WISE Mission Operations System CDR

WISE Science Data Center

Roc Cutri - IPAC
WISE Science Data Manager
Tim Conrow - IPAC
WSDC Lead Engineer

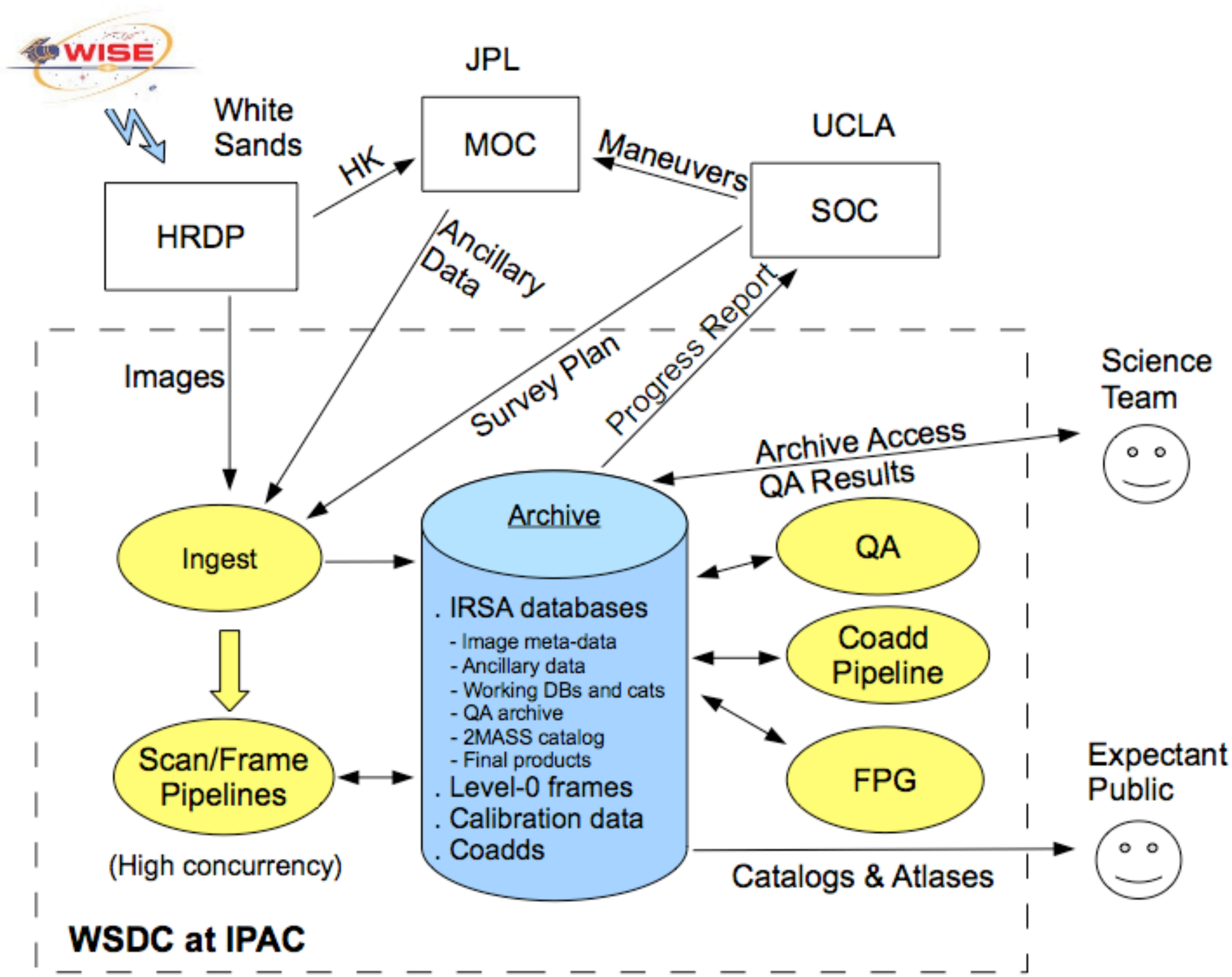
WISE Science Data System Architecture

Tim Conrow - IPAC
WSDC Lead Engineer

WSDS Architecture

- WSDS Subsystems
- Development Strategy
- Hardware Architecture
- WSDS Issues
- Road to CDR

