# Data Transfer from DME to Glacier – Software Design Document

Currently the DME system supports archival of data sets into an S3 archive (Cleversafe, Cloudian, AWS S3 – if configured.) With the need to move the data from DME Archive to AWS Glacier, DME system will be enhanced with additional API to support archiving to AWS Glacier, retaining and adding additional metadata (deep\_archive\_date, data\_transfer\_status – DEEP\_ARCHIVE) in iRODS, and submitting a restoration request for restoring an object from Glacier. The detailed requirements can be found [here](https://collaborate.nci.nih.gov/x/zEIKDw).

# API Design

In order to leverage the existing permission framework for who can access the DME API to transfer the data from DME to AWS Glacier, a new endpoint is recommended instead of adding on to the existing registration API. There will be an endpoint for a single object transfer, collection transfer and bulk transfer. Direct transfer from file system to AWS S3/Glacier will be supported via the existing registration API with an AWS S3 bucket configured as the S3 Archive and the lifecycle policy configured on the AWS S3 bucket.

## Existing Single File Registration

No changes required for existing Single File registration. With the AWS S3 bucket configured as the S3 archive with lifecycle policy to transition to Glacier, single file registration directly to Glacier Archive will be supported.

* Synchronized Upload to Glacier via AWS S3 bucket
* Asynchronous Upload from Globus to Glacier via AWS S3 bucket
* Asynchronous Upload from User’s AWS S3 to Glacier via AWS S3 bucket
* Asynchronous Upload from Google Drive to Glacier via AWS S3 bucket

## Bulk Data Object Registration

No changes required for existing Bulk Data Object registration. With the AWS S3 bucket configured as the S3 archive with lifecycle policy to transition to Glacier, bulk data object registration to Glacier Archive will be supported.

* Data Object Registration Item from Globus to Glacier via AWS S3 bucket
* Directory Scan Registration Item from Globus to Glacier via AWS S3 bucket
* Data Object Registration Item from User’s AWS S3 to Glacier via AWS S3 bucket
* Directory Scan Registration Item from User’s AWS S3 to Glacier via AWS S3 bucket
* Data Object Registration Item from Google Drive to Glacier via AWS S3 bucket
* Directory Scan Registration Item from Google Drive to Glacier via AWS S3 bucket

## Single File Transfer from DME to Glacier (New API)

* Data Object Transfer from DME to Glacier

## Collection Transfer from DME to Glacier (New API)

* Collection Transfer from DME to Glacier

## Bulk Data Objects Transfer from DME to Glacier (New API)

* Bulk Data Objects or Collections Transfer from DME to Glacier

## Changes to Single File Download from Glacier to DME (Sync) or to User specified download location for Async

* The current download API will be modified to support initiating the restoration request from Glacier (For sync, the system will notify the user when the data is available for download. For async, after the file is restored, the file will be downloaded to the user specified location.)

## Changes to Collection, Collection or Data Object list download from Glacier to User specified download location for Async

* The current download API will be modified to initiate restoration requests for each Data Object which is in Glacier. After the file is restored, the file will be downloaded to the user specified location.

## Task Status API

* The existing Task Status API will be used to retrieve the Glacier archive request status for a File, Collection or Bulk Data Objects Tiering request. (They will be added to the existing Registration task table with request type TIER\_REQUEST)
* The existing Task Status API will be used to obtain the status of a File or Bulk File Restoration request. (They will be added to the existing Download task table with status RESTORE\_REQUESTED)

# DME Upload to Glacier

Diagram

Description automatically generated

# DME Download from Glacier

Diagram

Description automatically generated

# Detailed Design

This section provides the detailed design of the changes to code base to support the new functionality to tier data file and bulk data object transfer from DME to AWS Glacier.

## Rest API

* **hpc-dto/HpcDataManagement.xsd**
  + The model (DTO) for the Data Tiering API needs to be added based on the API design in the section above. Add HpcBulkDataObjectTierRequestDTO to specify the bulk data objects/collections. Add HpcTierResponseDTO to return a taskId.
* **hpc-ws-rs-api/HpcDataManagementRestService.java**
  + Add tierDataObject() method based on the API design above, and bind it to /dataObject/{path:.\*}/tier endpoint. This method implements the single file transfer API from DME to Glacier.
  + Add tierCollection() method based on the API design above, and bind it to /collection/{path:.\*}/tier endpoint. This method implements the collection tiering API from DME to Glacier.
  + Add tierDataObjectsOrCollections() method based on the API design above, and bind it to /tier endpoint. This method implements the dataobjects or collections tiering API from DME to Glacier.
* **hpc-ws-rs-impl/HpcDataManagementRestServiceImpl.java**
  + Implement the 3 new API methods, using the Data Tiering model.

