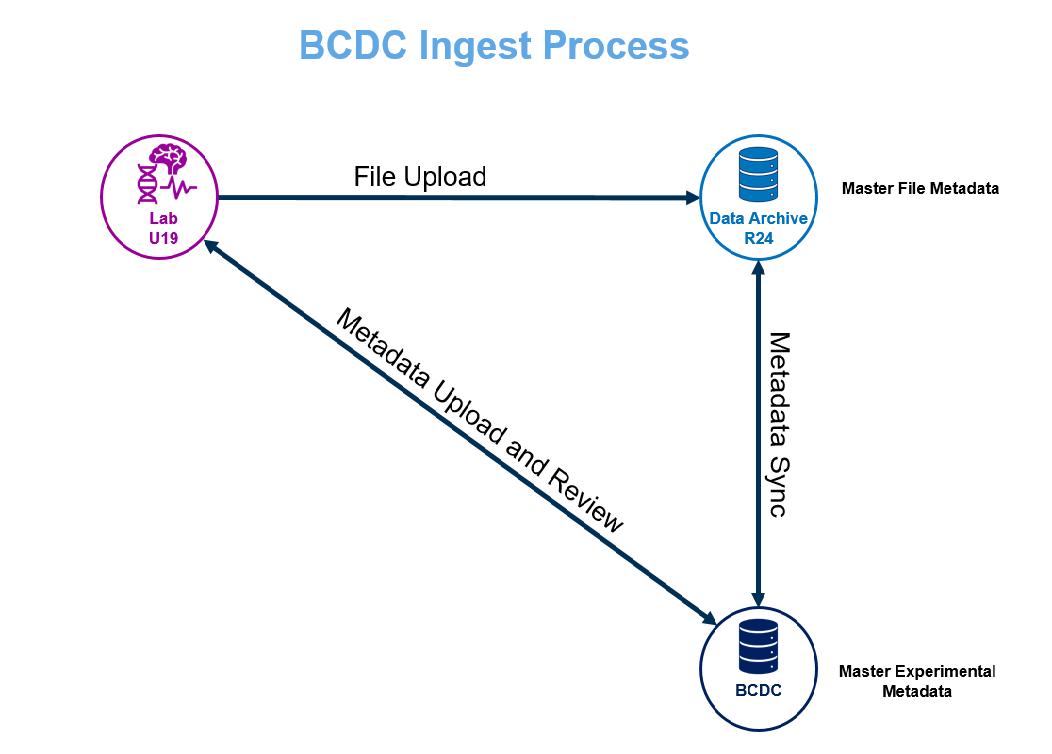
BCDC Ingest Proposal

This document will outline the interaction between the U01/U19, the R24, and BCDC.

In this document a “user” will represent a person from a U01/U19.



# Generic use cases:

## Use Case #1 – Initial Upload of Data

Users upload their file data and a file manifest into the R24. Users use either a command line or a web client to upload the experimental metadata to BCDC. The R24 can pull metadata from BCDC that corresponds to the file upload.

## Use Case #2 – Update of Metadata

Users verify the state of their metadata at BCDC using either a command line or web client. Users update metadata at BCDC their API client. Can verify that the metadata has been updated. The R24s can poll BCDC for metadata that has been added or updated since the last time they checked.

## Use Case #3 – Update of Files

Users may modify files at the R24, either by updating, adding, or deleting files at the R24. Users must update the file manifest to reflect the changes. Users must also modify the metadata at BCDC in order to reflect the changes at the R24.

# User Interfaces

## Command Line BCDC API Client

A command line client should be provided for programmatic ingest of metadata into BCDC.

## Web BCDC Client

A web client should also be provided to upload metadata into BCDC. This client should also be able to display what has already been uploaded into BCDC.

# User Artifact Uploads

## Ingest of Files into R24

Users are expected to upload their experimental data to the R24s. For each data modality, there is a R24 responsible for the repository of files associated with that experimental modality. For example, NeMO is responsible for holding the data related to the modalities of transcriptomics and epigenomics. Each R24 will define its own method for data ingest.

## Ingest Manifest of Files into R24

Along with the actual files, each user is expected to upload to each R24 a manifest of files that the user expects to be present at the R24. This manifest should contain for each file an identifier, a checksum, the size of each file. The identifier may be as simple as the file name or file path or a more complicated identifier. This is the identifier that will be used to look up the metadata associated with each file in BCDC.

## Ingest of Metadata into BCDC

Users are also expected to upload metadata about the experiment associated with the files to BCDC. The Ingest API will be a RESTful JSON API. Metadata that is uploaded into this API will be validated. Users may describe links between the various metadata objects that they upload (e.g. a virus used for an experiment). These linked objects must exist at the time of the upload or else a validation error will occur. Users will also be able to update metadata object that they have already uploaded.

# Behind the Scenes

## Metadata Object Retrieval by R24

The Ingest API will provide a means for an R24 to retrieve metadata to synchronize the R24 store of metadata. There will be a way to retrieve metadata objects. Useful filters will include filtering by object type, by updated date range, and by U01/U19. It may be the responsibility of the client of this API endpoint to retrieve all the linked objects to the result, so that the API doesn’t not have the responsibility of materializing all the links.

## R24 File Metadata API

The R24 will provide an API that, given a file identifier, will return the appropriate metadata about a file. This must include the URL for the file at the R24. This may also include the file name, checksum, size, and other attributes about the file.

## API Features

Authentication and Authorization must be implemented so that users can be identified and roles set for users to be able view and edit metadata. Namespaces should also be created to segment the data for each user. Both schemas and objects should be able to be found through the API. Objects should be validated with a schema. Objects should be linkable to other objects. Both objects and schema should be able to be versioned. Objects should be identified by a user-provided identifier.

## API Interface

The API interface definition will be detailed in another document.