

Notes on OUS-palooza

This page will hold our notes from implementing first fringes on an OUS level ALMA restore, as tracked in JIRA:

 [SSA-4811](#) - Jira project doesn't exist or you don't have permission to view it.

In addition, for familiarization purposes, we should upgrade our current ACS installation:

 [SSA-4916](#) - Jira project doesn't exist or you don't have permission to view it.

Test Data Set Details:

Project 2017.1.00886.L

MOUS ID: uid://A001/X1284/X265f

made up of four ASDM EBs: uid://A002/Xc7111c/X3979,
uid://A002/Xc7111c/X409e, uid://A002/Xc7111c/X86a7, uid://A002/Xc72427/X2dfe

All these were processed recently, so have the calibration products stored separately (note though the data are still proprietary, so should not be shown too widely).

Adding the almapipe group to our accounts might make life easier (particularly for look-but-don't-touch access into the working areas).

Other Test Sets:

- [uid://A001/X12a3/X80e](#)
- [uid://A001/X12cc/X4a](#)
- [uid://A001/X1284/X266](#)
- [uid://A001/X1284/Xf3b](#) – Failed, missed an empty Session initially.
- [uid://A001/X133d/X3681](#) - Failed due to a NULL stoptime value.

PPR Examples:

Kana was kind enough to provide a few example PPRs that perform multi-eb processing:

- A combined calibration/imaging run for an OUS for 2 EBs from a single Session: [calimage_2eb_1session_PPR_uid___A001_X12cc_X4b.xml](#)
- A calibration-only run for an OUS containing 2 EBs which spanned 2 Sessions: [calonly_2eb_2session_PPR_uid___A001_X1284_X267.xml](#)
- An imaging-only run for an OUS containing 2 EBs which spanned 2 Sessions: [imageonly_2eb_2session_PPR_uid___A001_X1284_X267.xml](#)

I'm just starting in on understanding these, and comparing them with the PPRs for CIPL and Vlass quicklook imaging.

XML Schema Home:

ALMA xml schemas live in the the Alma Common Software area, and each successive component adds to that compilation. It's in `${ACS_ROOT}/${ACS_VERSION}/ACSSW/idl/` (A concrete example in CV: `/home/acs/ACS-2018JUN/ACSSW/idl/`) The most likely schema of interest are:

- OUSStatus.xsd
- SciPipeRequest.xsd (i.e. the PPR)
- SciPipeResults.xsd (i.e. the Pipeline Manifest)

Software Details:

For our test data set, we want to be using CASA/pipeline version 5.1.1-5r40896. This is the ALMA-approved version of CASA for Cycle 5. See: <https://almascience.nrao.edu/processing/science-pipeline>

The new functionality for the ALMA Common Software (ACS) is built into the `asdmExportLight` command through a new option. This hopefully will make it fairly easy (once we understand the Project & Database structures) to retrieve the desired calibration information. One wrinkle, however, is that ACS is an overloaded acronym. There is the 'base package' of ACS, and extensions to it. Further investigation has revealed that we will need the Archive extension at minimum, and likely will want the Pipeline & ICD components as well.

The `pipelineMakeRequest` tool will be useful for the creation of comparison scripts for our more extended PPR generation software. It is ALMA focused, and builds the entire directory structure (including data download, if requested), complete with OUS Status files for each level.

The NAASC keeps a set of ACS installations for their purposes in `/home/acs`. Unfortunately, this area doesn't include the ACS-2018JUN release of the software with a fixed version of calibration retrieval. I'll perform a test installation of an updated Enhanced-ACS in `/home/alderaan_2/` for initial verification & prototyping.

SSA may also need to update the version of ACSSW in `/home/ssa/ACS`, as we are currently using a version tagged for Cycle 3. After the initial verification, the `/home/ssa/ACS` area can be updated with an updated version to support the restore functionality.

