Project Docs:Ingest SDS

From WiseWiki

Wide-field Infrared Explorer (WISE) Ingest Subsystem Design Specification

Prepared by Tim Conrow

Contents

Signatures

_____

Roc Cutri WSDC Manager

_____
Tim Conrow WSDC Architect, Ingest CogE

Revision history

Date	Version	Author	Description
2008-10-14	0.1	Tim Conrow	Initial Draft

Document number

WSDC D-D016

Scope

This document describes the design of the Ingest component of the WISE Science Data System (WSDS). Ingest of both MOS ancillary data nd HRP telemetry (science image) data are covered.

Since most of the function of Ingest is managing interfaces, that is mostly what is documented here. A few details of the implementation are included where they are deemed important for clarification or for the sake of having a record of some tricky issues.

Introduction

Subsystem Overview

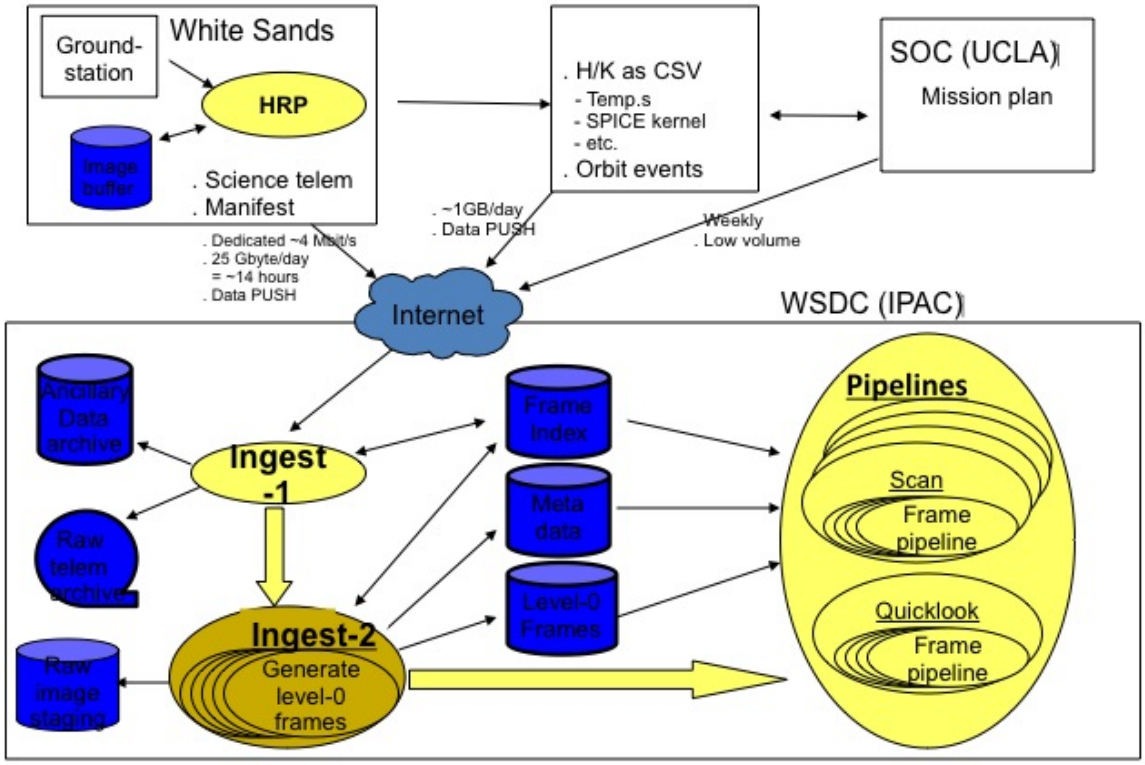
Ingest has six primary functions:

1. Detect data transferred via FastCopy to the WSDC ingest server from outside sources and move the data to internal archive locations ready for further processing.
2. Interpret mission ancillary idata (MAD) sent by MOS and constructing internal databases for use in data processing
3. Depacketize and decompress image data from the HRP and correlate it with mission ancillary data to create level-0 images (L0)
4. Initiate downstream Quicklook and Scan/frame pipeline processing
5. Write/update frame data to the Frame Index (FIX)
6. Provide QA metrics on the L0 images, mission ancillary ancillary data, and the ingest process

Ingest carries out these functions in three distinct stages:

- Data transfers are handled asynchronously by two small daemons that trigger subsequent steps
- MAD ingest when new MOS data is transferred
- L0 ingest when a HRP data is transferred; this stage also does pipeline kickoff and FIX updating

All three stages write QA data, though most comes from the final stage.



Applicable Documents

- WSDC Functional Requirements (http://web.ipac.caltech.edu/staff/roc/wise/docs/mgmt/WSDC_Functional_Requirements_all.pdf) (WSDC D-R001)
- WSDS Functional Design (http://web.ipac.caltech.edu/staff/roc/wise/docs/WSDS_FDD_v1.pdf) (WSDC D-D001)
- MOS/GDS Interface Control Document (JPL D-34372)
- WISE – CCSDS Interface Control Drawing (BATC Drawing Number 2216143), especially section 3.4.2
- Wide-field Infrared Survey Explorer Time Correlation Tool L4 Software Requirements and Description Document
- WISE L2.5 C&DH High Rate Data Processor Requirements

Acronyms and Definitions

API
: Application Programming Interface. The means by which an application accesses the functionality of a library or module.

Bandframe
: Image for a single band for one exposure.

CCSDS
: The Consultative Committee for Space Data Systems. Refers to the standard defining the packet structure used in WISE telemetry, in particular the source packet structure encoding WISE science images delivered by the HRDP at White Sands to the WSDC.

CK
: A SPICE C-kernel, describing spacecraft orientation (i.e. the boresite).

