Preservation Workflow for Imaging Optical Media

Table of Contents

Introduction

This document was drafted in Spring 2022 by Emma Clarkson as a Dirks Internship project at the University of Washington Libraries. The internship was supervised by Digital Preservation Librarian Moriah Neils Caruso; the project was supervised by Media Preservation Librarian Andrew Weaver.

Style Guide

Formatting

Actual and necessary workflow steps are numbered, with alphabetical substeps; additional commentary is further indented as a hyphenated list.

File paths are rendered in italics with forward slashes, like *C:/Users/emmma/iromlab*. Menu navigation is rendered in italics with a > character, like *Open with > IsoBuster*.

Spreadsheet column titles, dialog box warnings, and other text is put in quotation marks, as “Container description”. Punctuation follows quotation marks, even though it looks bad, so that the contents of the quotation marks corresponds exactly to the original.

Command line input and output is rendered in gray-shaded Courier New, like `pip install.` Where it seemed useful to highlight an element like a filename or extension, this may also be rendered computery (e.g. `.droid` file).

Spelling: Disk images of discs

See https://www.merriam-webster.com/words-at-play/disc-vs-disk-usage-history-spelling for some justification of the consistent-but-inconsistent spelling, which uses “discs” to refer to the physical objects and “disk” in the phrase “disk image.”

Hardware and Software Tools

Nimbie USB Plus Disc Autoloader NB21 Series

The Nimbie disc robot is named Johann.[1] It has a power adapter and connects to the PC by USB. It consists of a Pioneer BDXL disc drive mounted in a housing that controls loading discs automatically from a top-loaded stack and unloading them in one of two directions[2] based on a pass/fail outcome. The robot's native software, QQboxx Pro 3, is not used in this workflow. Its drive will be recognized as "BD-RE Drive" in Windows' *This PC > Devices and drives* and assigned a drive letter that will need to be passed to Iromlab in the configuration setup.

Iromlab

Available at https://github.com/KBNLresearch/iromlab, Iromlab is a Python tool developed by Johan van der Knijff at the Royal Library of the Netherlands (KB) that controls the Nimbie robot to batch-process several common optical media formats. The basic output is `.iso` disk images of data CD and DVDs and `.wav` files of audio CDs, along with various processing metadata. Iromlab wraps around several processing and validation tools that will be explained in more detail below.

Isolyzer

Also developed at KB, Isolyzer (https://github.com/KBNLresearch/isolyzer) is included in Iromlab to validate disk images but is also available directly at the command line. With the command `isolyzer /path/to/disk/image`, Isolyzer will provide more detailed information than what is logged in the Iromlab processing, including the size discrepancy and file system information for a "truncated" disk image.

BitCurator

The BitCurator Environment is a suite of digital forensics and data analysis tools packaged as a Linux distribution that runs as a virtual machine (VM) using Oracle VM VirtualBox. While BitCurator also includes disk imaging tools like Guymager and ddrescue, this workflow will use Iromlab (on Windows) to create the disk images and suggests BitCurator to explore additional reporting and quality control (QC). BitCurator can be set up using the Quick Start Guide available here: http://distro.ibiblio.org/bitcurator/docs/BitCurator-Quickstart-v2.2.pdf.

---

[1] Full name Johann Sebastian Batch. Namesakes: Johan van der Knijff (Iromlab developer) & J.S. Bach.
[2] This functionality has been written over with front-eject-only for this workflow; that edit will be described below.

Photo Station

The physical discs and their containers should be photographed and these photos associated with the processing metadata that is preserved along with disk images in preservation packages. This aspect of the workflow was not explored as part of this project, so specific recommendations for equipment and setup are not included in this document. It is recommended, however, that discs be photographed after they are imaged, as part of the handling that returns discs from the Nimbie bucket to their individual containers. This would prevent additional handling of materials while also avoiding potential confusion from having to photograph discs and populate the metadata spreadsheet at the same time. The University of Michigan "Optical Disk File Transfer Manual" includes a detailed description of how discs are photographed as part of their imaging workflow.

Setup and Configuration

Physical setup of processing workstation

The PC, Johann, and photo station should be set up at a desk or workstation that is clean and spacious enough for batches of 30 or more discs to be handled in and out of their individual containers without losing track of any or subjecting the materials to unnecessary handling or contamination. Space to move materials sequentially through the describing, imaging, photographing, and rehousing steps may not need to be linearly arranged but should be clear to the processor.

Configuration of project settings and metadata

This section assumes that Iromlab and BitCurator have been configured according to their setup instructions and focuses instead on the parameters that are more likely to be project-specific edits. For more details about initial setup–including documentation pitfalls and potential customization of tools–see the Appendix.