WSDS Subsystems: **Overview**

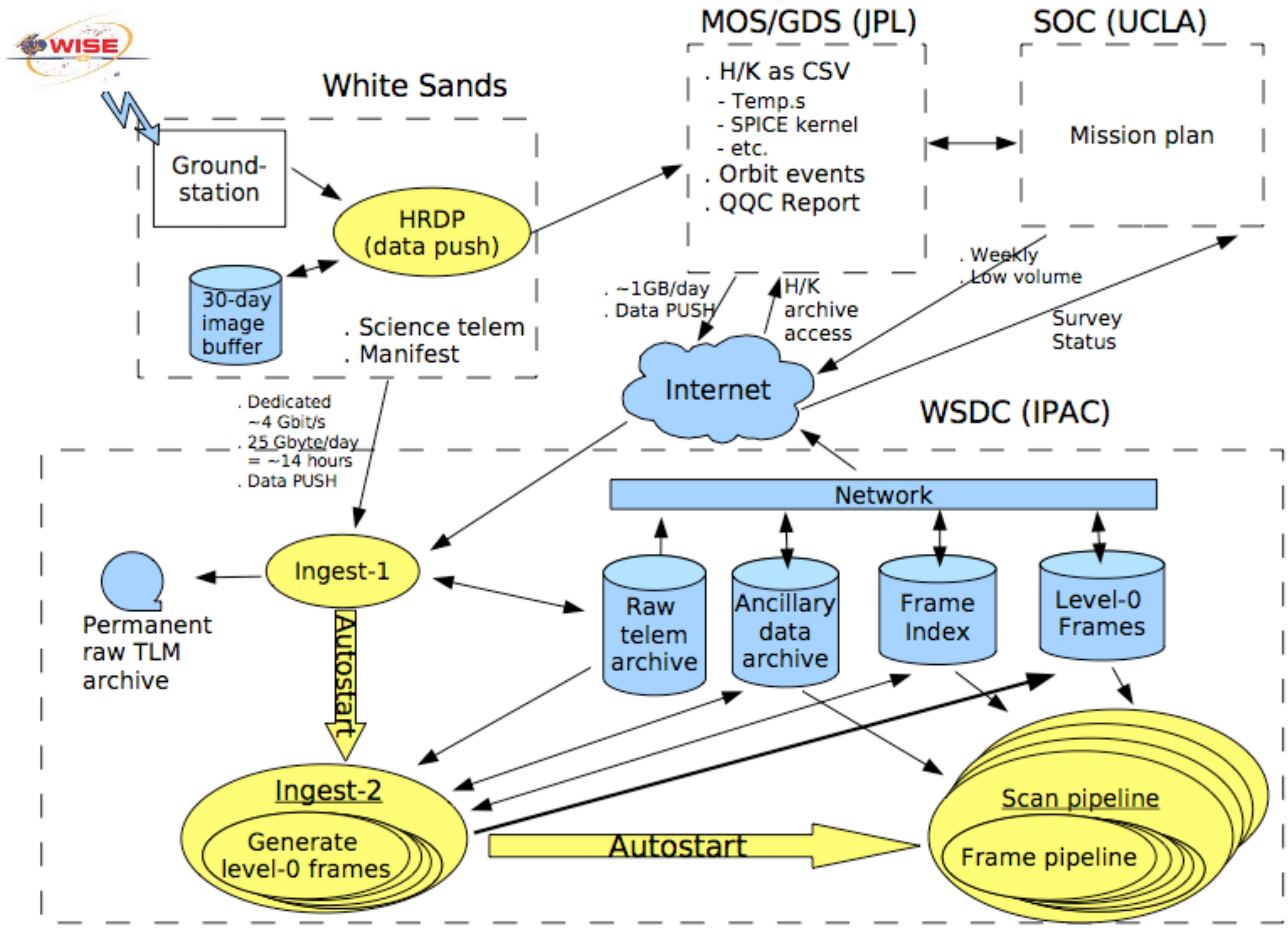


WSDS Subsystems: **Ingest**

Receive and verify high-rate science telemetry from White Sands. Decompress and assemble complete frames with ancillary data from MOS. Write level-0 image archive. Start scan pipeline.

- Receive manifest at start of transfer from White Sands
- Detect completed transfers from HRDP and verify against manifest
- Archive raw frame telemetry and ancillary data from MOS
- Decompress and assemble all images and update frame index
  - Images may come out of order from multiple deliveries, so a scan may need to be stitched together using an index of data in the image archive
- Correlate data to ancillary data and generate frame meta-data
  - WCS info, time, other (temperatures, voltages?, correlation to science plan?, …)
  - Write to image headers and stand-alone meta-data file
- Write compressed FITS files with meta-data to level-0 frame archive
- Identify frames in complete scans (pole crossing to pole crossing) and start the scan pipeline
  - Locate frames for each complete scan in the delivery. For incomplete scans, process when re-transmissions complete the scan or 72 hours (TBD) elapses
  - The quicklook pipeline is launched after each delivery without regard for missing frames