## Business / Application Services

* **hpc-domain-types/HpcDataTransferTypes.xsd**
  + Add DEEP\_ARCHIVE to HpcDataTransferUploadStatus
  + Add RESTORE\_REQUESTED to HpcDataTransferDownloadStatus
* **hpc-domain-model/HpcDataManagement.xsd**
  + The model for the bulk request needs to be added based on the API design in the section above. Add HpcBulkTierRequest and HpcBulkTierItem to specify the bulk data objects/collections paths and its configuration ids. (config id is required for requests across multiple DOCS.)
* **hpc-bus-service-api/HpcDataManagementBusService.java**
  + Add tierDataObject(), tierCollection() and tierDataObjectsOrCollections() methods.
* **hpc-bus-service-impl/HpcDataManagementBusServiceImpl.java**
  + Implement the tierDataObject(). Validate the file is in ARCHIVED state, submits a tiering request via life cycle policy creation, record the request in registration task and return the task ID.
  + For collection, tierCollection() method, validate data object exists under this collection, submit a tiering request for the collection, record the request in registration task with the items list for all data objects under the collection and return the task ID.
  + For data objects list, tierDataObjectsOrCollections() method, loop through all data objects and create the lifecycle rule, then create a single task with the items.
  + For collection list, tierDataObjectsOrCollections() method, loop through all collections and create the lifecycle rule, then create a single task with the items list for all data objects under the list of collections.
* **hpc-app-service-api/HpcDataManagementService.java**
  + Add tierDataObjects() method.
* **hpc-app-service-impl/HpcDataManagementServiceImpl.java**
  + Implement the tierDataObjects() method for recording the tier task.
* **hpc-app-service-api/HpcDataTransferService.java**
  + Add tierDataObject(), tierCollection() tierDataObjects() and tierCollections() method.
* **hpc-app-service-impl/HpcDataTransferServiceImpl.java**
  + Implement the tierDataObject(), tierCollection(), tierDataObjects() and tierCollections() for transferring a data object from DME Archive to Glacier by creating life cycle policy.
  + Modify existing downloadDataObject() to initiate a restoration request and create data object download task if the transfer\_status is DEEP\_ARCHIVE.
* **hpc-app-service-impl/notificationFormats.json**
  + Add event type for RESTORE\_REQUEST\_COMPLETED.

## Integration

* **HpcDataTransferProxy**
  + Add method putLifecyclePolicy() and restoreDataObject()
* **hpc-integration-impl/s3.impl/HpcDataTransferProxyImpl.java**
  + Implement putLifecyclePolicy () for creating a lifecycle rule with filters
  + Implement restoreDataObject() for creating a restore request for data objects

## DAO

* **hpc-dao-impl/HpcDataRegistrationDAOImpl.java**
  + Modify upsertBulkDataObjectRegistrationTask () to account for task with null request object.

## Scheduler

* **hpc-scheduler/HpcScheduledTasksImpl.java**
  + Add completeTierTasks() to periodically check for storage class for each data object in the task and will update the iRODS data transfer type to DEEP\_ARCHIVE if storage class is Glacier.
  + Add completeRestoreRequestTask() to periodically check for restoration status and toggle the status to RECEIVED for async download and toggle the status to RESTORED and populate HPC\_EVENT to notify the user for sync download.

# Questions

* **What to do with data objects which are in Cleversafe bucket?**
  + We’ll need to find a solution which will work for other systems in the future as well. For immediate solution, we can include only data objects from Cloudian in the ITEMS list for tier request task.
* **Shall we record the life cycle policy we added to Cloudian in DB for reference or in case we need to remove it?**
  + We should keep a record for auditing purposes.
* **When to initiate a restoration request?**
  + Data transfer status is in DEEP\_ARCHIVE
  + No ongoing restoration request (metadata restore requested can be added when the restoration is initiated and removed when restored.)
  + File does not exist in Cloudian (Covered by second bullet)
* **Data integrity of Cloudian transfer to AWS S3 bucket.**