There's a tool which will provide information about projects & what's happening with what, so you can confirm what you suspect. ProjectTracker, but you need 'staff' privileges and that makes things dangerous (you CAN alter the state or start a processing run, for instance). Currently [James Sheckard](#), [Rick Lively](#) [X], and [Stephan Witz](#) have privileges if you need to dig into what's going on with an alma project.

ALMA Archive Questions:

We need to generally understand the layout of their archive, and how things interlink. But if we come up with specific questions as well, record them.

- Each OUS has a status associated with it.
 - This information, while possibly nice to have, is not critical the processing. Even for initial calibration or imaging, those files are not accessed.
 - Still lacking a source for all but the SB Status file ID.
 - Update Timescales.
 - Update of our release date with the ALMA MDDb (see below)
 - proprietary status claw-back can happen.
 - If no imaging products are present for a MOUS, the calibration products might indicate a work in progress.
 - Full Structure: Plan vs Observation vs Reduction
 - Potential:
 - Proposal (i.e. Phase 1):
 - Science Goal(s)
 - Each Science Goal is tied to a single ALMA receiver.
 - Each Science Goal has an associated SGOUS once the proposal becomes a project.
 - Observation:
 - Project (i.e. Phase 2):
 - SGOUS
 - GOUS
 - MOUS
 - Each MOUS contains data from a single array configuration.
 - Typically 1:1 with Scheduling Blocks, unless there's a problem with the SB.
 - Scheduling Block(s)
 - Each Scheduling Block contains:
 - Once/session calibration scans (bandpass, flux calibrator, etc)
 - Structure for switching between science target(s) and more variable calibrators (phase, polarization, etc)
 - Session(s)
 - Due to system & atmospheric stability issues, each SB is limited to about 2 hours. If conditions permit, the SB can be run multiple times in succession. Those contiguous runs are called sessions.
 - They seem to be important for CASA, and so should be tracked.
 - Execution Blocks
 - SDM+BDFs for a single execution of an SB. The basic data units (stored as tar files in ngas)
 - Failed observations handled how?
 - Reduction:
 - Project
 - Science Goal OUS(s)
 - Group OUS (1:1 with science goals for the foreseeable future, no pipeline processing at this level for the foreseeable future).
 - Member OUS(s)
 - Calibration & Imaging performed at this level via the pipeline.
 - Proprietary Period begins upon the release of Imaging products to the PI.
 - Session(s)
 - Execution Blocks(s) or ASDM(s)
- How will we deal with extracting calibration tables?
 - Are we gonna do that, or is handling it an ALMA infrastructure thing?
 - There currently is no streamlined method for extracting the calibration information within the ALMA infrastructure software.

- **ToDo:** We'll need a homegrown methodology of extracting the calibration tables directly from ngas with wget (see: [listfiles.py](#))
 - Are they gonna go back and split the calibrations out of the older observations?
 - The older calibrations will be split out at some later time, allowing restores of the older data.
 - **ToDo:** How will we know when this is accomplished for a particular MOUS?
- Details which must be tracked:
 - How do we distinguish hand-calibrated data from pipeline calibrated data?
 - How do we distinguish between standard and MPI CASA having been used?
 - The pipeline provides a restore script which handles the details of the arguments (including optional ones) in the restore_data call.
 - Can we use that via another pipeline task?
 - How can we reasonably handle both existing and future flags which might have to be handled?
 - gainmap=True for VLASS
 - There was one script I saw with an additional step in its restore script, how do we handle those cases?

ALMA Metadata Database Linkages:

WARNING: The identifiers are not constantly used. The same MOUS ID (as given above) is referenced as a MOUS_ID, STATUS_ENTITY_ID, and ASA_OUS_ID in various tables. I'll try to be more explicit on which columns link to each other below: .