CSV
: Comma-separated values, a file format.

Delivery ID
: A date/time-based unique identifier for each delivery. <*YYDDD*>M<*HHMMSS*> for MOS deliveries, and <*YYDDD*>T<*HHMMSS*> for HRP deliveries.

Delivery name
: The base part of the file name as delivered by MOS to the WSDC. This name is specified in the MOS/GDS ICD referenced above. A typical delivery name might be WIS_EOS_SURVEY_PLAN_<*YYYY_DDD_HH_MM_SS*>.txt .

ET
: Ephemeris Time. See the Wikipedia ET page (http://en.wikipedia.org/wiki/Coordinated_Universal_Time) .

Frame
: Detector data from a single exposure. Might refer to a single band, or to all available bands, depending on context.

Frame index (FIX)
: A database storing meta-data on the characteristics, location and processing status of all framesets received at the WSDC.

Frame, Level-0 (L0)

Same as raw frame but with 4-byte floating point pixels (BITPIX=-32) with select MAD added to headers. This is the archival frame format.

Frame, Raw

Detector data exactly as assembled from science source packets. Pixels are unsigned 2-byte integers. Bands 1-3 frames are 1024x1024, band 4 is 512x512.

Frame Number

The number associated with a particular sequential time interval counted from the start of a scan. The frame number is a unique number associated with each frame within a given scan. Frame numbers are **not** necessarily sequential.

Frameset

Images in all available bands for one exposure.

HK, H/K

Housekeeping telemetry, sent to the WSDC as CSV files.

HRDP, HRP

High Rate Data Processor. The server and associated storage at White Sands which autonomously receives and processes WISE telemetry and pushes the reordered, decommutated source packets either to the WSDC or the JPL MOS.

Job

A process or logical collection of processes and threads performing a computational task.

LSK

A SPICE leap second kernel, describing the translation between ET and UTC

MAD, Mission or MOS Ancillary Data

Meta-data related to processing image data derived from spacecraft and payload housekeeping and PGEN and other MOS products. Examples: the VTC, SPICE kernels (derived from ADCS telemetry), instrument temperatures, etc.

Module

The level of functional grouping below sub-system.

MOS

Mission Operations System. The group/system at JPL responsible for operating the WISE satellite, including commanding and managing telemetry.

NAIF

A toolset and library for handling and using various SPICE kernel files. See the NAIF website at JPL (http://naif.jpl.nasa.gov/naif/) .

PEF

The Predicted Event File, a list of on-orbit events and associated start/stop times produced by the sequence generation process.

Scan

A period of observation running from one ecliptic pole crossing to the next. I.e. about half an orbit.

Scan ID

A unique ID associated with each science scan. Special scan IDs are also assigned to non-science scan periods, so effectively all frames can be assigned a scan ID.

Source Packet

A unit of telemetry in a standard format which contains primary telemetry data, such as image data or housekeeping engineering data.

S/C

Spacecraft

SCLK
: A SPICE clock kernel describing, with the LSK, the translation between VTC and UTC

SPK
: A SPICE kernel describing the ephemerides of various orbiting bodies, particularly the planets and the WISE satellite.

SOC
: Science Operations Center. The group/system at UCLA responsible for generating the survey plan and the ADCS commands necessary to implement it.

Sub-system
: The highest-level division, or grouping, of functions within the WSDC. See also "module."

UTC
: Universal Time Coordinate. See the Wikipedia UTC page (http://en.wikipedia.org/wiki/Coordinated_Universal_Time) .

VTC
: Vehicle Time Code. The time stamp marking frame data and ancillary data generated on-board the S/C. Converted to UTC in processing.

WSDC
: WISE Science Data Center. The group at IPAC responsible for processing of WISE science data and releasing data products.

WSDS
: WISE Science Data System. The collection of software, hardware, and facilities which performs all data processing at the WSDC.

Typography

Here are the typographic conventions followed in this document:

- A "typewriter font" is used to display standalone, multiline literal text, such as table or code fragments. E.g.

```
This is literal text.
```

- A token which will be replaced by a variable value is surrounded by angled brackets and in italics: e.g. <*token*>. This is used mostly in file names. A constant constant value which will not change, but for which the value is not yet known, will be in bold: e.g. <***constant***>.
- A special literal word will be in bold: e.g. **literal**. These are words which will be present in tables, code, databases etc. exactly as presented. Note that bold text may also be used for emphasis and for terms in an equation.

Input to Ingest

- Transfer manifest files: transfer contents
- HRP packet FE1A-D files: band 1-4 image data.
- SCLK, LSK files: clock kernels.
- PEF files: predicted events.

- CK files: S/C orientation.
- Planet SPK files: planetary body position.
- S/C SPK files: spacecraft position.
- H/K CSV files: housekeeping telemetry.
- Ground track files: S/C longitude, latitude, and altitude, and distance from SAA boundary.

Output from Ingest

- Level-0 image database
- SCLK, LSK database
- Event table
- CK database
- Planet SPK database
- S/C SPK database
- H/K database
- Ground track database
- Meta-data files

Ingest Processing

Ingest Overview

Data Transfer

Data Transfer Overview

Data is transfered to the WSDS via FastCopy (http://www.softlink.com/fastcopy_techie.html) to the ingest server (hostname WSDCIN) situated outside IPAC's Internal Network Service zone (INS), but behind the Perimeter Network Service firewall (PNS). All data is pushed by external users to WSDCIN and pulled by internal users (WSDCIN has no write access to the INS). FastCopy does data integrity checking and transfers to a temporary file until the data for each file is known to be complete and correct, at which time the file is moved (atomically) to its final location.

