WISE Science Data System Functional Design

Contents

Signatures

_____
Roc Cutri WSDC Manager

_____
Tim Conrow WSDC Architect

Revision history

2007-11-12: Draft 1
2007-12-05: Draft 2

Document number

WSDC D-D001

Scope

This document describes the functional design of the WISE Science Data System. The functional design describes in coarse detail how requirements in the Functional Requirements Document (http://web.ipac.caltech.edu/staff/roc/wise/docs/WSDC_Functional_Requirements_all.pdf) (FRD) will be implemented. For each subsystem a *Subsystem Design Document* provides greater implementation detail.

High-level Subsystem Design

The WISE WSDC functionality is organized in two levels: **subsystems**, and **implementation modules**.

Subsystems serve to group related functions together for purposes of assigning resources and aiding communication. The detailed descriptions below give each subsystem's purpose.

Implementation modules group smaller, closely related sets of functions or capabilities and reflect implementation details. Several implementation modules combine to form a subsystem. The detail sections below describe the purpose, requirements, and products of each implementation module.

Derived High-level Design Drivers

Most of the FRD (http://web.ipac.caltech.edu/staff/roc/wise/docs/WSDC_Functional_Requirements_all.pdf) is implemented by the modules making up the major subsystems. The design linking these sub-systems and modules together is driven by the following derived high-level motivators:

High Data Throughput
: Raw, decompressed telemetry is 50GB/day. Data flow including intermediate products exceeds 500GB/day for the frame pipeline. Since reruns and coadds occur in parallel, total throughput can exceed 1 TB/day.

Low Latency
: Quicklook turnaround is 24 hours from receipt of the raw telemetry.

High Availability
: In addition to low latency, the WSDS will provide services which must be available most of the time. The most important high availablility service is receiving raw telemetry.

Modest Hardware Budget
: WSDS budgeted with commodity, low cost hardware.

Rigid Mission Schedule
: Launch, ground characterization, mission scenario testing, etc. wait for no one.

Heterogeneous Application Development
: WSDS developers have various languages and development modes which they find most productive.

These drivers lead to the following design decisions:

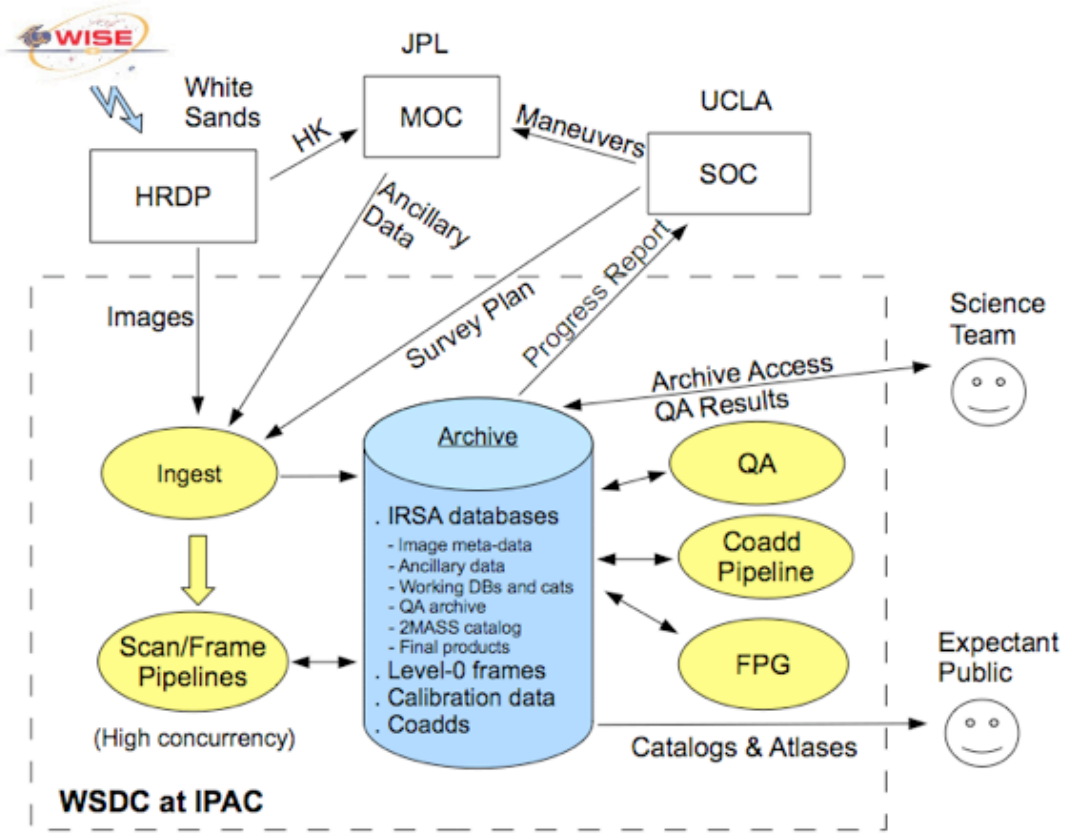
- A compute cluster based on interchangeable, inexpensive nodes linked to high-quality archival disk by a modestly scalable gigabit network. See Hardware Design and Throughput Requirements.
- Executive wrappers which provide a common external interface and services for underlying applications. See Execution Wrappers.
- Pipeline structure allowing high concurrency with frameset-level granularity. See Job Administration.
- Minimal reliance on external services, such as client/server-based RDBMS systems, for operational pipeline runs.
- Emphasize use of node-local storage to reduce network loading.
- Personnel dedicated to operations activities (as opposed to development and analysis). See Operations Concept.
- A robust disaster recovery plan (XXX TBD) including hardware architecture for backup/restore and failover. See Hardware Design.

Major Subsystems

The major subsystems (described in the next chapter) are:

- Executive Functions
- Ingest Functions
- Pipeline Functions
- Quality Analysis Functions
- Final Product Generation Functions
- Archive Functions

Here is a diagram relating subsystems to each other and to key external interfaces.



WSDS Input

The major required input data for the WSDC is

- Raw images from the four detectors as compressed pixel data in CCSDS source packets (ref to detector and hrdp ICDs?). Provided by the HRDP at White Sands.
- Ancillary housekeeping telemetry as comma-separated value files. Telemetry values in world (engineering) units (ref. to WSDC/MOS ICD). Provided by MOS at JPL.
- The mission plan as expected scan directions and frame counts. Provided by the SOC at UCLA.
- The sequence of events on orbit, as recorded in a machine-readable version of the SOE file from MOS.
- Ground and on-orbit calibration values, tables, and images for flats, darks, sky offsets, pixel non-linearity, spatial distortion, etc.

WSDS Output

The ultimate output of the WSDS is the released data products: the point source catalog, image atlas, level-0 frame archive (and possibly level-1b frame archive), and TBD asteroid data. In addition many intermediate products will be generated: the frame and coadd working source databases, transient intermediate images, meta-data, instrument telemetry, QA data and evaluations, etc.

