SAM PDS Procedure

This procedure outlines the process for creating a SAM Reduced Data Record archive. Make sure to follow all the steps outlined here. Edit this page if anything changes.

Important People

- **Heather Franz:** Heather wrote the SAM RDR SIS (Software Interface Specification), a user's guide for the RDR archive. Heather generally attends the MSL DAWG meetings and will keep you informed if anything important happens there (you usually don't need to attend). Heather also generates the high-level QMS products.
- **Jean-Yves Bonnet:** Jean-Yves generates the high-level GC products.
- **Greg Flesch:** Greg generates all TLS products.
- **Susan Slavney:** Susie is the "Geosciences" PDS Node. She is our main point-of-contact between the SAM and PDS teams. If you have questions about PDS validation, etc., contact her.
- **Joy Crisp:** Joy is the chair of the MSL DAWG, but usually does not get directly involved in SAM's RDRs. I believe she announces the delivery schedule.

Before the Delivery

The MSL DAWG discusses a release schedule, usually shortly after the last release. You will usually have at least two months to get the products together, but since assembling the archive requires inputs from other parties, it's important to contact the other parties as soon as possible.

When you get the email from Susie labeled "MSL PDS Release 13 schedule", there will be an attachment that indicates when certain things are due. The first thing asked is "PDS asks the data providers for their archive readiness reports" which will be an email that will come out on the date (about one month before delivery is due). You want to plan to email Heather, Greg, and Jean-Yves a couple days before this date basically saying the following:

> Hi all,
>
> Release XX covers sols yy-
>
> zz, (START_DATE – END_DATE). Heather can you please release a list of the TIDs that we need for this delivery.
>
> Delivery is due DELIVERY_DATE so I will need all materials by 2_WEEKS_BEFORE_DELIVERY_DATE.

All the information will be in the attachments from Susie's email.

When you get the email asking for readiness report, just indicate delivery will be made on time and there are no changes unless Heather/Jean-Yves indicate something will be new or if you do not believe you will have delivery made in time.

Now just wait until you get all the materials from the team members and if someone has not delivered them on the date make sure you message them (if they forget they can usually get it to you in less than 24 hours).

Preparing the Inputs

SVN

Make sure you have all the current SAM data checked out fresh from SVN. Make sure that your working copy does not contain any unversioned, modified, conflicted files.

EDRs

Generating an RDR requires EDRs as input. EDRs should not be kept in SVN, and so you will probably need to download them separately. The process for getting EDRs is the same as it is for SAM Ops. Create an edrhub folder in the fmdata branch, and inside it run `fei5kinit` (to log in), `feiget.py` (to download the RDRs), followed by `movefei.py` to move them to the appropriate telemetry directories.

An alternative approach is to get someone on the SAM team to generate the EDRs and submit the TIDs with them included. I usually contact Benito Prats who is part of the SAM team, he can usually get them updated in a couple of days so just let him know ahead of time.

RDR Configuration Files

For each of the directories in the release, you will need to create a file named "rdr.config". This file contains information that cannot be obtained automatically by the RDR generator, or at least could not at the time the software was written. It is a newline-delimited CSV file, with the first column as keys and the second as values. I recommend you copy an existing one into each of the new TID directories, and then edit each one to fill out the information. All of the following lines need to be present:

```
SOURCE,F
MSL:SAM_GC_COLUMN_NUMBER,5
EXPERIMENT_TYPE,SPYR
VERSION,1
RELEASE,XX
PYRO_OVEN_NUMBER,1
```

- The **SOURCE** field is a one-character description of where the data came from: calibration data ("C"), ATLO data ("A"), testbed data ("T"), or flight data ("F"). I do not know if we will ever archive a data set from any one of these alternate sources.
- The **MSL:SAM_GC_COLUMN_NUMBER** is the number of the GC column. You can determine it by searching the message log with "tmmsg.py column".
- The **EXPERIMENT_TYPE** is a four-character description of the type of experiment. The values are defined by the SIS, and in the code they are defined by the SAM_EXPERIMENT_TYPES dictionary. They can take the following values. If you are unsure of the classification for a given experiment, ask Heather. Note that some of these will never apply to flight data (but could in theory apply to testbed or calibration data).
  - SPYR: Solid sample pyrolysis with GCMS
  - SDER: Solid sample derivatization
  - CSOL: Solid sample calibration
  - ADIR: Direct atmospheric measurement
  - AENR: Atmospheric enrichment
  - AMET: Atmospheric methane enrichment
  - ANGE: Atmospheric noble gas enrichment