1. Edit the Iromlab configuration file, which is called `config.xml` and located in a folder called “iromlab” in your user folder. Some of these changes are specified in Iromlab’s Setup Guide, but some differ and may change by project, so they are listed here:
    a. Confirm Johann’s drive letter in the `<cdDriveLetter>` element. Be aware that Windows may reassign this periodically, prompting an error on launch (“‘K’ is not a valid optical drive!”), and it will need to be updated here.
    b. Update the `<rootDir>` element to a folder on a drive with sufficient space for the current project. Iromlab will create a folder in this location for each batch initiated.
        - For the initial project, `<rootDir>` was G:/spring_project_2, where the G drive was a 1 TB external hard drive labeled UW PRES REPL 004
    c. The `<prefixBatch>` element was defined as “2022-xx-xx-0x-wau” for this project, and instructions below will describe replacing xx element with the date and batch number. “wau” is just serving to separate visually the meaningful numbers from the UUID string that Iromlab will append to the prefix.
        - Since this nomenclature is not retained in the SIP, it may be unnecessary, but when multiple batches were being processed per day across different sets of discs, it was useful to have batch-level folders be human-readable and sorted in chronological rather than random UUID order
    d. The Setup Guide says to set `<startOnFinalize>` to “False” so that processing will begin as soon as one disc has been loaded and created as a job. For this workflow, however, it is recommended that `<startOnFinalize>` be changed to “True” so that Johann doesn’t eat your discs until the entire batch has been entered, loaded, and double-checked. This will slightly increase processing time but reduce the risk of handling errors.

e. Enter “wav” for the `<audioFormat>` element and follow the Setup Guide instructions for choosing “wave” in dBpoweramp setup as well.
    - Iromlab’s processing of audio discs into separate WAV files by track does not conform with Preservation’s current best practices for media preservation, but has been deemed acceptable in this workflow given the anticipated (low) proportion of audio CDs in Special Collections’ optical media materials and the context of the batch-processing workflow. In the event that a collection contains a high proportion of audio CDs, consult the Media Preservation Librarian.

f. Set `<enableSocketAPI>` to “False”. This functionality is specific to the KB use case, which queries their catalog to pull disc metadata.

2. Create or duplicate a spreadsheet for tracking disc processing and metadata. This project used Google Sheets in the UW Digital Preservation Shared Drive (projects/2022-03-opticalmedia/Imaging_Workflow_Processing_Metadata), whose column structure is reproduced here (italicized text following a colon is from a help-text row below the headers). Depending on adjacent workflows, another solution may become more appropriate, but this has allowed a relatively easy normalization of the Iromlab metadata and a way to guide the batch-renaming and -reorganizing processes along the way.
    - Collection-level metadata
        - Collection title
        - Accession & box number
        - SIP name
    - Batch-level metadata
        - Date initiated
        - Processed by
        - Iromlab batch ID: *Fill in processing date and batch number in UUID prefix before entering here. Otherwise note how the disc was processed (for example, PC drive + IsoBuster)*
    - Disc-level metadata
        - Assigned disc ID: *Assign sequential identifiers, unique within collection (even if processed in multiple batches)*
        - Identifying marks on disc: *Use to confirm accurate rehousing, not intended to serve as complete or definitive identification of discs.*
        - Container description
        - [Job folder rename]: *Copy formula down and create batch script to rename job folders: =concatenate("rename ", K4, " ", G4)*
    - Copied from Iromlab manifest.csv [paste values only to retain spreadsheet formatting]
        - jobID: *Paste Iromlab batch manifest.csv (excluding header row) into the cell below this one as "values only"*
        - PPN: *Hide this column; retained to match Iromlab manifest.csv*
        - volumeNo: *Hide this column; retained to match Iromlab manifest.csv*
        - title: *This column is conditionally formatted to validate against "Assigned disc ID"*

        - volumeID
        - success: *This column is conditionally formatted to highlight disc fails.*
        - containsAudio
        - containsData
        - cdExtra
        - mixedMode
        - cdInteractive
- Processing
    - Processing Notes: *For discs with success=FALSE, describe the error and actions taken.*
    - [Metadata restructure]

After the “Project Registry” tab, which offers top-level status tracking for each project’s tab, the spreadsheet has a “TEMPLATE” tab with headers, help text, and one row of sample data (to retain the batch concatenate formulas), which can be duplicated and renamed for each collection. This will be referred to throughout the workflow as the metadata spreadsheet or the project spreadsheet.

Disk Imaging Workflow

Initial metadata and disc transfer

The input here is assumed to be a box from a processed Special Collections accession that contains at least one optical media format. Boxes may contain a variety of legacy media formats. For example, Accession No. 20-007 (UW School of Social Work Records) Box 2 contained ten Zip disks, two commercial boxes with ten 3.5” floppy disks each, one cassette tape, and 27 CDs. All physically-CD-ish media (that is, thin 120mm-diameter plastic discs that may be any number of CD or DVD specifications) should be included in this workflow.