An Explanatory Supplement will be distributed. This supplement describes each released data product.

Subsystem Descriptions

Executive Functions (EXEC)

Purpose
: Provide high-level services for controlling and coordinating processing jobs and the information they need as input and produced as output.

Execution Wappers (WRAP)

Purpose
: Command line parameter handling, text output control, file transformations.

Requires
: Parameter and I/O file specifications

Provides
: Uniform interface for job execution

Description
: Command execution wrappers are small executables which themselves call the application which performs the desired function. Fundamentally, the job of the wrapper is to move high-level administrative tasks out of the underlying application so it can be more focussed and easier to write. The wrapper provides a uniform interface to the outside world through use of a single parameter passing API and a common way of handling textual output from the underlying application. Wrappers are also responsible for checking for successful completion of the executed application. Wrappers may well provide other services, such as: moving, renaming, massaging or otherwise preparing input or output files; checking for completeness and integrity of prerequisites; setting parameter defaults by examination of context; employing job administration functions to execute many instances of the application concurrently; other services necessary to prepare for execution of the application and cleaning up after it.

Job Administration (JOBS)

Purpose
: Control submission, monitoring, termination, and disposition of processing jobs.

Requires
: Job specifications

Provides
: Interface for job control

Description
: The jobs administration functions provide convenient access to users and programmers to the non-interactive processing jobs which implement the WSDS system. Job administration functions include:

  - Controlling the submission of jobs, including control of the concurrent execution of many related jobs
  - Deleting (terminating) running jobs, possibly in large numbers at one go
  - Monitoring their progress and key output
  - Providing information about the them (job ID, name, command line, output location, status, errors or warnings, resources used, etc.)
  - Determining their final disposition (success or failure)
  - Locate output data for a particular frame or coadd after the job has completed

  These capabilities will be provided through API's, command line and GUI interfaces. The ability to perform these functions on jobs scattered across many machines is crucial.

Ingest Pipeline Executive (INGESTX)

Purpose
: Organize INGEST functional elements into a single executable to produce level-0 frames and associated meta-data, and to archive the raw telemetry.

Requires
: HRDP telemetry, telemetry manifest, ancillary data, parameters

Provides
: Level-0 frames as FITS images and associated meta-data in FITS headers

Description
: The ingest executive manages the following ingest functions for each delivery from MOS or the HRDP:

- Detection of incoming telemetry from the HRDP and the completion of the transfer
- Detection and reading of the telemetry manifest associated with the telemetry transfer
- Deconvolution and decompression (if necessary) of science frames from the telemetry and reading and storing of related meta-data from the source packet headers (e.g. the VTC)
- Updating the frame index and generation of a delivery report
- Determination from the frame index of which frames are needed and archiving those frames
- Receiving, indexing and archiving ancillary data delivered from MOS
- Correlation of the ancillary data to the level-0 frames and adding to the frame meta-data from the ancillary data (e.g. ADCS-generated frame position and orientation, exposure start/stop times, temperatures, etc.)
- Starting the quicklook pipeline (a variant of the scan pipeline) on a selection of frames from the delivery
- Determining when a complete set of frames for a scan are are available and starting the scan pipeline on these frames

Multi-frame Pipeline Executive (MFRAMEX)

Purpose
: Organize PIPELINE functional elements into a single executable to produce coadds and associated catalogs and meta-data.

Requires
: Output from FRAMEPIPE, coadd specification, parameters

Provides
: A single coadd and associated catalogs, meta-data, and QA output.

Description
: The coadd pipeline executive manages the following pipeline functions:

- Locate level-0 frames covering a given coadd geometric specification
- Read or regenerate level-1b frame data
- Identify outliers using temporal stacking in interpolated space
- Match backgrounds of level-1b frames
- Project, interpolate, and resample level-1b pixels into coadd image
- Extract and band merge sources and archive extractions to the coadd working database
- Write and archive QA data and meta-data

Scan Pipeline Executive (SCANX)

Purpose
: Organize pipeline functional elements into a single executable which executes the frame pipelines on all available frames for one scan in a highly concurrent manner

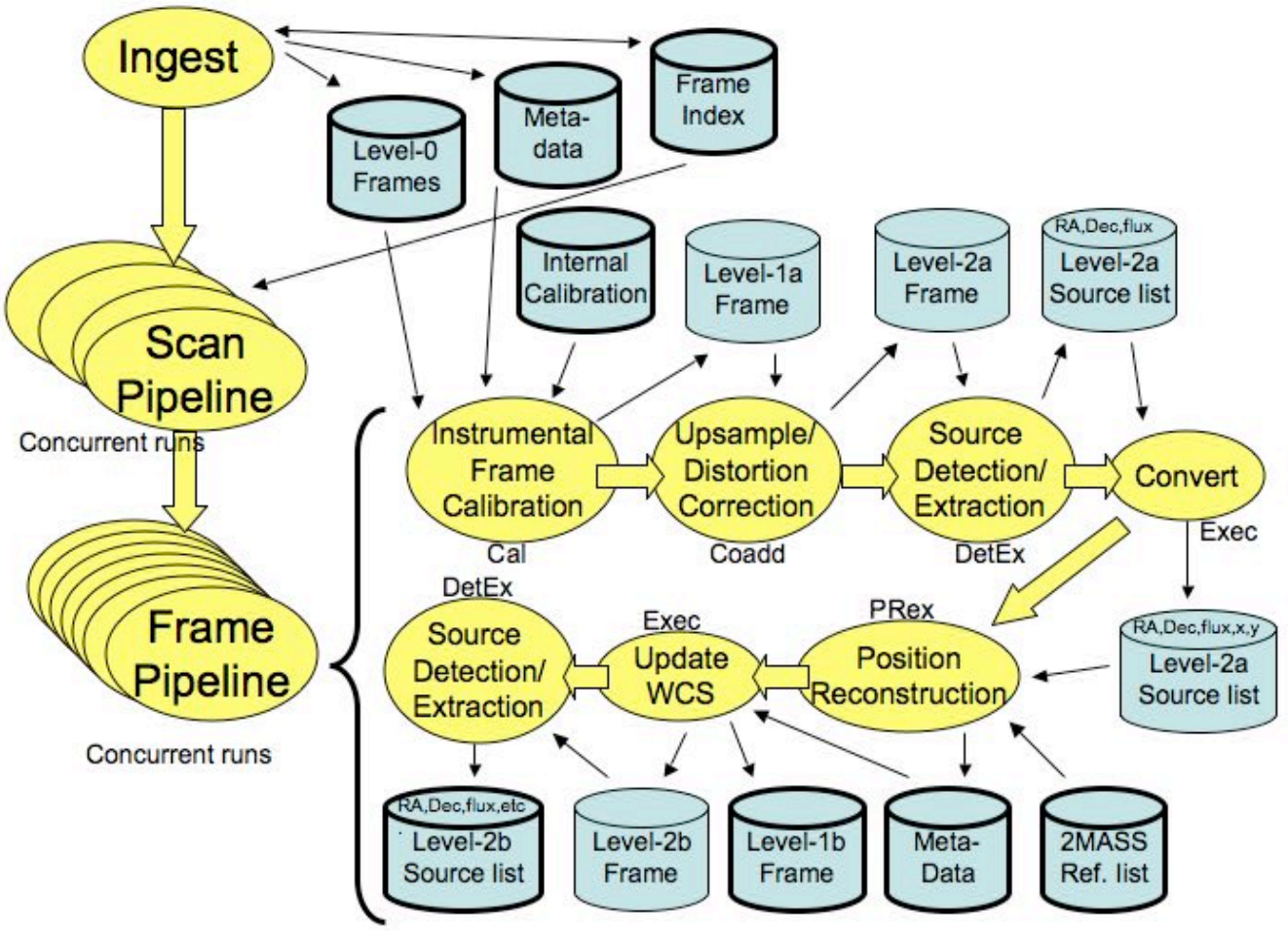
Requires
: Frame list, frame index, parameters