    - SCMB: Solid sample combustion
    - CGAS: Gas calibration
- The **VERSION** field is the release version of the data set. It should start at 1, but if you ever re-release a data set for some reason (usually because there was an error in a previous delivery), you increment it.
- For **RELEASE**, put the current release number of the delivery. (Delivery due 11/4/16 is release 13).
- The **PYRO_OVEN_NUMBER** field is more difficult to obtain. I find it much easier to set it to 1 at first and debug it as I do tm2rdr.py. I will explain how this is done below during the processing part and will include what needs to be done when pyro oven is set to 2.

Once you create these configuration files, commit them to SVN.

High-level GC Products

Jean-Yves' inputs are the easiest to include, because he delivers them directly to SVN. For each TID he was assigned, he creates a directory called rdr_gc. The directory should have files named "notes.txt", "noise.csv", "species.csv", and "species.jpg". To my knowledge, he creates all four of these files for each TID. All you have to do is make sure that the files are there and named correctly.

High-level QMS Products

Heather does not have SVN access, and her deliveries are small enough for email. She generally delivers her products as ZIP files, but names them .piz so they don't get block from the email server, so just rename them back to .zip and they should extract easily. For each TID she delivers, create an rdr_qms directory in that TID, and unzip the contents of her ZIP file into that directory. Her files should only have the following names: "NOTES.TXT", "ATMCOMP.CSV", "ISOTOPE.CSV", "EGA.CSV", and "EGACOMP.CSV." If you are unsure what a file is supposed to be called, ask Heather.

High-level TLS Products

Greg does not have SVN access, either, but his deliveries are so large he cannot email them. He usually uploads the deliveries on dropbox and will send you the link via email to the products. For each TID, make an "rdr_tls" directory in that TID directory, copy the contents there, and add them all to SVN. Make sure the three high-level products are named "notes.txt", "abundance.csv", and "ratios.csv", respectively. (He will prefaces them with the TID and names the notes file TID.txt.)

Creating the RDRs

Once the inputs are created, the RDR can be created with a single command.

Change directories to the `fmdata` branch of the working copy. individually go into each of the TIDs and run `tm2rdr.py` and fix any errors that come up.

While running tm2rdr.py and you have pyro oven set to 1 you may see the following error:

```
INFO Processing QMS science data
Traceback (most recent call last)
```

```
File "/Users/briancorrigan/labcode/699util/scripts-pds/tm2rdr.py", line 2017, in <module>
  exit(main())
File "/Users/briancorrigan/labcode/699util/scripts-pds/tm2rdr.py", line 2003, in main
  process_qms_data(tmfile, rdr)
File "/Users/briancorrigan/labcode/699util/scripts-pds/tm2rdr.py", line 1371, in process_qms_data
  pyro_time, pyro_temps = zip(*pyro_temps)
ValueError: need more than 0 values to unpack
```

The usual fix to this is that the pyro oven is actually 2 so do the following:

- `tmfields.py --sclk 86 223 > tmfields.txt`
- Go into tmfields.txt and delete the first three lines that begin with # so it is just the 3 columns of data.
- change PYRO_OVEN_NUMBER in rdr.config to 2
- re-run tm2rdr.py

If this still does not fix it, just set PYRO_OVEN_NUMBER to 0 and it will just skip over them (I only had to do this once).

If you want to run all the TIDs at once you can try by using runall.py but I have had errors with this in the past and it is easier to debug issues that come up with pyro ovens by doing them individually.

You should have already created the pds_MMM-NNN.txt file. You can now use it as an input to a `runall.py`.

Execute `runall.py -d pds_MMM-NNN.txt tm2rdr.py`. Running this command will take a long time. The program will generate RDRs for each TID, one at a time. You should go get lunch or do something else. If anything goes wrong, all progress will cease, and you will probably have to fix something in `tm2rdr.py`.

If an error comes up while processing QMS data, this is probably because you indicated pyro oven 1 was on when really it was pyro oven 2, so modify the rdr.config file and change pyro oven to 2.

Notes for Debugging

To create an RDR for a single TID, just run `tm2rdr.py` just like you would any other python script.