- ASA_SCIENCE
 - For each EB , this will have a row providing the SOUS/GOUS/MOUS UIDs.
 - Under the OBS_UNITSET_ID is the id of the Status File for the this EB (this should be extracted into the created directory structure).
 - There's also a field for the SB UID, which we need for the PPR.
- ASA_PRODUCT_FILES
 - For each MOUS, this table lists the files produced via the pipeline.
 - Of particular interest are the _caltables.tgz and _auxcaltables.tgz files.
 - If these files exist for an MOUS, then we can perform a restore.
 - Is it possible to have multiple calibrations for a given MOUS?
 - The complicated *.scriptForPI.py is the method for performing a restore.
 - It goes looking for every possible method (PPRs, restore scripts, calibration scripts, etc, in order to handle both pipeline and hand-calibrated cases).
- ASA_DELIVERY_STATUS
 - RELEASE_DATE is the Official word on whether a piece of data is available to the general public.
 - It is possible (but not common) for a claw-back to happen if a proposal is resubmitted due to requiring more data.
 - Proprietary countdown starts upon the release of Imaging products to the PI, but may be extended if an error in processing is detected.
- SCHED_BLOCK_STATUS
 - Contains the Status ID when given a scheduling block. Has a slot in the PPR.
- OBS_UNIT_SET_STATUS
 - Links the ID we're likely to be given to the XML (STATUS_ENTITY_ID = MOUS ID we were given gets us the XML file with session information)
 - The MOUS status XML contains session information for building the PPR.
- AQUA_OUS
 - Use the MOUS ID in the OUS_STATUS_ENTITY_ID column.
 - Provides the link between levels of the Status Entities (walk up the project)
 - Also directly ties an MOUS To the ObsProject + PartID needed for the ProjectStructure section of the PPR.

Tasks:

- develop (M)OUS metadata structure for the database
- develop an update system to learn about calibration status of ALMA data
- Python prototype
 - develop queries for detection & extraction of calibration products
 - develop queries for detection & extraction of status XML files
 - How do we go from the ASA_SCIENCE table information (ASDM UID and/or OUS UIDs to the status xml files, or their constituent information)
 - How do we extract the status information?
 - Are the files stored in NGAS, a database table as XML, or built from table data?
 - What else can we learn using them?
 - develop queries to provide relevant ProjectStructure data to the PPR
 - Session mapping in particular
 - PPR Generation (Successful(ish) hand generated PPR: [PPR_bespoke_worked.xml](#))
 - test via hand-running on the cvpost cluster
 - resolve restoration methodology questions
 - creation of directory structure
 - resolve ALMA vs VLA structural differences
 - comparison with the results of pipelineMakeRequest (restore version)

Relevant Links:

1. Restore Methodologies:
 - a. [CIPL Restore Instructions](#)
 - b. ALMA Restores follow this pattern:
 - i. Download products via the RH
 - ii. Download raw data via the RH or asdmExportLight
 - iii. Arrange directories appropriately
 1. raw: subdirectories for each ASDM
 2. calibration: *caltables.tgz files
 3. script: python control scripts, handling the details. Includes the overall scriptForPI.py.
 4. Creates: calibrated directory with a CMS.
 - iv. Open the appropriate version of CASA (see link under Software Details above)
 - v. `execfile('scriptForPI.py')`
 - vi. Caveats:
 1. For pipeline calibrated data, the CASA version is more flexible (later ones can be used)
 2. For hand-calibrated data (10-20%) the CASA version match is a must.
 - a. hand calibrated data don't 'restore' so much as re-run the entire calibration (thus it's more sensitive to CASA changes).
 3. There's also a question of parallel vs single-processor CASA measurement set differences.
 2. [Comparing Measurement Sets \(Rough Draft\)](#)
 3. [Base ACS homepage](#).
 - a. Suggested Version: 2018Jun
 - b. Download Location: [here \(requires login\)](#)
 4. Other ACS component homes?
 - a. ARCHIVE
 - b. PIPELINE
 - c. ICD
 - d. Others unlikely to be relevant