Provides
: Frame pipeline output

Description
: The scan pipeline is a thin(ish) wrapper around the frame pipeline which manages the concurrent execution of a given set of frames, usually the frames comprising a complete scan. The frame pipelines are started in a way that a large number can be run concurrently, potentially on many different machines. Assigning frames to machines is done through the frame index, which indicates which machines have which frames on local disk, in a manner which optimizes data locality. Internally executed frame pipelines are monitored for success or failure and this and other information is made available to operators or other users through JOBS functions.

Single Frame Pipeline Executive (SFRAMEX)

Here is a diagram showing the major subsystems/modules and data interfaces involved in the frame pipeline.



Purpose
: Organize pipeline functional elements into a single executable which produces level-1 pixel data, and associated catalogs and meta-data

Requires
: Raw frame and ancillary housekeeping data, calibartion, frame manifest, parameter files

Provides
: Meta-data for reproducing level-1 pixel data, level-1 frames, source catalogs, QA output

Description
: The frame pipeline executive manages the pipeline functions for a level-0 frame:

- Determine available resources, especially disk space, and remove level-1 frames as needed (oldest first) to make room for processing
- Performing relative instrumental calibration of the frame to level-1a
- Extracting and possibly merging sources for position reconstruction
- Performing position reconstruction, i.e. refining the frame's position and orientation and determining any scale corrections and recording this information as meta-data
- Build level-1b frames by adding a relative photometric zero point (in magnitudes), and refined astrometric information to FITS headers
- Build a level-2 frame with distortion-corrected pixels
- Performing source extraction and characterization on the level-2 frames and archiving the source list to the frame working source database

- Band merge the level-2 extractions
- Archive key frame meta-data
- Possibly asynchronously match working database sources to asteroids

Frame meta-data, source catalogs, level-1 frames, etc. will be kept in a directory which is unique to each processing attempt on each frame. Data may also be archived in IRSA-accessible tables. Level-1 frames will eventually be deleted. It will be possible to rerun the frame pipeline in a manner which reuses the extant level-1 frames, if they are still present, or which uses meta-data from a previous run to accelerate the processing, and/or to suppress certain processing steps as desired.

Ingest Functions (INGEST)

Delivery Handling (DELIV)

Purpose
: Detect incoming HRDP data and frame manifest, ancillary data delivery, detect delivery completion

Requires
: Incoming data directory location, frame manifest

Provides
: Telemetry or other transferred in a staging area

Description
: One or more servers will be dedicated to ingest of HRDP image data and MOS-generated ancillary data. These servers will have read/write access to a receiving area where external institutions (White Sands, JPL, UCLA, others?) will place data according to TBD protocol. DELIV will detect transfer completion and move the data to remote staging areas where further processing will commence. DELIV may initiate the downstream processing itself or it may begin asynchronously.

Decommutation and Decompression (DECOM)

Purpose
: Read pixel data from source packets, write level-0 frames as standard FITS files with meta-data from packet headers, especially the vehicle time code (VTC).

Requires
: Raw source packets from HRDP

Provides
: Level-0 frames with meta-data, frame completeness report, updated frame index

Description
: DECOM reads raw HRDP output and constructs raw images (compressed or uncompressed) from the source packets and gather meta-data from the packet headers (e.g. the VTC). If necessary the raw images are decompressed. The images are stored as BITPIX=-32 FITS files with meta-data in the headers. Band 4 images are upsampled by a factor of 2 through simple pixel replication. The resulting frames constitute the archival level-0 frame data. The level-0 FITS files may be stored compressed. In constructing the level-0 frames, incomplete frames or frames that could not be decompressed are not saved. A frame index database is updated recording the frame's VTC, and delivery date and time, as well as the name and location of the resulting FITS output file. A frame completeness report (format contents TBD) will be generated for each delivery.

Meta-data Correlation (INMETA)

Purpose
: Correlate level-0 frames to ancillary housekeeping data (SPICE kernel, etc.)

Requires
: Level-0 frames with VTC in meta-data, ancillary housekeeping telemetry for time covering frame exposure

Provides
: Selected ancillary meta-data in FITS frame image headers, updated frame index, meta-data file

Description
: Newly delivered level-0 frames will be correlated by time (the VTC) to the ancillary data derived asynchronously from payload telemetry delivered by MOS. If no ancillary data, especially the SPICE kernel, cannot be matched to the level-0 frames, INMETA will wait for a predetermined amount of time for the data to be ingested. If no ancillary data is matched even after the wait period is over, the level-0 frames that cannot be matched cannot be processed. The headers of all matched level-0 frames are updated to include ADCS position and orientation information from the decoded SPICE kernel, as well as TBD other ancillary data. The frame index is updated to include key meta-data for each matched frame. In addition, a meta-data file, which may, for example, be a FITS table file, be written in the same location as the matching level-0 frame to hold all pertinent meta-data.

Pipeline Functions (PIPELINE)

Image and Source Calibration (CAL)

Purpose
: Take level-0 frames and produce level-1 frames (i.e. with relative instrumental calibrations applied)

Requires
: Level-0 frames and meta-data, calibration

Provides
: Level-1a frames, QA data

Description
: The following paragraphs describe some of the various calibration functions. For an in-depth overview of the instrumental calibration steps and the input calibrations needed, see http://web.ipac.caltech.edu/staff/fmasci/home/wise/SingleOrbit_Cal.html and http://web.ipac.caltech.edu/staff/fmasci/home/wise/ScanPL_instrumental_cal.pdf

- Electronic Bias/Offset Correction

  Use the reference (aka border, aka "over-scan") pixels to apply a pixel offset correction, as: subtract output-by-output reference pixel column offsets, then compute a running average of 7 reference pixel rows and subtract that prior to computing noise/dark current/etc.