Data Locations on WSDCIN

All incoming data directories are subdirectories of /home/wsdcin/inbox. Here are the subdirectories:

- tmp: temporary FastCopy transfer space
- hrp: HRP transfer destination
- hrp/queue: location of completed HRP transfers yet to be transfered to the WISE ops database
- hrp/archive: location of completed HRP transfers which have been transfered to the WISE ops database
- mos: MOS transfer destination
- mos/queue: location of completed MOS transfers yet to be transfered to the WISE ops database

- mos/archive: location of completed MOS transfers which have transfered to the WISE ops database

The /home/wsdcin/inbox directory is exported read/write to the WISE ops network in the INS.

Transfer Protocol

The FastCopy transfers will create subdirectories within **hrp** or **mos** based on the date and time of the transfer of the form *YYYY_DDD_HH_MM_SS* where *YYYY* is the year, *DDD* is the day number, *HH_MM_SS* is the time in UTC hours, minutes, and seconds.

Data transfers begin with a manifest of all files to be transferred and their sizes. and end when the transfer of all files in the manifest has been completed. All files and the manifest will then be present in the date-named subdirectory. Multipe transfers may occur at the same time as long as all concurrent or overlapping transfers target unique directories.

Detection and Disposition of Data Transfers

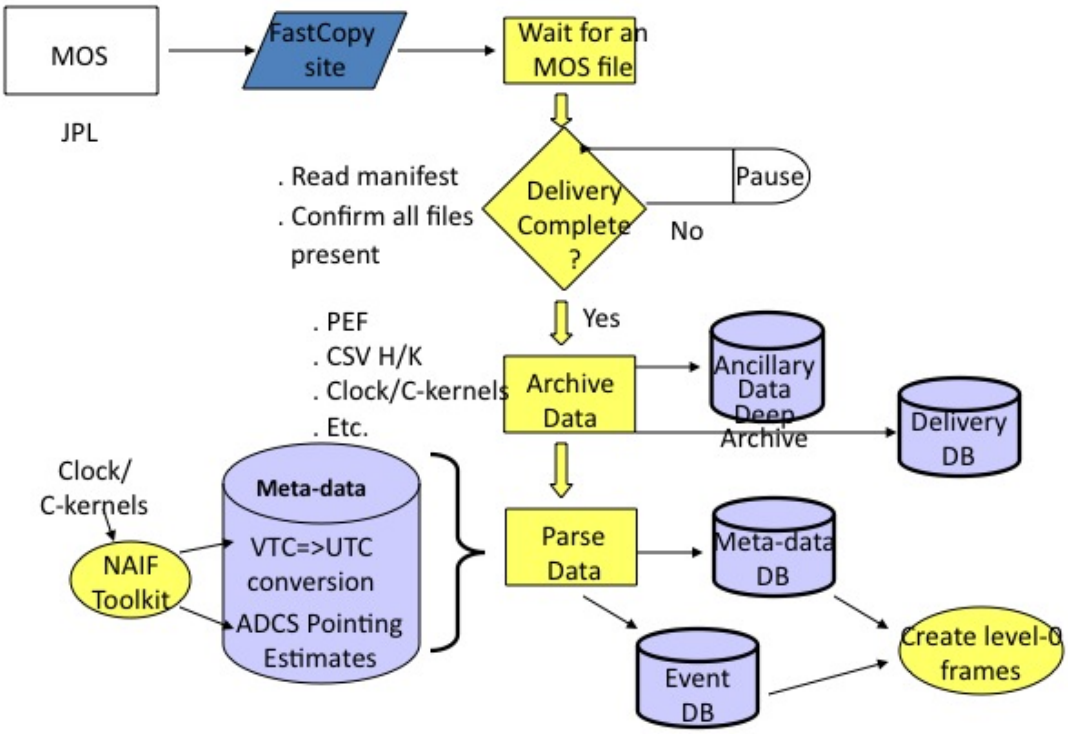
Ultimately, all transfered data must be pulled into the INS to archive space in the WISE ops database. This is accomplished by a daemon ingest process called **watchd** running on a node of the WISE cluster which watches the various subdirectories of /home/wsdcin/inbox looking for the appearance of a new transfer directory. When it sees such a directory, **watchd** writes a **watchd.txt** file to the directory and spawns a child daemon called **monitord** to monitor the transfer. **Monitord** reads the manifest file, and when all files listed in the manifest are present, the transfer subdirectory is moved (not copied) to the **queue** subdirectory and **monitord** then copies the data to a parallel directory in the ops archive under /wise/fops/ingest/delivs/*YYDDD*/*YYDDD[TM]HHMMSS*, where *T* is used for HRP transfers, and *M* is used for MOS transfers. When the copy is complete, the transfer directory is moved to the sibling **archive** directory. **Monitord** then kicks off the ingest pipeline **ingestpipe** with parameters set to ingest the just-completed transfer using the copy in the ops database.

The archive directory is backed up daily, and once a raw transfer directory has been backed up, it is removed from the archive directory.

MOS Data Ingest

MOS Data Ingest Overview

The Mission Operations System at JPL is responsible for delivering all data other than science images that are necessary for data processing and evaluation. Taken together this data is called the Mission Ancillary Data. It is delivered from JPL to the WSDC via FastCopy.



MOS Data Delivery Cadence

SCLK and LSK clock kernels will be updated irregularly and asynchronously. PEF (event) and ground track files will be delivered twice a week, associated with the command sequence generation cycle. All other MOS ancillary data are derived directly from a individual downlinks and will be transferred together preceding the transfer of the associated HRP data.

- LSK files: probably about two or three times over the life of the mission
- SCLK files: probably about once per week
- PEF, Mission plan, ground track and other SEQGEN/WOE files: twice a week
- CK, SPK files, HK telemetry files: delivered after each downlink, approximately four times a day, arriving prior to HRP data

MOS Data Deep Archive

The MOS group at JPL will send ZIP archive files which are regarded by Ingest as opaque and are to be deep archived. The arrival cadence of these archve files is unknown at thos time.