1. Preserving the order and arrangement of the materials, remove relevant discs from the box and assign them sequential identifiers beginning with “opt-001”.
    - Identifiers should be unique at the *collection* level, not the processing batch level, since metadata and packaging will be handled at the collection level, regardless of multiple Iromlab batches or individually reimaged discs
    - Label the discs’ containers with their identifiers in a reversible, unobtrusive way. Depending on the material, this may be pencil on a paper sleeve, pencil on a small sticky note attached to a plastic jewel case, or another method
2. Populate the metadata spreadsheet with these identifiers in the “Assigned disc ID” column. You should be able to drag down from the right corner of the first cell and auto-populate opt-002, opt-003, etc.
3. Transfer the discs to the loading stack of the Nimbie, populating “Identifying marks on disc” and “Container description” in the spreadsheet as you do. These do not need to be complete transcriptions or unique descriptions (many will be unlabeled discs and undifferentiated white paper sleeves), but should corroborate the physical processing and help prevent dissociation of container, disc, and disk image.
    - At least initially, It is recommended that batches consist of thirty or fewer discs. This is a number that will fit into the Nimbie without the extension pins, will not overflow the post-processing disc bucket, and will leave a manageable stack of empty containers
    - Move emptied disc containers to a second stack without flipping them over: since the first disc will end up at the bottom of Johann’s bucket, with the last disc on top, you want the stack of containers to be likewise reversed in order.
    - Some collections may contain loose discs with no container or multiple discs in a single sleeve. It is recommended that these be rehoused in individual paper sleeves, each numbered with its unique identifier, but this would require both curatorial approval and a supply of sleeves. If one-to-one correspondence of discs and containers is not possible, ensure that handling, numbering, and description nonetheless retain original order (for example, by inserting a piece of paper into your stack of sleeves as a reminder that a naked disc should not just be put into the next sleeve)

Iromlab processing

1. Turn on Johann. Wait until the calibration noises have stopped before double-clicking the “iromlab” desktop icon to launch. Confirm that the disc bucket is aligned in front of Johann to catch the processed discs.
2. Click “New” to create a new batch, and “OK” to accept the dialog box that appears.
3. Working from the list of assigned disc IDs in your metadata spreadsheet, generate a new job for each disc by entering (or copy-pasting) the disc ID into the “Title” field and clicking “Submit”. Do not adjust the “Volume number”.
    - The dialog boxes that appear assume KB’s use case of querying a library catalog and loading discs one by one, so they don’t make sense: just click “Yes” and then “OK”, then enter the next ID.
4. Once you have entered all the discs, double-check for typos or omissions in the Iromlab GUI, then click “Finalize” and then “Yes” in the dialog box that appears.
    - There is no way to edit the title or order of jobs, other than clicking “No” immediately after you entered a title. If something is incorrect, close Iromlab, navigate to the batch folder it has created, delete it, and relaunch to start over
5. When you click “Finalize”, Johann will eat a disc. Discs will be ejected into the bucket as they are processed, and Iromlab will print its working log to the lower portion of the GUI. A successful batch will end with a dialog box that reads “Finished processing this batch”. Clicking “Exit” or the “X” in the top right corner of the GUI will generate the dialog box “Quitting because user pressed Exit, click OK to exit”, then the program will close.
    - Since the dBpoweramp drivers have been reconfigured to eject failed discs from the front of the machine as well, the imageability of the disc should not impact successful processing: failed discs will be evaluated and reimaged in a later step.
    - During initial configuration and research, there were times when Iromlab would fail on a job and sit indefinitely, not ejecting the disc or completing the batch. These issues were resolved, but see the Appendix for more backstory if something similar starts to happen

Photographing and rehousing discs

1. The photographing station was not set up during this project. Instructions for how to set up equipment and photograph discs, naming conventions for photographs, and details of transferring files into the correct metadata folders can be interpolated here in the future.
2. After a disc is photographed, take its container from the top of the stack and rehouse it.
    - Again, containers should be placed into a second stack and not inverted. Since this happens at loading and unloading, you should end up with a top (opt-001) to bottom (opt-xxx) stack that is exactly what you removed from the box
3. Return discs to the box in their original arrangement.
    - If a box takes multiple batches, it may make sense to process all the batches before discs are returned to the box.

- If multiple boxes from the same collection are being processed at one time, it probably makes sense not to have a batch continue across multiple boxes, but the assigned identifiers *should* be continuous (i.e. unique) across all boxes.

Post-processing and quality control

1. Navigate to the batch-level folder that Iromlab has created in your top-level project folder. Rename it to populate the xx-xx prefix bits with the date, structured YYYY-MM-DD, using the last two xx as the batch number on that day. Copy this updated UUID into the “Iromlab batch ID” column of the metadata spreadsheet for each disc.
2. In the batch-level folder, open `manifest.csv` and copy everything except the header row. Paste this into the purple section of the metadata spreadsheet, just below the header rows, as “values only.” If your alignment is off or anything has gone wrong, and the “title” field from Iromlab does not match the “Assigned disc ID”, conditional formatting will flag the cell red.
3. Create and copy down a concatenate formula to rename each disc folder with its assigned identifier. It will be something like

   ```
   =concatenate("rename ", K4, " ", G4)
   ```

   where K4 is the long jobID assigned by Iromlab and G4 is the short assigned disc ID. Create a text file in the batch-level folder and paste all the concatenate strings in. Save the file as `rename.bat` (accepting the warning about changing file extension), then close and execute the batch script by double-clicking the file. You should see each job-level folder renamed.