- Dark Subtraction

  Subtract dark frames to eliminate dark current. Dark frames will only be available from ground calibration.

- Droop Correction

  The droop correction is a global offset correction based on the total illumination on the detector. For the purposes of droop computation, the exact pre-calibration pixel values will be used. The contribution of saturated pixels will need to be estimated for a robust determination of droop.

- Pixel Response Correction

  Divide frames by a response map to correct for variations in pixel-to-pixel response. Response maps will be made during ground calibration and updated in flight using "super-flats".

- Non-linearity Correction

  Make a linearity correction to pixel values. This correction depends on the pixel value (as a fraction of full well). Non-linearity corrections will be determined on the ground and can be updated in flight.

- Saturation Flagging

  Raw pixel values will be examined for one of the pixel values indicating SUR value was saturated. The presence of saturation will cause a pixel to be flagged in a mask (though the value is preserved). No calibration information is required for saturation flagging.

- Cosmic ray hits

  Flag pixels contaminated by bright cosmic rays. Method unknown at this time.

- Detector Artifact Handling

  Known and anticipated artifacts at this time are:

  - Pixel persistence, AKA latent images
  - Diffraction spikes
  - Internal reflections
  - Out of field scattered light

- Relative Photometric Calibration

  Identify photometric standard stars in level-1b frames of a scan (after DETEX and PREX steps have run), derive relative photometric zero-points, and insert into frame headers

Source Detection and Extraction (DETEX)

Purpose
: Detect and extract sources and source characteristics from a frame or coadd image

Requires
: COADD output: Level-2 frames or level-3 coadds and associated products

Provides
: Source catalog and meta-data, QA data

Description
: DETEX performs detection ("segmentation" in image analysis terms) and extraction (source characterization), including astrometry and photometry, in a number of different contexts. Each context may conceivably use a different detection or characterization engine from other contexts, if necessary.
The different contexts identified thus far are:

- From level-2a (upsampled frames with distortion-corrected pixels prior to WCS refinement) frames for PREX, optimizing for speed and positional accuracy; the level-2 frames have
- From level-2b frames (level-2a frames with updated WCS) for solar system object matching and QA trending and other epoch-dependent science
- From coadds for addition to the coadd working database and in the final catalog

The complete list of required source characteristics is TBD, but includes at least

- position in sky (possibly more than one coordinate system) and image coordinates,
- flux, possibly in a few different states of calibration,
- position and flux errors,
- source shape information (e.g. PSF FWHM estimate, 2nd moments),
- background and noise metrics,
- and track back information on the source frames.

Detection (segmentation) and extraction will occur in different applications. The detection step is called MDET and is a multi-band detector. The photometry/astrometry characterization step is called MPHOT. MPHOT is in turn divided into two applications, one doing aperture photometry/astrometry, and one doing profile fitting photometry/astrometry.

Band Merging (MERGE)

Purpose
: To positionally associate extractions for a given image from multiple bands to provide enhanced extraction data

Requires
: DETEX output for a level-2 frame, or a coadd, from one or more bands

Provides
: A new band merged source list, with additional or updated characterization data, QA data

Description
: Extractions for all bands present for a given frame or coadd are read and positionally associated. For each source, color information as well as enhanced position estimates and reliability are generated and recorded, along with the original single-band data, in a new band merge source file. For uniformity's sake, MERGE needs to be able to generate a band merge source file even if there's only one band.

This module's output may be partially or fully accomplished in the DETEX module since DETEX will do multi-band detection.

Pointing Reconstruction (PREX)

Purpose
: Determine refined position, orientation, and distortion corrections for level-0 frames

Requires
: Level-1a frames for at least TBD bands, DETEX output, point source astrometric reference catalog (from the 2MASS PSC)

Provides
Level-1b frames, with corrections for frame position and orientation and distortion
Description
Using matches between all extractions from level-1a frames for all available bands--at least including bands TBD, and TBD--to the astrometric reference catalog, refine our knowledge of the frame's position and orientation. Write the needed corrections as meta-data, and apply the frame WCS position center and orientation corrections to the level-1a frames to create level-1b frames.

Solar System Object Matching (SSOFIND)

Purpose
Match level-2b frame extracted sources to known solar system objects (SSO's)
Requires
Level-2b frame working source database, known SSO ephemerides
Provides
TBD known SSO match database
Description
Reading the level-2b frame working source database to get candidate source positions and observation epochs, match to known SSO's using ephemerides of known objects.

Resampling and Coaddition (COADD)

Purpose
Produce coadded images from level-1 pixels
Requires
Level-0 images and other frame pipeline products
Provides
Coadd images and related prerequisites for DETEX.
Description
COADD encompasses these functions:

- Locating frames matching spatial and temporal requirements using the frame index
- Resample/interpolate level-1b pixels onto a common grid to identify outliers using a temporal stacking method
- Gain/throughput matching of level-1b frames by rescaling pixel values to a common photometric zero-point. This will be the photometric zero point for the output coadd
- Background (offset) matching of level-1b frames to produce seamless coadds
- Reproject, resample and combine level-1b pixels into a level-3 coadd or level-2 frame
- Generate accompanying coverage and uncertainty (sigma) coadds
- Provide prerequisites needed by DETEX

Quality Analysis Functions (QA)

Purpose
Provide Quality Analysis reports on level-1 frames, coadds and catalogs
Requires
Frames or coadds, catalogs, pipeline QA output
Provides
QA reports
Description
The QA subsystem performs these functions:

- PSF and Scan-sync Analysis
- Survey Coverage Analysis
- Source Count and Confusion Analysis
- Artifact Analysis
- N/M Analysis
- QA Report Display and Distribution

Final Product Generation Functions (FPG)

Purpose
: Create exportable products from coadd images and catalogs

Requires
: Coadds, coadd catalogs, other products from the coadd pipeline

Provides
: Exportable coadd images (atlas images), exportable catalog

Description
: The FPG subsystem is a collection of utilities for aiding in the preparation of the releasable products; the image atlas and the source catalog. Much of the work of product preparation will be manual--such as composing a query to select the rows of the working database to release as the catalog product--but some activities can be aided by support utilities. Here are some examples:

  Catalog Generation

  - Interfaces to IRSA functions for easy command line source selection
  - Summary statistics generation
  - Generating compact but readable source lists

  Image Atlas Generation

  - Image filtering tools?
  - Convenient mosaic generation

Archive Functions (ARCHIVE)

Purpose
: Provide tools and APIs to access the varied archival data generated by the WSDS

Requires
: Raw or processed archival data, indices, parameters, required data specs, etc.

Provides
: Requested data location or other meta-data information, and/or the actual data, in toto or in part

Description
: WSDS archival data can be split into two general types, depending on whether it's loaded into IRSA servers or not.

  - IRSA databases

    Much WSDS data, final or intermediate, will be loaded into IRSA DBMS tables. Existing IRSA access tools and APIs can be used to read/write this data. New tools/APIs may also need to be written.