Directory: /wise/fops/ingest/delivs/mos/deep File: As delivered

MOS Data Requirements

Every ingest of HRP data, i.e. WISE science images, requires that the following MOS ancillary data be up to date, i.e. the most recent data available must already have been transferred and ingested:

- LSK, SCLK files
- PEF files
- Ground track files
- Planet SPK files
- CK files
- HK files

If any of these files are missing or out of date, science processing or evaluation of the science images will be incomplete. Files not mentioned above are for reference or contingency use.

Ancillary Data Database Generation

All MOS deliveries are handled by a common ingestpipe command line, simply specifying all delivered MOS files. Each file type is then determined by the file name suffix, and each different type is handled with its own clause in ingestpipe. I.e. ingestpipe does not require knowledge of the delivery cadence; it simply processes whatever it sees in a particular delivery. Here is how the various MOS data are handled.

LSK, SCLK DB

The clock kernel files are cumulative, that is they include data that covers the whole mission up to the final time covered by the file. Thus each new file simply replaces previous versions. This is accomplished by keeping a symlink with a standard name pointing to the most recently file.

- Directory: /wise/fops/ref/mos/naif
- File: wise.tsc (SCLK) and wise.tls (LSK)

All LSK and SCLK files are on-line in their respective delivery directories. A link to the previous files used is kept in the **prev** subdirectory when an update is made.

SPK DB

SP-kernels (like C kernels) form a complete database when all SPK files are read in time order by the NAIF routines. Thus delivery of an SPK file simply amounts to copying the most recently delivered file into a standard directory.

- Directory: /wise/fops/ref/mos/naif/<*delivery_group*>, where the <*delivery_group*> is the *YYDDD* portion of the delivery ID
- File: <*delivery_name*>.bsp, where <*delivery_name*> is the file name as delivered to the WSDC

The NAIF routines are directed by ingestpipe to read all C-kernels within a selectable time window. By default the most recent 30 days of C-kernel data are read. If there is overlapping data, more recent data supplants older data.

Ground Track DB

The ground track file contains record-oriented, ASCII time-stamped spacecraft longitude, latitude, altitude, and SAA distance information. The sampling frequency is TBD. No cumulative ground track database is currently contemplated. Ground track files would simply be stored for each delivery.

- Directory: /wise/fops/ref/mos/track/<*delivery_group*>, where the <*delivery_group*> is the <*YYDDD*> portion of the delivery ID
- File: <*delivery_name*>.trk, where <*delivery_name*> is the file name as delivered to the WSDC

Exact file format and name is TBD.

PEF Event DB

The Predicted Event File is a SEQGEN product which records start and stop times for a set of interesting on-orbit events. Events of greatest interest to ingest and pipeline processing are extracted and reformatted into an event table, the columns of which record the event name and start and stop times for all events of interest. The orbit number and scan ID of the scan in which the event occured are also recorded.

PEF files overlap, but each contains a time range (in the PEF file header) which defines a set of events strictly unique to this PEF file. Thus PEF files from separate deliveries are merged by appending new events from this exclusive time range to the end of a single accumulating event table. Since each row of the event table contains both start and end times, events which start in one PEF file but end in another have to be handled by searching the existing cumulative event table for unterminated events and resolving them in the new PEF file. The previous event table is stored in the **prev** subdirectory.

Events which are missing either their starting or ending time are identified with special names in the table and produce warning messages.

It's possible the cumulative event table will become too long to manage efficiently, depending on the number of events per orbit we end up saving. If this happens, the event table can be broken up into overlapping weekly or monthly time ranges. Only the most recent table needs to be merged with incoming data.

- Directory: /wise/fops/ref/mos/events/
- File: events.tbl

Events of interest:

SCAN
: Science scans.

NEP, SEP
: North/South ecliptic pole crossings.

ANODE
: Ascending node crossings.

SAA
: SAA crossings.

TDRSS
: TDRSS contacts.

ANNEAL
: Detector anneals.

MOON
: Lunar avoidance zone crossings.

To be added
: Other detector commands? Momentum dumps? ADCS states/commands? Other bright planet impingements?

CK DB

C-kernels (like SP kernels) form a complete database when all CK files are read in time order by the NAIF routines. Thus delivery of a CK file simply amounts to copying the most recently delivered file into a standard directory.

- Directory: /wise/fops/ref/mos/naif/<*delivery_group*>, where the <*delivery_group*> is the <*YYDDD*> portion of the delivery ID
- File: <*delivery_name*>.bc, where <*delivery_name*> is the file name as delivered to the WSDC

The NAIF routines are directed by ingestpipe to read all C-kernels within a selectable time window. By default the most recent 30 days of C-kernel data are read. If there is overlapping data, more recent data supplants older data.

Mission Plan DB

Mission Plan files, which originate at the SOC (UCLA), are test files passed to the WSDC via MOS. At present, the plan is for these files to simply be archived for future reference.

- Directory: /wise/fops/ref/mos/plan/<*delivery_group*>, where the <*delivery_group*> is the <*YYDDD*> portion of the delivery ID
- File: TBD

HK DB

The HK data is delivered to the WSDC as a series of CSV files, one for each telemetry point, correlating the sample time to the resulting telemetry point value. To compose a cumulative, SQL-searchable DBMS of the contents of these various files, they will be read into an internal data structure which is indexed on time. The various telemetry points are sampled with various frequencies, so to make use as simple as possible, all must be resampled onto a common time step without wasting too much space. This is accomplished by having separate tables written for each native frequency which has columns for all HK values sampled at that frequency. These tables all have a column for the time of each sample. Then, a single join table is written which has time intervals at the maximum frequency, probably 1 Hz, each row of which has the times of the rows

from the other tables which are nearest in time. All HK telemtery for a given time can then be assembled by using the join table to produce a view of all tables at 1 Hz.