4. Evaluate failed jobs. Conditional formatting in the “success” column will flag with red highlighting any discs that Iromlab reports as failures. The `manifest.csv` file just says success=“FALSE”, but slightly more information can be found in `batch.log` by searching for the disc ID and reading down to the “ERROR” line. There seem to be two distinct categories of failure, which call for different actions:
   a. **Irregular disk image.** Iromlab uses Isolyzer to validate the disk image. Sometimes, a job-level failure will be based on an error that is logged as “Isolyzer detected truncated ISO image” in `batch.log`. In these instances, the disk image and metadata seem to have been generated correctly, `isobuster-report.xml` will contain `<fileobject>` elements, and the disk image will mount in Windows. For more details, the disk image can be inspected with the command line version of Isolyzer, which will identify the size discrepancy in bytes and sectors and provide some file system analysis. Whenever this happened in testing, reimaging the disc with Iromlab, with IsoBuster, and in BitCurator (with Guymager and ddrescue) all yielded comparable, slightly-“truncated” disk images. That is, Iromlab is getting what it can from a disc that may be handling its file systems weirdly or have some physical damage. This outcome should be noted in the “Processing Notes” column of the spreadsheet, but it is not necessary to reimage the disc.

b. **No disk image.** The other kind of failure is when Iromlab does not create a disk image for a given disc. This also shows up as success="FALSE" in `manifest.csv`, but the job-level folder will not contain a disk image file and some or all of the other metadata files will be missing. The error message in `batch.log` is either:
    i. "no disc loaded" : Rarely, this may be a physical disc handling issue from Johann. If the handling mechanism picks up and tries to load two discs, for example, and nothing drops into the drive tray, then Iromlab continues to process but has nothing to look at. More often, the discs were loaded and unloaded normally and Iromlab just didn't see it for some reason.
    ii. "Unable to identify disc type" : This may indicate that the disc is a format that Iromlab or its underlying tool, cd-info, is unable to identify, but it also seems to be a catch-all for weird processing events.

    In both of these cases, the disc should be re-imaged separately; it is very likely that you will be able to create an accurate disk image with any of the following:

    1. Process multiple failed discs in a separate Iromlab batch. Use the originally assigned disc IDs (e.g. this second batch may include just opt-005, opt-006, and opt-020) and drop the `.iso` and metadata files into the (renamed) folders from the first batch.
        - In this case, the batch-level metadata (`batch.log` and `manifest.csv`) from the second batch should be included in the final preservation package's *metadata/processing_metadata* folder, renamed and mentioned in the `readme.txt` to disambiguate between those from the first batch
    2. Process a failed disc directly from IsoBuster, using either Johann (manually opening and closing the drive using its button) or your PC's internal drive.
        - Since Iromlab's configuration suppresses IsoBuster's looking for drives, you will need to launch IsoBuster from the recognized disc in File Explorer.
        - IsoBuster will name the file `Track 01.iso`. You may want to rename it to the disc name that IsoBuster identified, which would be more consistent with Iromlab's process.
    3. Image with the command line tool ddrescue in BitCurator.
        - A guide to using ddrescue is available here. A basic command is `ddrescue -d -b 2048 -r4 -v /dev/sr0 output.iso output.log`, where `-b` specifies block size, `-r` is re-attempts per bad sector, and `dev/sr0` is how Linux addresses the drive (you can see the available devices and their addresses by opening Guymager; Johann was `dev/sr1` at the time of writing)
    4. Image with Guymager in BitCurator.
        - Requires BitCurator's VM password ("bcadmin") to launch

- Various guides are available online, e.g. here.

These options were not researched extensively enough to rank by preference. In one case, Iromlab's `[reimage].iso` and IsoBuster's `[reimage].iso` were perfect MD5 matches, and ddrescue's `[reimage].iso` and Guymager's `[reimage].dd` were also MD5 matches, indicating some variation but a high level of reliability among these tools.

Once a failed disc has been reimaged, move the disk image into the original job folder that Iromlab made for it. IsoBuster should be used to generate a disc-level metadata report comparable to Iromlab's `isobuster-report.xml`. See the Appendix for more information about IsoBuster's reporting capabilities.

Packaging SIPs for preservation

The goal of this step is to specify a structure for submission information packages (SIPs) for digital preservation that includes a responsible but reasonable amount of metadata from the disk imaging process and takes advantage of the programmatic reporting of the tools involved, without relying too heavily on the metadata structure of any of those tools. The following factors were considered in the recommendations made below:

- When discs fail in batch processing and need to be imaged separately, we should be able to generate parallel processing metadata without using exactly the same tools (e.g. not having to run a one-disc Nimbie batch)
- Collections may contain just one CD, in which case direct imaging with another tool may be simpler
- Much of Iromlab's processing metadata is gratuitous (e.g. SHA checksums, when bagging and Archivematica handle MD5) or minimally useful (`cd-info.log`) and would only clutter the packages and distract from useful higher-level processing information
- The structure of the metadata should be consistent with department practices for other media; submissionDocumentation folders from other projects (e.g. 2022 streaming media SIPs) were considered
- Maintaining the job-level folders lets us retain the disk image names assigned by Iromlab (which are based on volumeID), and reproducing that folder structure in the metadata provides some logical symmetry, allows us to leave `isobuster-report.xml` files named as-is (i.e. identically), and provides some architecture within which the item-level metadata specifications could evolve