  - Other databases

    Databases exist distributed in a predetermined directory structure on data servers. ARCHIVE will provide tools and APIs to find, write and access this data, however it is distributed.

Reference Hardware Design

Here I describe one possible hardware implementation which could meet the throughput requirements (c.f. Throughput Reuirements) for the frame pipeline (FRAMEX).

The WSDC hardware system will have five major components:

A compute cluster
: Composed of up to about 30 nodes, each of which has dual quad-core x64 processors (i.e. 8 cores), >=500GB internal disk storage, and >=8GB RAM. Some nodes may be super-sized for use in coadding, which may require more memory. Notionally, 20 of these nodes are devoted to scan/frame processing.

A single gigabit network
: Interconnects all compute hardware and disk and backup storage. Possibly with multiple virtual subnets for scaling access to the compute cluster.

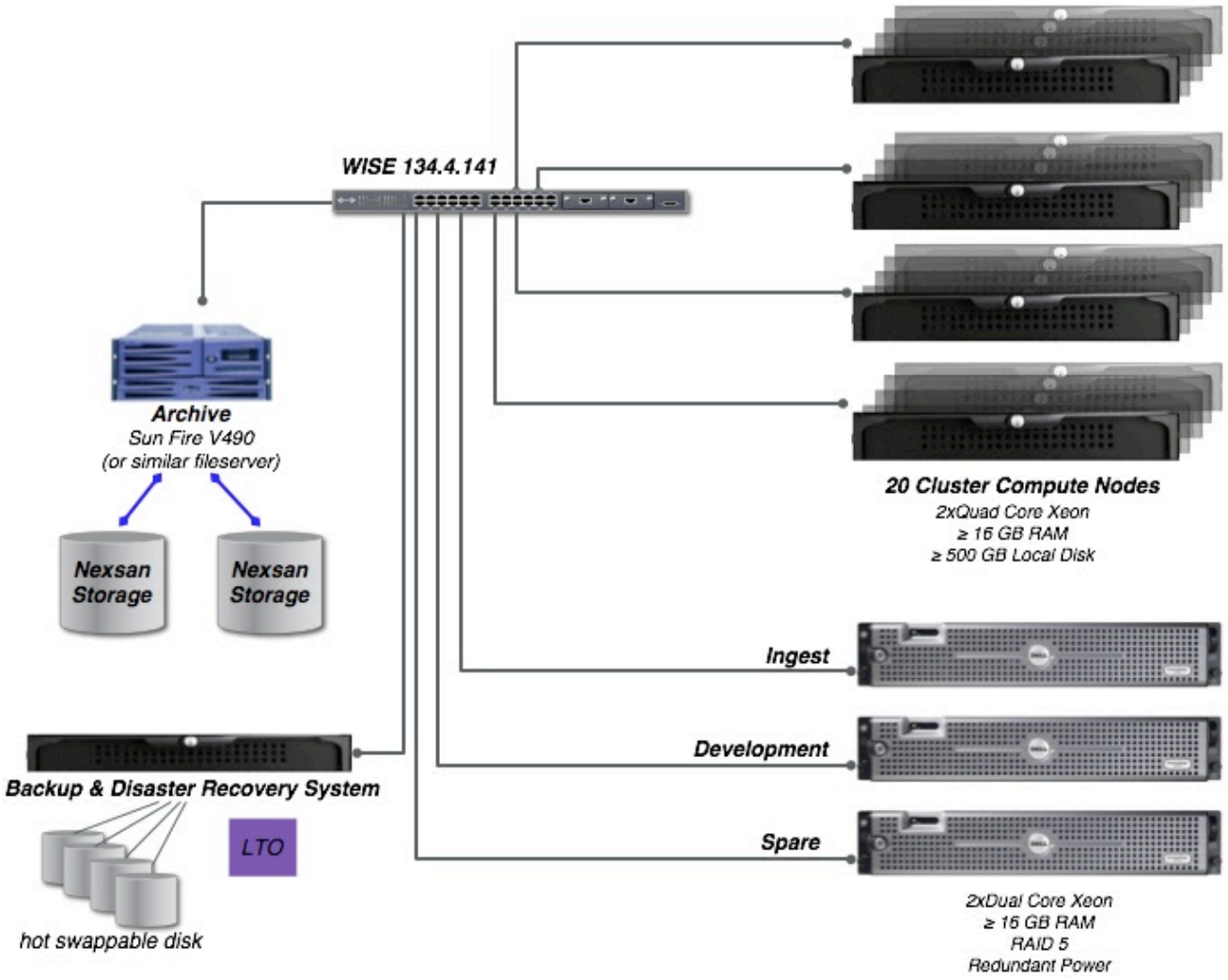
Dedicated servers

These are more robust than the compute nodes (dual power supplies, backplanes, mirrored disk, etc.). I know such servers are required for development and ingest.

Archival disk
: 50-100 TB of RAID-5 (or RAID-X?) disk and associated servers, probably using multiple devices.

A backup and offline-archive system
: Both tape and removable disk based.

Here is a diagram laying out the main features of one possible implementation of the above features.



Throughput Requirements

Assumptions

- Frame processing through pixel resampling (but not coaddition)
- Same amount of work per pixel as 2MASS
- Clock cycles mean about the same thing on 2MASS h/w as on WISE h/w
- CPU unconstrained by memory (no paging)
- A "frame" below referes to one exposure from a single band

Computations

- 2MASS run time per scan is 4047 secs

```
total   pixcal posfrm pixphot mapcor bmrg pospts +  gal + ohd
4047 =     787 +  167 +  1485 +  106 + 36 +   33 + 1397    37
```

- 2MASS pixels: 273 frames/scan * 256x256 pixels/frame = ~16Mpix/scan
- 2MASS CPU: single ~300MHz SPARC
- Benchmarks (from http://serv.apphy.fukui-u.ac.jp/~tajima/bench/result.html):

```
matvec  :   48 Mflop/s (Mega-floating point ops per sec)
intgl4  :  134 Mflop/s
runge   :  192 Mflop/s
```

- Use 100Mflop/s as an average. Implies about 3 clock cycles per flop.
- 4000s*100Mflop/s / 16 Mpix = 25kflop/pix
- WISE core: 2.7GHz
- Assume same ~3 clock cycles per flop => ~1Gflops
- 25kflop/pix * 1Mpix/bandframe = 25Gflop/bandframe
- 25Gflop/bandframe / 1Gflop/s = 25s/bandframe * 4 bands = ~2 min.s/frameset/core

Discussion

If we make the totally unjustified assumption that we'll run at about 50% CPU time (i.e. 50% time doing I/O and paging, etc.), then we get about 4 minutes per frameset.