E.g. I postulate here three telemetry points sampled at 1 Hz, 0.5 Hz, and 0.2 Hz. Following these tables is the join table. Note that the time formats are purely illustrative, not literal.

Telemetry point Foo; 1 Hz:

```
| time | value |
  1.1    12.3
  2.1    14.3
  3.1    15.6
  4.1    17.2
  5.1    20.1
  6.1    18.9
```

Telemetry point Bar; 0.5 Hz:

```
| time | value |
  0.3    0.4
  2.3    0.6
  4.3    0.3
  6.3    0.7
```

Telemetry point Baz; 0.2 Hz:

```
| time | value |
  1.7    -12
  6.7    -120
```

The join table for this interval:

```
| time | time_1hz | time_p5hz | time_p2hz |
  1.0    1.1        0.3         1.7
  2.0    2.1        2.3         1.7
  3.0    3.1        2.3         1.7
  4.0    4.1        4.3         1.7
  5.0    5.1        4.3         6.7
  6.0    6.1        6.3         6.7
```

Indexing the time columns in all tables will make the joins quick. With this join table one could construct a SQL **select** statements to present all telemetry values on 1 second intervals regardless of their real sampling interval.

It may be necessary to have separate tables for a given frequency if different telemetry points sampled at that frequency were sampled at different phases. E.g. it may be that 0.2 Hz telemetry from the instrument and that from the S/C are sampled at different times. Another complication is that the S/C clock will be periodically adjusted, which means the sampling phase will change midstream. As long as this jump is reflected uniformly across all telemetry points, this should not be a problem.

Probably there will be more telemetry points delivered than will be needed in the cumulative DBMS.

The cumulative DBMS:

- Directory: /wise/fops/ref/mos/hk
- File: hk.db (while SQLite is the DBMS engine)

The HK file archive:

- Directory: /wise/fops/ref/mos/hk/<*delivery_group*>, where the *delivery_group* is the <*YYDDD*> portion of the delivery ID
- File: <*delivery_name*>.csv, where <*delivery_name*> is the file name as delivered to the WSDC

The HK archive may be cleaned up as necessary to save disk space.

MOS Ingest Meta-data Table Output

Various summary statistics and other data will be written to a meta-data table for each MOS ancillary data delivery.

- Directory: /wise/fops/ref/mos/meta/<*delivery_group*>, where the <*delivery_group*> is the <*YYDDD*> portion of the delivery ID
- File: <*delivery ID*>-meta-ingest.tbl

HRP Data Ingest

HRP Data Ingest Overview

The High Rate Processor at White Sands receives telemetry from the ground station, strips off communication packet wrappers and organizes the underlying source packets into four output files, one each for all images in a given band included in that transfer. This is a **delivery**. In each band's file for a given delivery, images are stored as a stream of compressed images tagged with limited meta-data--mainly the image's VTC (time)--in CCSDS packet format. The ingest process strips off the source packet meta-data and assembles, decompresses and rotates the images into a standard format and orientation. Ingest then associates (correlates) it via the VTC with various MOS ancillary data. The assembled image is written as a FITS file with selected ancillary data included in its header and in a meta-data file. Also added to the image FITS header are various calibration constants, such as the image scale and distortion. The FITS image and meta-data files are then added to the Level-0 Archive.

Each delivery will, on average, include data from about 8 scans. A Quicklook Pipeline run is submitted for a selected subset of these frames. Any or all of these may be incomplete due to missing images or missing ancillary data (without which a level-0 image cannot be created). Completed scans are submitted for processing by the Scan Pipeline. The scan IDs of incomplete scans are kept in a delivery database. When Ingest is run on subsequent deliveries, the delivery database is consulted for unprocessed scans, and those which are completed in the current delivery, and those which are still incomplete after a TBD interval has elapsed, are submitted for scan processing and the delivery database entry is updated.

Key information about each frame added to the Level-0 Archive is stored in the Frame Index, a DBMS used to track meta-data and processing status for ingested frames.

Any period within a scan longer than the inter-frame interval (about 11 seconds) indicates a frame is missing. An accounting of missing frames, along with other statistics, is written to a meta-data file for each delivery.