The recommended structure of the SIPs is as follows:

wau-shorttitle-[oclc#]

|_ data *(folder created by bagging)*
- |_ opt-001
  - |_ `volumeID.iso`
- |_ opt-002
  - |_ `volumeID.iso`
- |_ opt-003
  - |_ `volumeID.iso`
- |_ opt-004
  - |_ `01.wav`
  - |_ `02.wav`
  - |_ `03.wav`
- |_ *etc…*

|_ metadata *(moved in after bagging)*
- |_ processing_metadata
  - |_ `batch.log`
  - |_ `manifest.cvs`
  - |_ opt-001-metadata
    - |_ `isobuster-report.xml`
    - |_ disc_photos
      - |_ `opt-001-disc.jpg`
      - |_ `opt-001-container.jpg`
  - |_ opt-002-metadata
    - |_ `isobuster-report.xml`
    - |_ disc_photos
      - |_ `opt-002-disc.jpg`
      - |_ `opt-002-container.jpg`
  - |_ opt-003-metadata
    - |_ `isobuster-report.xml`
    - |_ disc_photos
      - |_ `opt-003-disc.jpg`
      - |_ `opt-003-container.jpg`
  - |_ opt-004-metadata
    - |_ `dbpoweramp.log`
    - |_ disc_photos
      - |_ `opt-004-disc.jpg`
      - |_ `opt-004-container.jpg`
  - |_ *etc…*
- |_ `readme.txt`
- |_ `ACC.pdf`
- |_ `finding_aid.xml`
- |_ `metadata.csv`

If a single collection has been processed in multiple batches, the job-level folders should be collocated (and the batch-level metadata disambiguated, e.g. `batch-01.log`, `manifest-01.csv`) at this point, so that further description and organization operates at the collection level.

The reorganization, renaming, and metadata weeding required to transform Iromlab's batch-level folder(s) into a collection SIP could be accomplished in a variety of ways, more or less programmatically, depending on the operator. One possible process, which relies primarily on concatenating batch scripts from the metadata spreadsheet, is as follows:

1. Copy disc-level processing metadata out into a folder structure for the collection SIP. The following concatenate string is embedded in the working spreadsheet template and

should find the `isobuster-report.xml` or `dbpoweramp.log` file in each disc-level folder (depending on whether it was a disk image or WAVs) and copy them into structured metadata folders, adding an empty “disc_photos” folder to each disc’s metadata folder for the photos taken as the discs were rehoused.

```
=concatenate("xcopy ", F4, "\", G4, "\isobuster-report.xml ", $C$4, "_metadata\processing_metadata\", G4,"-metadata\", CHAR(10), "xcopy ", F4, "\", G4, "\dbpoweramp.log ", $C$4, "_metadata\processing_metadata\", G4, "-metadata\", CHAR(10), "mkdir ", $C$4, "_metadata\processing_metadata\", G4, "-metadata\disc_photos")
```

Confirm that the concatenation looks right, then proliferate the formula down to all disc rows, copy the cells, and paste them into a `.txt` file in the top-level project folder. If that generates quotation marks around each cell’s three lines, find-and-replace with a blank to remove them all, then rename the file `move.bat`, save and close, and double-click to run it.

- `xcopy` was preferred to `move` because it creates the specified folder structure as it goes; it does leave the metadata behind in what will become the “data” folders, but since those will require further editing/QC in the next step, it didn’t seem like the worst; xcopied files could also be batch-deleted from the old location with a line added to the concatenation

2. Generate standard Special Collections metadata as described in the AM Workflow:
    a. metadata.csv
    b. finding-aid.xml
    c. ACC.pdf
    d. readme.txt

Include in the readme a brief transfer note, including the basic batch process and any specific disks that were reimaged in a different manner. For example:

```
This Information Package was constructed in the Preservation Department of the University of Washington Libraries in May 2022.

The "objects" folder contains objects associated with Accession [Accession Number], [Title].

This folder ("metadata") includes a number of metadata files that have not been incorporated into the Information Package's master METS file.

The METS file draws from the metadata.csv file found in this folder, which reflects descriptive metadata for the collection derived from the bibliographic MARC record.

The other metadata files contain descriptive information as follows:
```

```
ACC.pdf
Contains accession-level administrative, technical, and descriptive
metadata pulled from the Libraries Special Collections instance of the
ArchivesSpace information management system, including data that may
not be represented in the finding aid.

finding_aid.xml
This is the EAD encoded finding aid for the collection.

processing_metadata folder
This folder contains metadata generated during the disk imaging
process.

Transfer notes:
Disk images were generated by a batch process that uses a Nimbie USB
Plus Disc Autoloader NB21 Series disc robot running Iromlab software
(https://github.com/KBNLresearch/iromlab), which controls IsoBustser
and other tools to generate and validate ISO disk images.

Discs opt-05, opt-06, opt-10, and opt-20 yielded errors during batch
processing. When investigated individually with IsoBuster, opt-05 and
opt-06 were successfully imaged; opt-10 yielded an identical (MD5
match) disk image with the same "truncated ISO image" warning, so the
original image was retained; opt-20 was identified as a blank disc.
```

3. Clean up the “data” folders. Delete all `cd-info.log`, `checksums.sha512`, and `isobuster.log` files, either from File Explorer (search top-level folder, highlight all to delete) or from the command line. Visual inspection of each folder is recommended, to confirm that each ends up with only an `.iso` and/or `.wav` files and other metadata has been moved or removed successfully.

4. Rename the batch-level folder according to SIP naming convention, then bag it, move the metadata in, and process according to the digital preservation workflow.