I've previously computed that we can afford about 30 dual quad-core (8 core) machines reasonably configured (500GB disk, 8GB RAM). Assume we use 20 of them for scan/frame processing. With 160 cores we can process (assuming 1 frameset per core) 160 frames in ~240 seconds, or about 1.5 s/framesets. Add in extra time for more complex coaddition, analysis, and 100% pad, and we have roughly 4 s/framesets.

At roughly 7000 framesets/day, this will require ~28000 seconds, or about 8 hours. This leaves time for one rerun in parallel, and 8 hours of downtime for backup, maintenance, and other activities.

Data Directory Layout

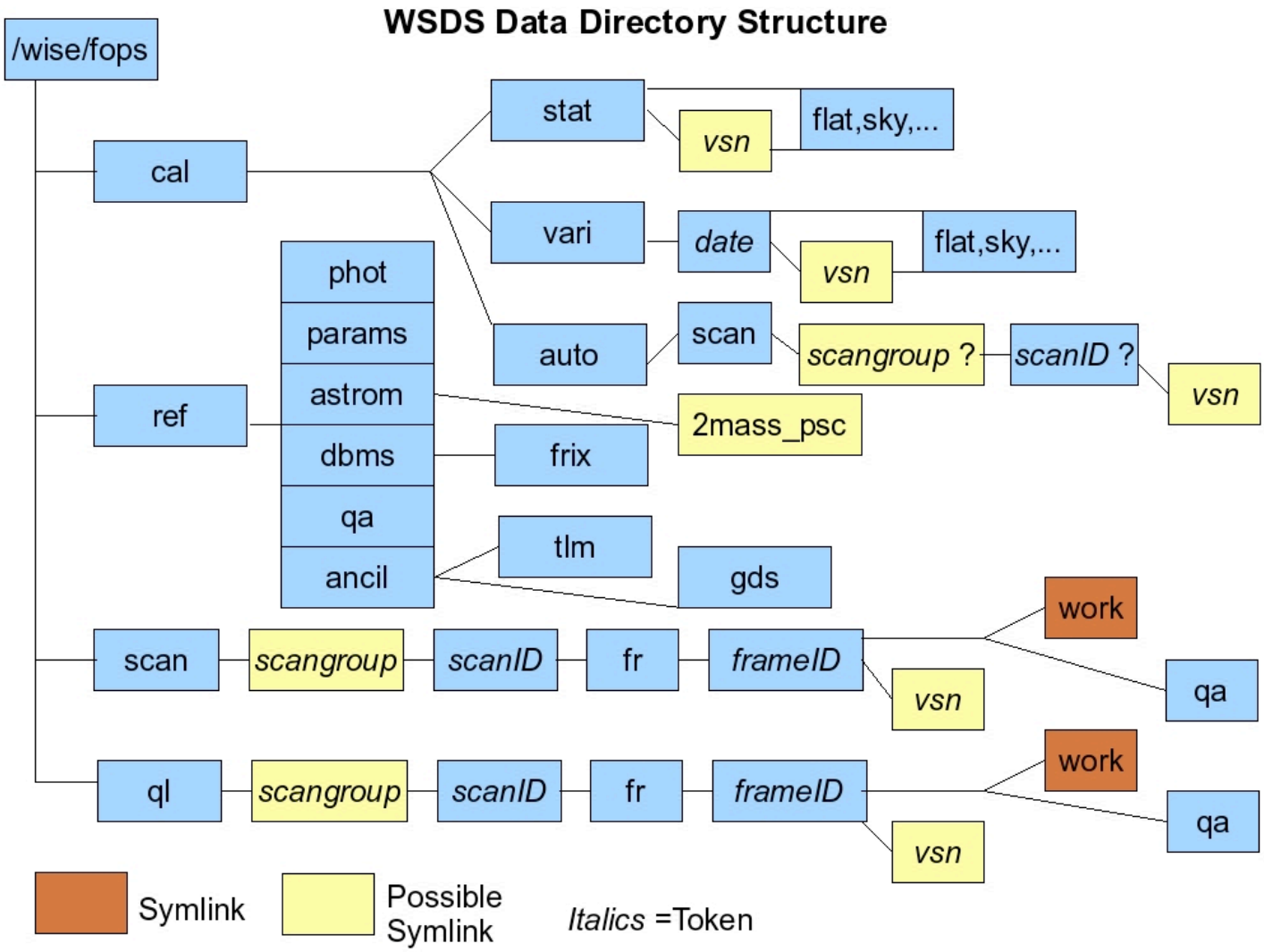
Introduction

This section describes the location on WSDS networked storage of the various data files needed by the WSDS processing system.

The layout of many kinds of data on our distributed processing system must meet these requirements:

- Data must be in predictable locations derivable from more fundamental data (e.g. a frame ID).
- Sometimes data must be viewable across the network, other times it need only be visible to the currently-running pipeline.
- Our storage must be extensible without having to move large amounts of data around.
- Some data, e.g. calibration data, may need to be segregated by date/time (or, equivalently, scan ID and/or frame ID)
- Frame data (archival and otherwise), coadd data, and their derived products will be spread out over a combination of network-accessible disk archives and local storage.
- Reference data (quasi-static calibration data, reference catalogs, etc.) must be findable by all processes.
- We must clearly segregate ground data, flight data, and sim data at a high level.
- Data granularity must be such that a pipeline (or subsystem) will generally find all it needs in one or a few places, but a given directory must not have so much data in it that the file count becomes excessive.

Here is a diagram summarizing most of what is described.



The Use of Symlinks

Symlinks are a handy way to overlay a single, regular directory structure on top of data with may be physically scattered among many different physical devices. If one chooses the right granularity for one's directories (providing sufficient opportunity for indirection) one can also expand available space without displacing other data. For these reasons much of our data access will be through symlinks. In principle, one could move a great deal of data to a completely different location, and as long as the overlying symlinks were adjusted, nothing would appear to have changed for programs or code needing access to the data.

But symlinks have a dark side too. The very fact they obscure the underlying physical layout of the data can sometimes engender confusion (you can easily go down through symlinks, but going back up ain't so easy). Also, some standard utilities (such as 'find') will not work so easily through symlinks. Also, if one wishes to record for posterity where something is/was located, special care might be required to record not just the overlying symlink-based path, but the true path as well. Symlinks are allowed to point to non-existent targets, which is a great and useful thing, but can also be confusing and sometimes frustrating.

In the sections below, it is possible that any given path element may actually be a symlink, though most symlinks will be limited to those documented in the next section. To a coder this means one has to be careful using relative paths. E.g. if one navigates to directory /a/b/c/d (i.e. it is your CWD) and plans on getting to directory /a/b/c/e by cd'ing to ../e, you will fail if d is in fact a symlink pointing to a disjoint directory. We will try to arrange things so that one is unlikely to fall into such a trap.

Overall, symlinks are too handy not to use, but we will try to limit their use in ways that will mitigate problems.

