl.java**
  + Implement the uploadDataObject() method to use Globus and S3 source locations

## Integration

* **hpc-integration-impl/s3.impl/HpcDataTransferProxyImpl.java**
  + Implement uploadDataObject() for AWS S3
  + Implement scanDirectory() for AWS S3

## DAO

* **hpc-dao-impl/HpcDataRegistrationDAOImpl.java**
  + Update upsertBulkDataObjectRegistrationRequest() and getBulkDataObjectRegistrationRequest() to persist AWS S3 requests (including encrypted account information)