Reporting to Special Collections

This step was not resolved to the point of a recommendation, although one promising option is the configurability of the `isobuster-report.xml` file via the text string in the `<reportFormatString>` element of Iromlab’s `config.xml` file. That option and others are described in the Appendix.

Appendix: Details, Backstories, References

This section contains more detailed information about the tools and processes deployed in the workflow above. It should not be necessary for the person executing the workflow but may prove useful for troubleshooting, initial setup of a new workstation, or when considering changes to the workflow.

Iromlab

The Iromlab documentation on GitHub is robust, but the tool itself can be opaque in its logging and stubborn in its failures. It is built to the KB use case, which diverges from UW's in several ways. A few important notes that go beyond the Iromlab documentation:

- When it is initially installed and configured, Iromlab uses the file `config.xml`, which is part of the installation package (located somewhere like *C:/Users/emmma/AppData/Local/Programs/Python/Python36/Lib/site-packages/iromlab/conf/*) to create a `config.xml` file in an `iromlab` folder in your user folder; configuration edits must be made to *iromlab/config.xml* and **not** to the template in AppData
- Especially if BitCurator is run simultaneously with Johann, Johann's assigned drive letter may change. If this happens, Iromlab will launch with the error, "'K' is not a valid optical drive!" where "K" is the previously assigned drive letter. Open the `config.xml` file and update the `<cdDriveLetter>` element to match what is assigned in *This PC > Devices and drives.*
- The default functionality of Iromlab is to eject successful rips from the front slot of the Nimbie and failed rips from the back slot. The back slot doesn't have a receiving bin to contain the ejected discs, however, and splitting the output also disrupts original order, so Iromlab was reconfigured to eject all discs into the front bin by simply editing `config.xml` so that the `<rejectExe>` element pointed to the same driver as the `<unloadExe>` element, rather than to the actual `Reject.exe` file; this could be reversed by restoring the `<rejectExe>` element to the original file path, which is something like *C:/Program Files/dBpoweramp/BatchRipper/Loaders/Nimbie/Reject/Reject.exe.*

Debugging issues with Iromlab:

1. **CD-DA configuration issue.** After the initial configuration of the Nimbie and Iromlab, we found that it wasn't able to process audio discs. The GUI would print "***Ripping audio***" to the log and then sit there indefinitely. The source of the issue was obscured by switching between the NYPL and KB forks of Iromlab around the same time (see below), but after a lot of fruitless code-reading, it turned out to be the naming convention setup from the dBpoweramp configuration. Iromlab apparently relies on CD Ripper passing files called 01.wav, 02.wav, etc., to notice that the ripping is complete and

continue with batch processing, so following that step of Iromlab's dBpoweramp setup guide is crucial, even though we skip the adjacent AccurateRip stuff.

2. **Batch misalignment.** This is mentioned in the Iromlab documentation as "synchronisation errors," and was an issue with Andrew's initial Iromlab test batches: for unknown reasons, Iromlab and/or the Nimbie and/or IsoBuster can get out of alignment and map the user-entered Iromlab metadata onto the wrong discs, potentially running out of "jobs" before it runs out of discs. I was not able to replicate this issue at any point during the project, so I don't have any specific observations or recommendations, but if the point of error is not easily identifiable, the entire batch should probably be reimaged.
3. **Multitasking IsoBuster lies to Iromlab.** At one point, I forgot that I had IsoBuster open to investigate a disk image and started an Iromlab batch of three discs. Iromlab logged "Error: Unable to locate Nimbie robot, cancelling batch rip" but then ran through the queued jobs and each time it tried to call IsoBuster, IsoBuster gave it the disk image that was already open instead of loading and investigating a physical disc. This should be easy to detect because the physical discs were left in the loading stack, but Iromlab generated job-level folders and copied the same `.iso` file (the one that had been open in IsoBuster) into each one, so it initially looked like the batch had been successful.

The New York Public Library (NYPL) forked Iromlab in 2019 and made significant adaptations, including restructured logging and an integration of bagit.py to process logical file copies. This fork was investigated as a possible implementation for UW, but it presented its own issues and ultimately it seemed more solid to stay with the main product. Some observations:

- The NYPL fork can be downloaded with `pip install git+https://github.com/NYPL/iromlab` (after uninstalling a pre-existing version of Iromlab) or by downloading the code folder from the repository and swapping out the scripts where your previous version had them
- NYPL did not update the documentation for Iromlab; critically, the GUI will not launch until you open the `config.xml` file in your user folder and add an additional element called `<staff>`. The comma-separated string in this element will be used in the GUI to populate a drop-down list of who is running the process.
- The NYPL GUI allows you to set the batch prefix by text entry and switch between bagging and disk imaging
    - It was enforcing some some format rules on the UUID, e.g. prompting "Collection ID must be format M#####" if you entered something with letters in it, but accepting strings with any number of numbers
    - The default drop-down option is bagging and it's easy to forget to switch to disk imaging and then have to cancel out entirely
- NYPL's process is described in a 2021 code4lib paper, "Assessing High-volume Transfers from Optical Media at NYPL," which points to an additional Python tool, optical-media-check, that gathers their modified Iromlab logging into a CSV output; this seems like valuable functionality but, given its dependency on the NYPL-specific workflow and fork, not worth adopting wholesale