Known Symlinks

work
: The **work** directories in the frame pipeline output directories will be symlinks to node-local storage. All volatile images created as part of the frame pipeline will be stored here to avoid using the network. Note that this scheme has complicating repercussions:

  - Since the work directory must be local to the node running the frame pipeline, subsequent reruns of the same version must delete the old work directory and create a new one on the currently-running node.
  - All nodes must see all other nodes' local storage using the same name everywhere. (This appears to be tricky due to surprising interactions with NFS and the automounter.)

latest version
: The latest, highest numbered **v1, v2, ...** subdirectories (labeled **vsn** in the diagram) of the frame pipeline output directories will be a symlink to '.'. The other version directories will be real subdirectories of the output directory. Various calibration directories will also utilize this scheme.

scangroup
: In order to be able to load balance frame piepline output across disks and servers, we may break the scans into groups which will be pointed to by symlinks. This is why the **scangroup** path element exists.

Scan and Frame IDs and Frame File Basenames

Scan IDs are the 5 digit representation of the sequential orbit number over the mission with an 'a' or 'b' appended, e.g. '07462a'. Frame IDs are the 3 decimal digit representation of the frame's sequential number within the scan. E.g. '215'. Frame set basenames are constructed of the scan and frame IDs thusly: '07462a215'.

The Data Root

All data, products, etc. received for or generated by pipeline and associated applications will be accessed through

- /wise/fops for actual flight operations.
- /wise/tops for pre- or post- flight test operations.

While we are still far from flight, we can use 'fops'. We just have to remember to clear it out (copy to 'tops'?) when we get close to launch. When we are preserving the sanctity of 'fops' (near or after launch) we can conduct tests etc. in 'tops'.

Below I will always use 'fops'.

Reference Data

Reference data is anything generated somewhere else which is fairly static. Astrometric or photometric catalogs is one example. Obvious subdirectories might be

- /wise/fops/ref/astrom for astrometric reference catalogs, each i its own sub-directory.
- /wise/fops/ref/photom for photometric reference catalogs, each i its own sub-directory.
- /wise/fops/ref/params for parameters. There will be a 'wrap' sub-directory for standard perl wrapper parameters. Other applications/functions may require their own directories. These are high-level parameter files and will be overridden by the contents of parameters in /wise/base/deliv/[ops,dev,...]/params and by any run-specific parameters.
- /wise/fops/ref/frix for frame index DB files (i.e. related files which describe frame data locations and status).
- /wise/fops/ref/ancil for frame-related ancillary data, e.g. H/K telemetry. This is a refined set of DBMS (SQLite?) tables to be used by ingest in constructing the level-0 frames, not the raw ancillary data archive, which may be very little more than a date-tagged copy of what was sent from MOS).
- /wise/fops/ref/qa for global QA data. Placeholder. Not sure what really goes here.

Calibration Data

Calibration data is divided into sections based on how often the data are expected to be updated.

- /wise/fops/cal/stat for static calibration data (ground data or IOC data generated only once. Cal. data may be versioned within the static directory. The active version is always in the main directory; other versions are in subdirectories named "vd [.d]", e.g. v1, v2, v2.2, etc. E.g. /wise/fops/cal/stat would contain the active static calibration files, and /wise/fops/cal/stat/v3 points to a particular, inactive version.
- /wise/fops/cal/vari for data which will periodically be adjusted throughout the mission, but is not generated autonomously by the pipeline. Subdirectories will use dates or date ranges encoded as yyddd or yyddd-yyddd, Below the date range, the data will be versionedas above.
- /wise/fops/cal/auto for data generated autonomously by the pipeline. Subdirectories will scan groups and scanIDs. Within the scanID data will be versioned as above.

Pipeline Input/Output Directories

The two main pipelines are the scan/frame pipeline, and the coadd pipeline. I'll be mainly concerned with the scan/frame pipeline.

The scan/frame pipeline is run in two different modes after ingest: the quicklook mode, and the full mode. These two modes will run in completely separate hierarchies:

- /wise/fops/ql: Quicklook output.
- /wise/fops/scan; Full scan/frame pipeline output.

Scan and Frame IDs in the Directory Structure

I will just refer to /wise/fops/scans below; /wise/fops/ql parallels its structure.

The next level down segregates the frame data by scan IDs and frame IDs. But because there will be so many scans over the life of the mission (easily > 10,000), for convenience the scans should be subdivided into groups of no more than a few hundred scans per subdirectory. Accordingly, we divide the data up based on the first three digits of the scan ID. Scans within one scan group directory use their full 5 digit+letter ID:

/wise/fops/scans/001/00142b

Below the scan, the frame ID is the sequential number within that scan, a 3 digit number, within the scan's 'fr' subdirecotry:

/wise/fops/scans/001/00142b/fr/185

Each frame may have multiple processing attempts, each represented by a version number of the foprm vd[.d] (e.g. v1, or v2, or v2.2). The latest run will always be in the main directory. So a full path name to scan/frame pipeline output would be /wise/fops/scans/001/00142b/fr/185 and that for a particular old version would be /wise/fops/scans/001/00142b/fr/185/v3. The current, active version is pointed to by a version symlink, i.e. if the active version is v4, the v4 symlink would point to '.'.

Frame Output Directory

Each frame output directory will contain most relevant data athe the top level, but will have two supplementary subdirectories, 'qa', and 'work':

/wise/fops/scans/001/00142b/fr/185/qa /wise/fops/scans/001/00142b/fr/185/work

'qa' holds qa-specific output, and 'work' will point to local disk space and the contents are to be considered volatile.

Ingest Directories

The ingest process which reads and uncompresses the pushed HRP data from White Sands and creates level-0 data frames sets will write to its own hierarchy:

- /wise/fops/ingest/tlm; ingest meta-data divided by delivery IDs. Delivery IDs are TBD but will probably be a shortened version of the date/time-based files names used for the telemetry files. E.g. YYMMDD_HHMMSS. Since there will be upwards of 700 of these over the course of the mission, these will be divided by month. Thus each delivery will have a direcotry of the form YYMM/YYMMDD_HHMMSS. E.g. '/wise/fops/ingest/tlm/0912/091225_011256'.
- /wise/fops/ingest/meta; ingest meta data, such as a delivery index for frame and h/k (see below) data.

Archives

At the moment, four WSDC archives are contemplated:

- /wise/fops/arch/l0: Level-0 frame archive. Follows the naming scheme for scan/frame pipeline output.
- /wise/fops/arch/ancil: Raw ancillary data archive. Structure TBD.
- /wise/fops/arch/fr: Level-1b frame archive. Follows the naming scheme for scan/frame pipeline output.
- /wise/fops/arch/coadd: Coadd archive. This is for the pre-planned, layed out, official, proto-atlas coadds, not the on-the-fly, on-demand coadds.

WSDC Operations Concept

Here is a brief accounting of a day-in-the-life at the WSDC during operations. This is not an exhaustive treatment of WSDC resource utilization, but shows just one example how resources for one random day might go. Exact timings are imaginary, but represent reasonable guesses based on what we know.

