# 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, deep\_archive\_status – GLACIER, 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

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 leveraging the lifecycle policy to transition the objects to AWS S3. Once the object is transitioned to AWS S3, the lifecycle policy on AWS S3 bucket can transition the objects to Glacier or Glacier Deep Archive. 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

* A new table will be created to record the Lifecycle policy created for a File, Collection or Bulk Data Objects Tiering 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 for sync, and displayed as Restore In Progress)

# 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 restoreInProgress to HpcDataObjectDownloadResponseDTO, HpcDataObjectDownloadStatusDTO and add restoreInProgessItems to HpcCollectionDownloadStatusDTO to be used to display download that are in Restore in progress.
* **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 values IN\_PROGRESS, GLACIER, DEEP\_ARCHIVE and DELAYED to HpcDeepArchiveStatus
  + Add values TIER\_DATA\_OBJECT, TIER\_COLLECTION to HpcLifecycleRequestType
  + Add RESTORE\_REQUESTED to HpcDataTransferDownloadStatus
  + Add restoreInProgress to HpcDataObjectDownloadResponse, HpcDataObjectDownloadTask, HpcDownloadTaskResult, HpcCollectionDownloadTaskItem
  + Add object HpcS3ObjectMetadata to hold S3 metadata, checksum, restorationStatus and deepArchiveStatus
* **hpc-domain-types/HpcNotificationTypes.xsd**
  + Add RESTORE\_REQUEST\_COMPLETED/FAILED to HpcEventType for sync download notification
* **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.)
  + Add tieringBucket and tieringProtocol to HpcDataTransferConfiguration
* **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 lifecycle table and set the object deep\_archive\_status to IN\_PROGRESS.
  + For collection, tierCollection() method, validate data object exists under this collection, submit a tiering request for the collection, record the request in lifecycle table and set the deep\_archive\_status to IN\_PROGESS for all data objects under the collection.
  + For data objects list, tierDataObjectsOrCollections() method, loop through all data objects and create the lifecycle rule, record the request in lifecycle table and set the deep\_archive\_status to IN\_PROGESS for all data objects in the list.
  + For collection list, tierDataObjectsOrCollections() method, loop through all collections and create the lifecycle rule, record the request in lifecycle table and set the deep\_archive\_status to IN\_PROGESS for all data objects under the collections.
  + Modify generateDownloadRequestURL to check for restoration status for objects in deep archive before returning the presigned download URL.
* **hpc-bus-service-impl/HpcSystemBusServiceImpl.java**
  + Implement completeDeepArchiveInProgress().
  + Implement completeRestoreRequest().
* **hpc-app-service-api/HpcDataManagementService.java**
  + Add getDataObjectsDeepArchiveInProgress() method.
* **hpc-app-service-impl/HpcDataManagementServiceImpl.java**
  + Implement the getDataObjectsDeepArchiveInProgress() method for retrieving all IN\_PROGRESS data objects pending to transition to deep archive.
* **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 deep\_archive\_status is GLACIER or DEEP\_ARCHIVE.
* **hpc-app-service-impl/HpcEventServiceImpl.java**
  + Add addRestoreRequestCompletedEvent() and addRestoreRequestFailedEvent()
* **hpc-app-service-impl/HpcMetadataServiceImpl.java**
  + Modify updateDataObjectSystemGeneratedMetadata() to update deep\_archive\_status and deep\_archive\_date if supplied.
* **hpc-app-service-impl/notificationFormats.json**
  + Add notification template for event type RESTORE\_REQUEST\_COMPLETED and RESTORE\_REQUEST\_FAILED.

## Integration

* **HpcDataTransferProxy**
  + Add method getDataObjectMetadata(), putLifecyclePolicy(), restoreDataObject() and existsLifecyclePolicy()
* **hpc-integration-impl/s3.impl/HpcDataTransferProxyImpl.java**
  + Implement getDataObjectMetadata() to retrieve S3 Object metadata for restoration status and storage class.
  + Implement putLifecyclePolicy () for creating a lifecycle rule with filters
  + Implement restoreDataObject() for creating a restore request for data objects.
  + Implement existsLifecyclePolicy() to check if any lifecycle policy rule exists for a given archive file location.

## DAO

* **hpc-dao-impl/HpcDataDownloadDAOImpl.java**
  + Add getDataObjectDownloadTaskByStatus() to retrieve all download tasks with status RESTORE\_REQUESTED.
* **hpc-dao-api/HpcLifecycleDAO.java**
  + Create insert API to insert lifecycle rule into HPC\_S3\_LIFECYCLE\_RULE table.
* **hpc-dao-impl/HpcLifecycleDAOImpl.java**
  + Implement insert() method.
* **hpc-dao-impl/HpcDataManagementConfigurationDAOImpl.java**
  + Modify row mapper to include tieringBucket and tieringProtocol

## Scheduler

* **hpc-scheduler/HpcScheduledTasksImpl.java**
  + Add completeDeepArchiveInProgressTask() to periodically check for storage class for each data object in the task and will update the iRODS deep\_archive\_status to GLACIER or DEEP\_ARCHIVE if storage class is Glacier or Glacier Deep Archive. If it does not toggle in configured number of days, the status will be updated to DELAYED.
  + Add completeRestoreRequestTask() to periodically check for restoration status and toggle the status to RECEIVED for async download 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 toggle the deep\_archive\_status to IN\_PROGRESS for only data objects from Cloudian or AWS.
* **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?**
  + Deep archive status is in GLACIER or DEEP\_ARCHIVE.
  + No ongoing restoration request
  + File does not exist in Cloudian (Covered by the first two bullets)
* **Data integrity of Cloudian transfer to AWS S3 bucket.**
* **Glacier delete files**
  + For objects that are tiered to Glacier from S3, deleting the S3 will be sufficient to delete from Glacier. (Confirmed.)
* **Presigned URL allowed?**
  + Check for deep\_archive status and restoration status if it can be generated.
* **Once we move to Glacier, can we allow additional uploads to the collection?** 
  + Allow registration to occur and if the lifecycle rule exists, then add metadata deep\_archive\_status IN\_PROGRESS to be transitioned to GLACIER or DEEP\_ARCHIVE.
* **How do we know if data object is in GLACIER or transitioning?**
  + **Keep ARCHIVED status at data object level and add deep\_archive\_status IN\_PROGRESS.**
* **De-tiering?**
  + **Stop tiering the collection and/or move everything back from Glacier. Involves remove lifecycle rule, restore everything, and set storage class to Standard.**
  + **Needs further research on how Cloudian supports this and new APIs.**
* **Glacier vs. Glacier Deep Archive?** (x-amz-storage-class GLACIER vs. DEEP\_ARCHIVE)
  + Cloudian only allows us to specify GLACIER.
  + However, if we use the S3 protocol from Cloudian and create a lifecycle policy on AWS S3 bucket to transition to DEEP\_ARCHIVE, it is possible.
* **Known limitation when collection or data object is moved in iRODS : lifecycle rule will only apply if requested for the original collection.**
  + Data object gets created in collection abc -> abc/dataobect
  + Collection in irods moved to def but dataobject remain with abc prefix.