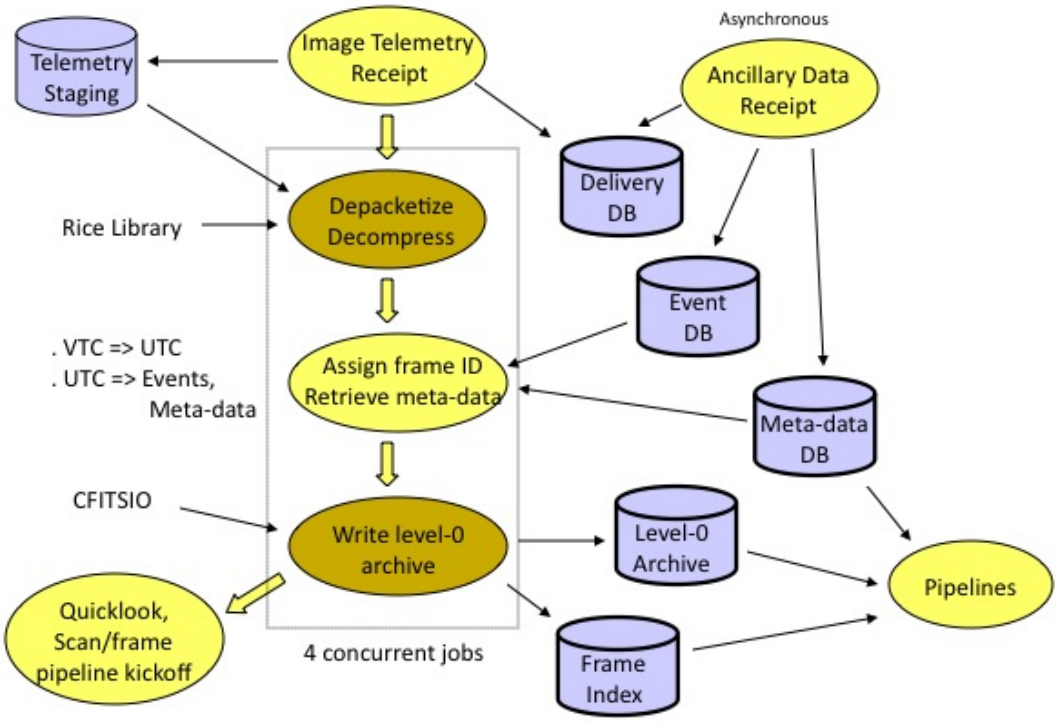


Image Reconstruction

The telemetry files sent from the HRP are time-ordered CCSDS source packets for one band covering the time range for that delivery. The packets are read as streams, i.e. the files are not read in all at once but packet-by-packet. The CCSDS packet headers are read to establish the start and end of images, the VTC, whether it is compressed or not, and how much data is in the terminal packet (the only one which might have fill data). The data portion of each source packet (i.e. the packet headers are stripped off) for a given frame are appended to one another to produce a complete frame, compressed or uncompressed. If compressed, the frame data is passed to the Rice decompression library to produce a decompressed frame. Packet header data, mainly the VTC, is saved for each frame for later use.

This is the key information in each packet header:

- VTC; The vehicle time code.

- Sequence count; sequential packet number count.
- Grouping flag; 1=first packet for an image, 0=internal packet, 2=last packet.

The first packet for an image will have a group ID of 1 and a sequence number of 0. Packets are read and appended until a packet with a group ID of 2 is encountered. At this point a full image has been assembled.

Every packet header is examined for internal consistency by checking the following:

- Do sequence numbers start at 0 and increase sequentially?
- Are all the VTC's for a given image the same?
- Are packets the expected size (1092 bytes)?
- Are all packets full (no fill data) except the final packet?
- Are other CCSDS packet header fields as expected (version, type, apid, secondary header, spare)?

A further check is whether or not the image will decompress to an image of the expected size, since often (but not always) a corrupted compressed image will be have internal inconsistencies rendering it impossible to decompress, or causing it to decompress to the wrong size. Note that sometimes corrupted images will never-the-less decompress with no apparent errors, though the image will display either subtle or gross decompression artifacts. We have no way of checking for this situation and images bearing decompression artifacts are expected to enter the level-0 archive, though this should be fairly rare.

If any of these checks fail, the packet is assumed to be corrupted and processing stops and an error is generated. Ingest will count the errors and report them, but a (presumably) corrupt packet will not halt processing. See below.

If all checks are successful, a raw image will now occupy an internal data structure and is ready for correlation with the ancillary data.

Correlation to Ancillary Data

Image data is useless unless it is associated with certain, key ancillary data. In addition, other ancillary is often or occasionally useful, but not essential. At present, there is no separation the ancillary data into degrees of importance; all expected ancillary data is either present, or processing cannot continue. This simplifies the coding at a small (I hope) cost to robustness. The ancillary data required is listed below. For informational purposes only, they are listed, roughly, in order of importance.

- SCLK and LSK files covering the data's time range to convert VTC to UTC.
- PEF file to record event table entries for
  - the orbit number, and thus the scan ID
  - start/stop times of science scans
  - entry/exit times to/from the SAA
  - anneal times

    - instrument parameter changes
- CK file to convert the VTC to a boresite orientation
- HK files to record temperatures, ADCS state, instrument values, etc.
- Planetary SPK file to establish moon and planet positions
- Ground track to establish distance from the SAA boundary

Other SEQGEN files, such as the mission plan file, will also be delivered, but these are not used in level-0 image construction.

All data listed above are queried by the time for each frame to get the requested data. This data is then saved as meta-data in an internal data structure associated with each frame for later use in creating the level-0 image.

Level-0 Image Creation and Archive Update

With the raw image and meta-data in hand for each image, the level-0 FITS image is created as follows:

- Image pixels are converted from two-byte integers to four-byte IEEE floats.
- The pixels are byte-swapped to the FITS standard.
- All bands are flipped/rotated to align with band 1.
- The header is updated with the meta-data:
    - The VTC is directly converted to UTC using the SCLK/LSK databases. The time is recorded in the FITS header. The time recorded may be adjusted by a constant to produce a time which corresponds to the middle of the integration.
    - The C-kernel is accessed by time to establish the boresite orientation which is used to provide initial FITS WCS values.
    - Events are found by seeing if the frame time lies between the start and stop times in the event record. Any events found are recorded in the FITS header. The most crucial event determined is whether or not the frame is in a science scan. See below for how the critical assignment of scan ID is done.
    - Planetary SPK files are queried by time and any lunar or planetary impingements are records in the FITS header.
    - The ground track entry closest in time is found and the S/C latitude, longitude, altitude and SAA boundary distance will be recorded in the FITS header.
    - The band number is recorded in the FITS header.
- Image pixel statistics are computed and stored in the FITS header.
- The FITS file name and location is determined from the scan ID and frame number (see below).
- The level-0 FITS image is saved.

Some of these various meta-data are also saved to a meta-data file so it can be retrieved without reading the FITS header. Some is also stored in the Frame Index database. See below.