Assumptions

- 4 deliveries a day, 5 hours apart;
- 8 scans completed per delivery
- 3 FTE QA analysts
- 2 FTE operations personnel
- 5/40 staffing
- A delivery of science data requires 3.5 hours from arrival at the HRP to complete transmission to the WSDC
- Ancillary data arrives 1 hour before the science data and takes 10 minute to transfer and 20 minutes to archive
- Ingest of science data requires 2 hours
- Quicklook requires 30 minutes to complete
- ScanPipe requires 2 hours to complete for the ~8 new scans completed in a single delivery
- Operations staff require
  - 15 minutes to log and archive to disk and tape newly ingested telemetry (doesn't count actual recording time)
  - 15 minutes to monitor, examine and report on a ScanPipe run
  - Thus each delivery requires 30 minutes of ops personnel time
- QA analysis requires
  - 5 minutes for a normal scan, and 30 minutes for troubles scan
  - 1 scan per delivery (1/8) will be troubled
  - Thus each delivery requires about 1 hour for QA analysis
- It is a Monday, so 8 weekend deliveries must be handled
- 2 overnight deliveries must be handled

WSDC Operations for a Sample Monday

Time	Ops Personnel Activity		QA Personnel Activity			Automated Activity
	1	2	1	2	3	

00:00	-	-	-	-	-	Receipt of ancillary data delivery 1 begins.
00:10	-	-	-	-	-	Ancillary data delivery 1 ends. Data are archiving begins.
00:30	-	-	-	-	-	Ancillary data delivery 1 archiving complete.
01:00	-	-	-	-	-	Receipt of science data delivery 1 begins.
04:30	-	-	-	-	-	Delivery 1 receipt complete. Ingest begins.
05:00	-	-	-	-	-	Receipt of ancillary data delivery 2 begins.
05:10	-	-	-	-	-	Ancillary data delivery 2 ends. Data are archiving begins.
05:30	-	-	-	-	-	Ancillary data delivery 2 archiving complete.
06:00	-	-	-	-	-	Receipt of science data delivery 2 begins.
06:30	-	-	-	-	-	Delivery 1 ingest complete. Quicklook initiated on selected frames. ScanPipe initiated on newly completed scans.
07:00	-	-	-	-	-	Delivery 1 Quicklook completed.
07:30	-	-	-	-	-	
08:00	-	-	-	-	-	
08:30	-	-	-	-	-	Delivery 1 new scan ScanPipe processing completed.
09:00	Weekend runs	Overnight runs	Weekend runs	Weekend runs	Overnight runs	
09:30	"	"	"	"	"	Delivery 2 receipt complete. Ingest begins.
10:00	"	-	"	"	"	Receipt of ancillary data delivery 3 begins.
10:10	"	-	"	"	"	Ancillary data delivery 3 ends. Data are archiving begins.
10:30	"	Delivery 1	"	"	"	Ancillary data delivery 3 archiving complete.
11:00	Status Review	Status Review	Status Review	Status Review	Status Review	Receipt of science data delivery 3 begins.
11:30	Weekend runs	-	-	-	Delivery 1	Delivery 2 ingest complete. Quicklook initiated on selected frames. ScanPipe initiated on newly completed scans.
12:00	-	-	-	-	"	Delivery 2 Quicklook completed.
12:30	-	-	-	-	-	
13:00	"	-	Coadds	Trending		
13:30	"	Delivery 2	"	"	Delivery 2	Delivery 2 new scan ScanPipe processing completed.
14:30	"	-	"	"	"	Delivery 3 receipt complete. Ingest begins.
15:00	Run Coadds	Backups	"	"	-	Receipt of ancillary data delivery 4 begins.
15:10	Monitor Coadds	"	"	"	-	Ancillary data delivery 4 ends. Data are archiving begins.
15:30	"	"	"	"	-	Ancillary data delivery 4 archiving complete.
16:00	"	"	-	-	-	Receipt of science data delivery 4 begins.
16:30	-	"	-	-	-	Delivery 3 ingest complete. Quicklook initiated on selected frames. ScanPipe initiated on newly completed scans.
17:00	Status Review	Status Review	Status Review	Status Review	Status Review	Delivery 3 Quicklook completed.
18:00	-	-	-	-	-	
18:30	-	-	-	-	-	Delivery 3 new scan ScanPipe processing completed.
19:30	-	-	-	-	-	Delivery 4 receipt complete. Ingest begins.

21:30	-	-	-	-	-	Delivery 4 ingest complete. Quicklook initiated on selected frames. ScanPipe initiated on newly completed scans.
22:00	-	-	-	-	-	Delivery 4 Quicklook completed.
23:30	-	-	-	-	-	Delivery 4 new scan ScanPipe processing completed.

Glossary

Ancillary data
: Meta-data related to processing image data derived from payload housekeeping and ancillary telemetry. Examples: the VTC, SPICE kernels (derived from ADCS telemetry), instrument temperatures, etc.

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

Atlas
: A collection of specially generated coadds which comprise the releasable WISE image product.

Bandframe
: Image for a single band for one exposure.

Catalog
: A selection of entries and column values (possibly refined) from a working source database constructed to meet certain completeness, reliability, astrometric and photometric requirements.

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.

Coadd
: An image generated from the coadded pixels from one or more level-1 frames covering a defined rectangular region of sky. See also "Image, Level-3"

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

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

Frame, Level-0 frame
: Same as a raw frame but with 4-byte floating point pixels (BITPIX=-32) and band 4 replicatively upsampled to 1024x1024. This is the archival frame format.

Frame, Level-1a frame
: A level-0 frame with relative photometric calibrations applied.

Frame, Level-1b frame
: A level-1a frame with refined frame position and orientation WCS information

Frame, Level-1c frame
: Old terminology for a level-2b frame

Frame, Level-2a frame
: A level-1a frame which has been resampled, and possibly upsampled, to correct for distortion

Frame, Level-2b frame
: A level-1b frame which has been resampled, and possibly upsampled, to correct for distortion

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.

Frameset
: Images in all available bands for one exposure.

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.

ICD
: Interface Control Document

Image, Level-3
: A coadd

Internal Calibration
: See Relative Calibration

IRSA
: Infrared Science Archive. The group/system at IPAC which provides the archive and delivery service for WISE data products and other internal databases.

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

Latent image
: A false "ghost" image in frame data caused by a persistent excess current and/or a change in the response of pixels illuminated by a bright source in a prior frame. May be either a positive or negative deviation from the background.

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.

Relative Calibration
: Instrumental frame calibrations which result in all pixels having flattened, linear, zero-offset response. Pixel fluxes are not yet tied to an absolute photometric reference.

Scan
: A period of observation running from pole crossing to pole crossing. I.e. about half an orbit.

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

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."

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

Working source database (WSDB)
: A database of raw source extractions from frames (level-1) or coadds.

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.