If you need to re-run the RDR generation script over again, you should run `rdrclean.sh` for each TID before re-running. You will notice the second time you run, the script will run must faster. Each major step in the RDR generation process will create a `.pickle` file, which is basically a large stored Python object. If you need to change sections of the code that create the data that goes into these files (e.g., housekeeping extraction), you should delete the associated pickle file.

Assembling and Delivering

Creating an Archive Directory

Download the mslsam_1xxx directory from the current archive:

```
wget -r -nH --cut-dirs 2 -l 2 ftp://pds-geosciences.wustl.edu/msl/msl-m-sam-2-rdr-l0-v1/mslsam_1xxx/
```

Make sure the data folder is empty. Move all the files out of the index directory to a temporary place. The tools will generate the delta rows from this delivery and you will have to manually add them to these index files later.

Summarize Your Changes

Edit the file mslsam_1xxx/errata.txt. In "SECTION A", copy and paste an entry from a previous release and put it at the top of the section. Fill out the information with the current release date (consult Joy's email) and the changes in this delivery. If you did not update/change any old files, you can write "N/A" under "REASON FOR UPDATES". Save the changes.

Install Software

The first time you do a delivery from a computer, you will need to install `VTool`. This can be downloaded from the PDS website here: http://pds.nasa.gov/tools/label-validation-tool.shtml. To install it, unzip the TAR archive and put it somewhere on your filesystem. I put mine under /usr/local. Then, add the archive's "bin" directory to your path.

You will also need to download and install md5deep. You can get the source code here: http://md5deep.sourceforge.net/. It's been a long time since I installed it, but I imagine it's a normal `./configure`, `make`, `[sudo] make install` installation.

Link the new deliveries

Navigate to mslsam_1xxx/data. Run the following command with each TID in the delivery as an argument:

`rdrlink TID1 TID2 TID3 TID4 ...`

This will create symbolic links to the RDR products you just made.

Update Index Files

This step requires that you have the index files you downloaded from the archive and moved aside.

Start by going into mslsam_1xxx/ folder. Run `rdrindex.py` which should add 8 files to your index folder (4 lbl files and 4 tables for each level). Next, open a new folder window and go to the previous delivery's index folder. You are just going to concatenate the old folders index tables to the new ones.

- open PREVIOUS_RELEASE/mslsam_1xxx/index/l0_index.tab
- open NEW_RELEASE/mslsam_1xxx/index/10_index.tab
- copy every line in previous release file
- paste it on the first line of the new release file
- repeat for remaining tables

Now all that needs to be done is to update the label files to match the table files.

- open NEW_RELEASE/mslsam_1xxx/index/l0_index.lbl
- open NEW_RELEASE/mslsam_1xxx/index/10_index.tab
- indicate how many lines are in the index table
- change FILE_RECORDS in the label file to the number of lines that were in the index table

- change ROWS in the label file to the number of lines that were in the index table

Archive is now ready to be packaged and released.

Package the Archives

Navigate to the directory that contains mslsam_1xxx. Run `rdrpackage` which should create a "mslsam_1xxx.tar", "mslsam_1xxx_manifest", and "mslsam_1xxx_checksum". On Mac, you should be able to control + click the .tar file and hit "compress" which will make a much smaller .tar.zip.

Deliver the Archive

Now all you need to do is use an SFTP client to connect to wuftp.wustl.edu and copy the .zip as well as both the checksum and manifest over. Email Susie and let her know you have made the delivery and fix any errors that she has.

BELOW ARE OLD INSTRUCTIONS FOR PACKAGING I NO LONGER DO.

Navigate to the directory containing mslsam_1xxx. Create a directory called "rdrstage". Now, run the following command (with each TID in the delivery specified): `rdrsemipackage TID1 TID2 TID3 ...`. This will create a tar.gz archive in the rdrstage folder. If you want to include older TIDs in the delivery (perhaps because you reprocessed them), include them in this command as well.

The rdrsemipackage not only archives the files you want to deliver (and excludes the other stuff), but it also creates a "manifest" file and a "checksum" file, both of which Susie needs to validate we got her everything correctly. Look at the script to see how these files are created. There is also an rdrpackage script, which I used for the first delivery. I have not needed to use it since.

Revision #2
Created 23 March 2023 14:45:12 by Nick Dobson
Updated 13 April 2023 15:21:48 by Nick Dobson