- Directory: /wise/fops/l0/<*scan group*>/<*scan ID*>/fr/<*frame number*>
- File: <*scan ID*><*frame number*>-w<*band*>-int-0.fits

Scan ID and Frame Number Assignment

The event table provides start and stop times for science scans, and with them a unique scan ID for that scan. The scan ID in the event table is derived as follows:

- The orbit number is given for each NEP and SEP crossing listed in the PEF file. These numbers are assigned by the SEQGEN software at MOS. NEP crossings result in integer orbit numbers, SEP crossings result in half-integer numbers. The last such orbit number is remembered as the PEF is read.
- When a science scan event is encountered, it is assigned a scan number of twice the orbit number, AKA the half-orbit number. The half-orbit number is zero-filled on the left to 5 digits.
- To complete the scan ID, a letter is assigned based on sequential count of scans started in this half-orbit so far. So the first scan begun in a half-orbit gets 'a', the next 'b', etc.
- Frames not in a science scan are assigned the half-orbit number and the letter 'x'.

E.g. suppose the PEF lists the orbit number as 1506.5 after a particular SEP crossing. This is half-orbit number 3013. The second scan that start in this would produce the scan ID *03013b*.

Note: There is a built-in assumption in the Ops Archive structure that no more than 4 scans will start in a given half-orbit, i.e. that the highest scan ID letter (discounting 'x') will be 'd'. It should not be possible to get beyond 'b' unless the S/C is coming out of a safing event. It should not be possible to get beyond 'd' unless the S/C enters and exits a safe condition more than once in the same half-orbit.

Frame numbers are assigned based on the time elapsed from start of the scan (as recorded in the event table) to the start of the frame's exposure. This interval is divided by a constant bin size selected for the mission. Call the bin size ***tbin***, the frame time **t(frame)**, the scan start time **t0(scan)**, the elapsed time into the scan **dt(frame)**, and the resulting frame number **n(frame)**. So we have

- **dt(frame)** = **t(frame)** - **t0(scan)**
- **n(frame)** = int(**dt(frame)** / ***tbin***) + 1

tbin must be chosen carefully. It must be less than or equal to the shortest possible interval between frames. This sounds like it should just be the known frame interval of about 11 seconds, but the fact that the VTC to UTC conversion can change abruptly by an unknown amount makes things more complicated. Suppose the VTC drifts with respect to UTC by 1 second a week and that MOS corrects the SCLK kernel twice a week. In this case each adjustment could be up to around 0.5 seconds. In this case, if the adjustment occurs during a science scan, there will be two frames whose UTC interval will be 10.5s instead of 11s. If one was unlucky, these two frames might end up with the same frame number using the equation above if one used a ***tbin*** of 11s. This would lead to an Ingest error.

Frame Index Update

Some of the meta-data associated with each level-0 frame is written to the Frame Index, a set of RDBMS tables that track vital information and progress of every frame as it is processed. Ingest data that end up in the frame index is:

- Scan ID, orbit number, and frame Number
- Frame time in UTC
- The band
- The delivery ID
- The delivery time, file name, and file size
- The ingest success/failure status
- The level-0 file name
- The ADCS boresite RA, Dec, Twist
- A spatial bin
- The scan start/stop time
- The preceding and subsequent frames in the scan
- Some frame statistics
- Some events associated with the frame
- Some H/K data associated with the frame

The frame position, spatial bin, and frame time columns are indexed.

Subsequent processing, such as by the Scan Pipeline, will update other columns in the Frame Index.

Recovery From Telemetry Errors

It will be common for the image telemetry packet stream to generate errors do to packets being corrupted or lost in transmission. Whent his happens, processing must continue so all available frames are extracted. After an error occurs, the image being assembled is discarded and processing continues at the next source packet. A cascade of errors are expected at this point since we may be resuming processing in the middle of an image; these errors are suppressed, with each error simply resulting in skipping the current packet. Processing continues with errors suppressed until a valid stream of packets is traversed in full and a valid image is assembled.

Frame Accounting

Frae accounting is tricky. When frames are lost in communication from the satellite to the WSDC, they may be retransmitted. This means frames for a any given scan may occur not only separated by multiple deliveries but out of order. In any given run of Ingest it is impossible to know for sure what frames in a scan will ultimately appear. Thus the only useful time to do frame accounting is when it is known no more frames for a scan will appear (see below in the Scan Pipeline Kickoff section). At this time, one may try to count missing frames in a few different ways.

1. A total expected number of frames for a scan can be computed (based on the length of the scan) and compared to the number of frames in the level-0 archive. These numbers are recorded in the meta-data for the scan. **Note however** that these number will often disagree even though there is no loss of data (see the paragraph on frame numbering in the Scan ID and Frame Number Assignment section). Thus results derived from these numbers showing the loss of one or two frames must be regarded with a somewhat lax attitude.

1. A more reliable result could come from checking that the centers of all images in a scan are within about a degree of one another. Allowing for ADCS errors to two neighboring frames should be further apart than this, so missing frames **internal** to the scan will be correctly identified. Even this may occasionally go wrong if the ADCS is misbehaving during a scan. And it doesn't indicate if frames are missing off the beginning or end of a scan since we don't have an independent record of where a scan should start or end.

1. Another result will come from checking that no two frames are further apart than 11s plus the maximum time adjustment delta. This can discover that frames are missing, but may not be able to accurately determine exactly how many if there is a large gap.

At present only method 1 is implemented.

Quicklook Processing Kickoff

The Quicklook Pipeline is started for every delivery. About 5% of the delivered frames are selected based on location, statistical properties, and observation time. The selection criteria (details TBD) are meant to find frames that will give clean PSF measurements to confirm that the scan mirror is in sync with our scan rate.

The Quicklook Pipeline is comprised mainly of a run of the Scan Pipeline with command line parameters set such that

- no dynamic calibration will be done,
- higher-than-usual SNR thresholds are applied to source detection,
- specialized QA is done, and
- the Frame Index is not updated.