The Iromlab dependencies are as follows:

IsoBuster

Install and authenticate a Pro license of IsoBuster, and follow the Iromlab IsoBuster setup and configuration guide. Note that the `/nodrives` configuration described there will mean that you can't launch IsoBuster normally to do separate disk imaging on the workstation PC. You can either reverse the executable edit (removing `/nodrives`) to use IsoBuster directly or just launch it externally: right-click on a recognized disk (in *This PC* > *Devices and drives*) and select "Investigate with IsoBuster" or right-click on a disk image and select *Open With* > *IsoBuster.*

Iromlab's setup configuration attempts to suppress dialog boxes from IsoBuster that would interrupt processing, but these still occur occasionally. For example, attempting to image an SVCD disc, IsoBuster gave "Extracting aborted. Do you want to delete the file?" Most boxes will have an option "Do not bother me with this again", and all prompting behavior can be controlled from IsoBuster's *Options* > *GUI* menu.

dBpoweramp

Since the KB use case involves imaging commercial discs, the parts of the Iromlab dBpoweramp setup and configuration guide around AccurateRip does not apply to this workflow. Note the following:

- Under "Configure general settings," choose "Rip to *Wave"* (this should match your `config.xml` setting)
- Skip the "*Add DSP / Action"* but do **not** skip the "*Naming"* step! If dBpoweramp is not configured to set track numbers to *"TrackNum"* rather than its default, Iromlab will stall out on any audio CD
- Under "Configure ripper options," deselect "Query Ripping Results with AccurateRip," then skip the whole "AccurateRip configuration" section: we do not want to pull internet metadata or album art for any rips

Isolyzer

Iromlab uses Isolyzer to verify the disk images that it creates. Until version 1.4, which was pre-released during the development of this workflow, Isolyzer had issues with Apple file systems like HFS because of how it was interpreting the Apple partition maps, so certain seemingly successful rips would be marked success="FALSE" in the Iromlab manifest with the error "Isolyzer detected truncated ISO image" in `isobuster.log`. Since many discs contain multiple file systems, and IsoBuster is pretty clever at using all of them, it was hard to identify reliable patterns of disc type and imaging failure, and the truncated-image error still occurs sometimes (see note in workflow above), but the Isolyzer update seems to have helped.

More thoughts and options for reporting

My understanding is that the most useful format of reporting to offer back to Special Collections is a basic pass/fail assessment of the imaging process and a directory print of each disc. When it works, DROID seems to generate these in the GUI, but overall it seems like an unreliable

substitute for the ideal tool that seems not to exist–something that would take a directory containing multiple disk images for input and return a file tree:

- **DROID** can do this in the GUI, but does it unreliably, and does not export the data in a logically sortable way
- **Brunnhilde** can take a directory as input but will only list the disk image as a file; the `-d` flag allows input to be a disk image, in which case its contents *will* be explored, but `-d` and directory input are mutually exclusive
- **fiwalk** (also used in Brunnhilde) will only look directly at a disk image
- **Demystify** (on GitHub here) is an analysis and reporting engine for DROID and Siegfried outputs; I came across this via its brand-new browser version very late in the project but there might be some promise here!

DROID

Available at https://github.com/rhubner/droid, DROID (Digital Record Object Identification) is a software tool developed by the UK National Archives for batch identification of file formats using the PRONOM technical registry. The main advantage of this tool for exploring disk images is that, in a simple graphical user interface (GUI), it can be pointed at a directory with multiple disk images and used to explore the file structure of the content of those disk images. Multiple directories can be added to a single "DROID profile," which can be saved as a `.droid` file format and relaunched, making this a possibly valuable tool for providing a explorable-file-tree style of reporting to Special Collections curators.

Unfortunately, DROID stumbles over different issues than Iromlab, and sometimes freezes mid-processing with no indication of what has caused an error, so its use in QC at this point may be more confusing than helpful.

Launch DROID from the `droid.bat` file in the install folder. It will initiate a profile called "Untitled-1". Click "Add" and select a directory containing one or more disk images, then click "Start". A blue progress bar at the bottom of the window will list file names as they are assessed, but the file name listed when it freezes does not appear to be the actual item causing an issue; there is no indication of when DROID has reached a critical failure and will not continue processing, and sometimes the program has to be ended from Task Manager. The plus icon to the left of your top-level folder allows you to expand the file tree and explore the contents of disk images.

Processing can be paused and resumed, and "Save" will create a DROID profile (file extension `.droid`) that can be relaunched to resume processing/investigation. Note that a saved DROID profile *will not work if it is renamed.* "Export" can be used to generate a full directory print CSV of file format identifications, but the way that disk-image-internal files are sorted away from the higher-level tree makes this a somewhat difficult document to read and probably not a useful export for curators. Sending the DROID profile itself, which would allow anyone with the program installed to navigate the expandable file tree in the GUI, would seem like a better option, except the irregularity of DROID's results lowers my confidence in recommending it.

For example, it will sometimes freeze on files in a disk image that IsoBuster reported no issues with, which can be opened without a problem from the mounted disk image in Windows. Run against a batch-level folder with 27 job folders, it froze on opt-015 the first time and opt-027 the second time, reporting empty folders (indicated with a red folder logo with a zero on it) in opt-015 where the disk image was mountable and had files in those folders.