Because good composite scansync results will require combining all the frames in the Quicklook processing together, and because the scansync computation is done per scan, all the frames must be assembled into a single "virtual scan" regardless of their scan IDs. For this purpose the quicklook data is written to directories segregated by delivery ID rather than just scan ID. This makes it easier to select a fake scan ID without fear of duplicating output directories.

- Directory: /wise/fops/ql/<*delivery ID*>/<*scan ID*>/fr/<*frame number*>
- File: <*scan ID*><*frame number*>-w<*band*>-int-0.fits

In this case the scan IDs and frame numbers will be fake. The files will be symlinks to level-0 files with different scan IDs and frame numbers.

Scan Pipeline Kickoff

The Scan Pipeline is started for every scan which has all expected frames, or for which a drop-dead time interval since the last frame to be received has passed. This drop-dead interval is TBD but may be something like 24 hours.

The Scan Pipeline will create the output directories for frames known to be present in the scan and the Scan Piepline command line parameteres will be set to process all frames in the standard, default manner.

- Directory: /wise/fops/scans/<*scan group*>/<*scan ID*>/fr/<*frame number*>
- File: <*scan ID*><*frame number*>-w<*band*>-int-0.fits

The files will be symlinks to level-0 files with the same name.

HRP Ingest Meta-data File Output

Meta-data files are written out in several places.

1. A delivery table is updated to reflect the current state of scans encountered.

   The status is one of

   pending
   : waiting for more frames

   submitted
   : a scan pipeline has been submitted on this scan

   The table will also have delivery and submission times.

   - Directory: /wise/fops/ingest/delivs
   - File: scan_status.tbl
2. A table describing the scans present in the delivery
   - Directory: /wise/fops/ingest/delivs/<*delivery group*>/<*delivery ID*>/
   - File: <*delivery ID*>-scans.tbl

     This same table is copied to the level-0 archive directory for this scan with a different name and filtered to include only the relevant scan

   - Directory: /wise/fops/l0/<*scan group*>/<*scan ID*>/
   - File: <*scan ID*>_<*delivery ID*>-scans.tbl
3. A table listing the frames in the delivery
   - Directory: /wise/fops/ingest/delivs/<*delivery group*>/<*delivery ID*>/
   - File: <*delivery ID*>-frames.tbl

     This same table is copied to the level-0 archive directory for this scan with a different name and filtered to include only frames from the relevant scan

   - Directory: /wise/fops/l0/<*scan group*>/<*scan ID*>/
   - File: <*scan ID*>_<*delivery ID*>-frames.tbl
4. A meta-data table with QA and other high-level data about the delivery
   - Directory: /wise/fops/ingest/delivs/<*delivery group*>/<*delivery ID*>/
   - File: <*delivery ID*>-meta-ingest.tbl

5. A frame-specific meta-data table with information paritally mirroring what is in the FITS header
    - Directory: /wise/fops/l0/<*scan group*>/<*scan ID*>/
    - File: <*scan ID*>-meta-ingest.tbl

Some Ingest Implementation Details

Implementation Overview

The Ingest process is mostly incorporated in the **ingestpipe** Perl program. The transfer process is controlled with the **watchd** and **monitord** daemons, but all other work is done in **ingestpipe**. **Monitord** starts **ingestpipe** when a transfer is complete.

If it was a MOS ancillary data transfer, the **ingestpipe** parameter **-mos_files** will be passed a list of all transfered files. Processing procedes in an event-driven manner from there; i.e. whatever files are present are processed according to its file name suffix. E.g. ***.bc** are C-kernels and processed accordingly, ***.pef** files are processed into event the event table, etc. Since MOS data arrive at a variety of cadences, no global inventory or MOS data is done when a given transfer is complete since we cannot know at any given time whether all needed data has been updated. Instead, when image data arrives from the HRP, the level-0 image creation proceeds as though all needed data are present and ends when a frame is encountered which is missing some required MOS data. It is expected such missing ancillary data will be taken care of in the next transfer since there will be overlap.

For HRP (science image) transfers, the **ingestpipe** parameter **-tlm_files** will be passed a list of all transfered files (there should always be four files transferred). The four band files are processed in their own threads concurrently to improve ingest throughput. To avoid unnecessary I/O, raw images from the decompression process are not written to disk but held in memory and one by one correlated with the MOS ancillary data and written to the level-0 archive. I.e. images are handled as a stream, not en mass. This has implications for what is possible during ingest since no look-ahead logic is possible. A pre-scan of the telemetry files could be done to do certain checks--e.g. the up-to-dateness of the MOS ancillary data--but this is not currently planned.

APIs Used by Ingest

There are three main external libraries which facilitate ingest processing:

- The locally modified Rice decompression library
- The NAIF library
- The CFITSIO library

These libraries have Perl APIs which utilize the XS Perl extension language to connect C stub subroutines to the Perl language level. The Rice decompression library API is in the WISE::Ingest::Decom extension, and the NAIF API is in WISE::Ingest::NAIF (which uses WISE::Ingest::NAIFXS internally). The FITSIO library API is in WISE::FITSIO.

Additionally, there are internal APIs (APIs not involving external libraries) widely used by ingest. These are:

- PDL: The Perl Data Language module that provides fast array computation for pixel handling an manipulating.
- WISE::IPACTbl: Provides high-level IPAC table file I/O.
- threads: This standard Perl module provides threading for concurrent execution of program elements.

Data Formats

There are five broad categories of file formats employed by the ingest process:

- NAIF kernel files
- Miscellaneous MOS ancillary data files
- CCSDS source packet files
- IPAC table files
- FITS image files

All of these formats are defined and documented elsewhere. Specific contents as utilized by ingest are either described above or in appropriate SIS's or the MOS/GDS ICD.