The HTML report offers comprehensive statistics but little logical overview of the images (that is, it's difficult to actually understand what the content is, in the way that a file tree or directory print offers). A solution to this would be to drop the DROID export into Demystify Lite, but that is brand new (late May 2022) and a little buggy, and a two-layer solution seems less than ideal.

Brunnhilde

Available at https://github.com/tw4l/brunnhilde, Brunnhilde uses Siegfried and fiwalk to generate various comprehensive reports from disk images or directories that contain disk images. It can be installed in Windows, but direct disk image analysis (`-d` flag) is only available in Linux and macOS, so it's only really useful here in its BitCurator iteration. Since the workflow has ended up operating entirely in a Windows environment, it may seem inefficient to require BitCurator just for the reporting stage.

Within BitCurator, navigate to *Forensics and Reporting* > *Brunnhilde* (with the command line icon) to launch a terminal with the usage notes. A basic command is

```
brunnhilde.py -d -n -w path/to/ISO path/to/directory/for/output/
```

where `-d` sets the input to disk image, rather than directory, `-n` suppresses ClamAV virus scan (which is slow), and `-w` includes Siegfried warnings in the HTML report (e.g. file format "match on extension only").

Unfortunately, the `-z` flag ("Decompress and scan zip, tar, gzip, warc, arc with Siegfried") does not include scanning within `.iso` files, so the only way to see inside a disk image is to use `-d` and do one at a time. UMich uses Brunnhilde for reporting to Special Collections (see below).

Reconfiguring `isobuster-report.xml`

IsoBuster offers the functionality of a completely configurable reporting output, either via the command line or with predefined reports that are stored in the Windows registry. Iromlab uses this functionality to define a custom report with the `<reportFormatString>` element in `config.xml`. That is generated and saved as `isobuster-report.xml` for each job. The `<reportFormatString>` element could be edited so that `isobuster-report.xml` reads much more simply like a directory print of the disk image: these files could then be batch-copied, renamed, and transferred to Special Collections as the report, or concatenated into a single report file, or so on. The IsoBuster documentation for creating custom reports is here and here.

The trade-off would be losing the more robust `isobuster-report.xml` that is currently included in the SIPs as processing metadata, although any of its contents could be regenerated

by investigating the `.iso` file directly with IsoBuster at any point. It may in fact be the case that `isobuster-report.xml` is just reassuringly DFXMLy and robust-*looking* right now without actually providing necessary details, and the simpler report would be more useful in the SIP as well.

This functionality could also be used to add Iromlab's `isobuster-report.xml` template (or whichever customized version of it is being used) to the IsoBuster GUI, so that an identical report is generated for batch-processed discs and for discs that are individually reimaged. In the initial project, IsoBuster's standard DFXML template was used, which is functionally equivalent but not identical.

References

The following institutions have published relevant workflows around optical media preservation:

- University of Michigan Digital Preservation Unit. (2018). "Optical Disk File Transfer Manual"
    - Discs are imaged individually with FTK Imager; Brunnhilde for reporting; extensive notes on photographing physical items
- New York Public Library. (2021). "Assessing High-volume Transfers from Optical Media at NYPL"
    - Using the NYPL fork of Iromlab (see above) for logical copying and bagging
- The British Library. (2011). "Developing a Robust Migration Workflow for Preserving and Curating Hand-held Media"
    - Uses Nimbie, with emphasis on high throughput; this paper is referenced as the foundation of KB's development of Iromlab
- New York University. (2018). "An Optical Media Preservation Strategy for New York University's Fales Library & Special Collections"
    - Comprehensive overview of formats and standards, good source for background and other sources
    - Recommendations are inflected by NYU's established Windows/FTK usage
- Yale University. (2016). "To Image or Copy -The Compact Disc Digital Audio Dilemma"
    - Focus on CD-DA only, not strictly the same workflow, but they use IsoBuster to identify disc format and then Exact Audio Copy to extract CUE/WAV

Other resources consulted in the course of this project include:

- Duryee, A. (2014). "An Introduction to Optical Media Preservation"
- ISO/IEC 10149, "Information technology - Data interchange on read-only 120 mm optical data disks (CD-ROM)" aka the "Yellow Book"
- Kirschenbaum, M., et al. (2010). "Digital Forensics and Born-Digital Content in Cultural Heritage Collections" (2010)
- Byers, F. R. (2003). "Care and Handling of CDs and DVDs"
- George Blood Audio Video Film. (2014). "Preserving Write-Once DVDs: Producing Disc Images, Extracting Content, and Addressing Flaws and Errors"

- Library of Congress. (2009/Ongoing). "CD-R and DVD-R RW Longevity Research" and "Compact Disc Service Life: An Investigation of the Estimated Service Life of Prerecorded Compact Discs (CD-ROM)" and "Longevity of CD Media: Research at the Library of Congress"
- Consultative Committee for Space Data Systems. (2012). "Reference Model for an Open Archival Information System (OAIS)"
- Clarke, A. (1986). The British Library's compact disc experiment.
    - This is a print report available in the library (TK 7882.C56 C55 1986) that includes some excellent original Philips brochures as appendices