WSDS Subsystems: **Ingest Data Flow**



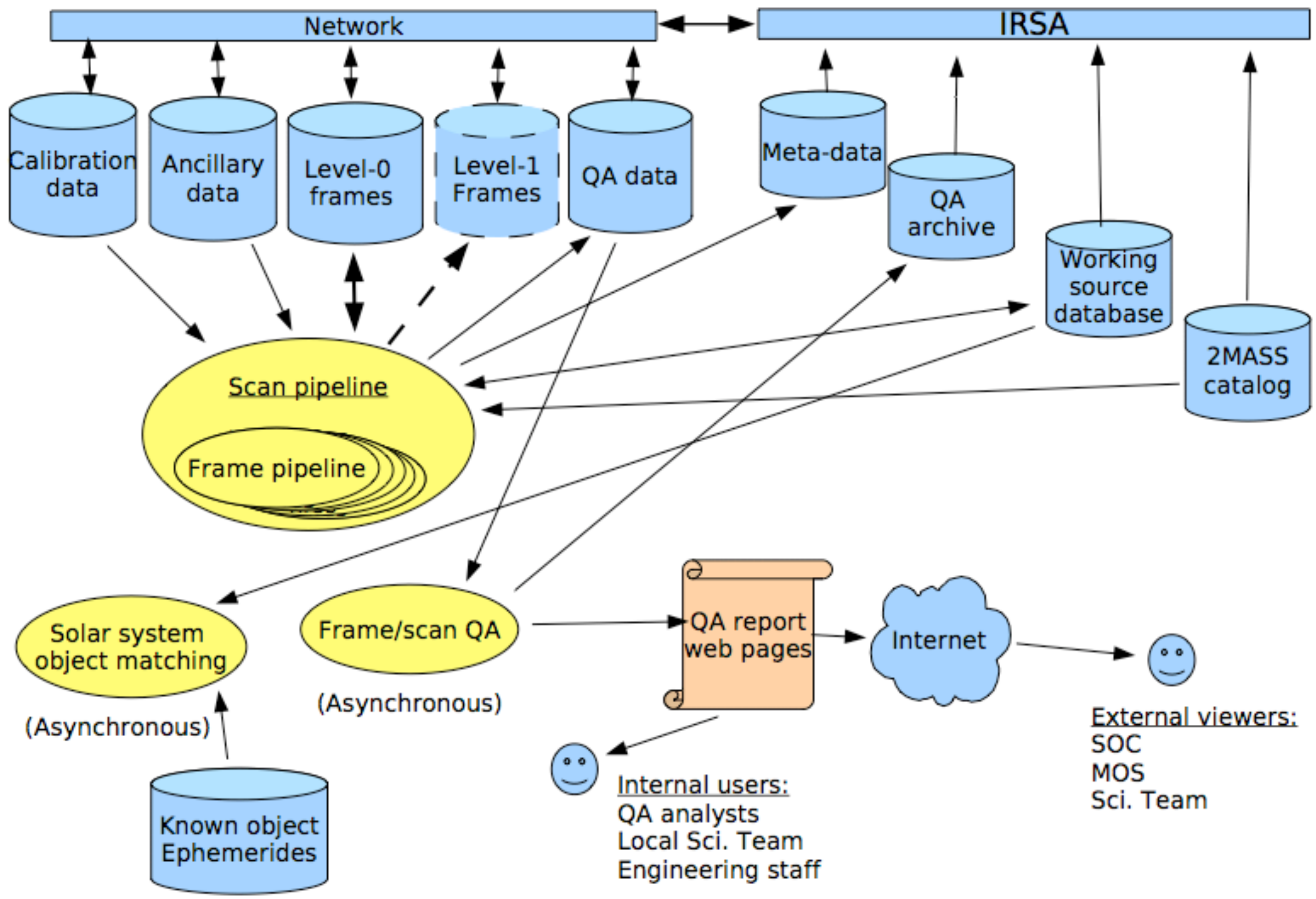
WSDS Subsystems: **Pipelines**

- **Scan and Frame Pipelines**

For a complete scan, calibrate frames, produce FITS level-1 frame images, archive extracted sources to frame Working Database, save meta-data and QA data.

  - Start and monitor **Frame Pipeline** for designated frames in scan
  - Calibrate frames
    - Flag and mask hot pixels, artifacts, saturation, cosmic ray hits, etc.
    - Offset correction from reference pixels (TBD)
    - Subtract darks, apply pixel response, linearity, and droop corrections
    - Sky offset correction from dynamic illumination profile
  - Extract, characterize and band merge sources from level-0 frames
  - Refine frame position/orientation/scale from comparison with 2MASS Point Source Catalog matched to high SNR extraction list
  - Apply refinements to frame meta-data and source positions
  - Apply photometric offsets to frame meta-data, source fluxes
    - Compute offsets from matched cal stars in most recent polar fields
  - Write source list and meta-data to frame Working Database
  - Search for solar system objects in level-1 working source DB
  - Compute and write QA data

WSDS Subsystems:
Scan Pipeline Data Flow



WSDS Subsytems: **Pipelines**

- **Quicklook Pipeline**

Provides image quality (scan mirror synch) and other assessments of each delivery within 24 hours of receipt by examining ~5% of the data.

  - Run on ~10 of frames for each scan
  - Launched after each delivery regardless of frame completeness
  - Same as the Scan Pipeline with reduced output
    - No level-1 frames generated, so less exacting astrometry and photometry
    - No archive output except QA-specific results and archived meta-data
  - Verify scan mirror synchronization from PSFs
  - Compute QA metrics and compare to predefined standards
    - System throughput from photometric standards
    - Image noise, sensitivity
    - Monitor artifacts, detector behaviour
  - QA output examined by mission planning at SOC (UCLA) and problems reported to MOS (JPL)
    - Remotely accessible HTML report and exported machine readable report

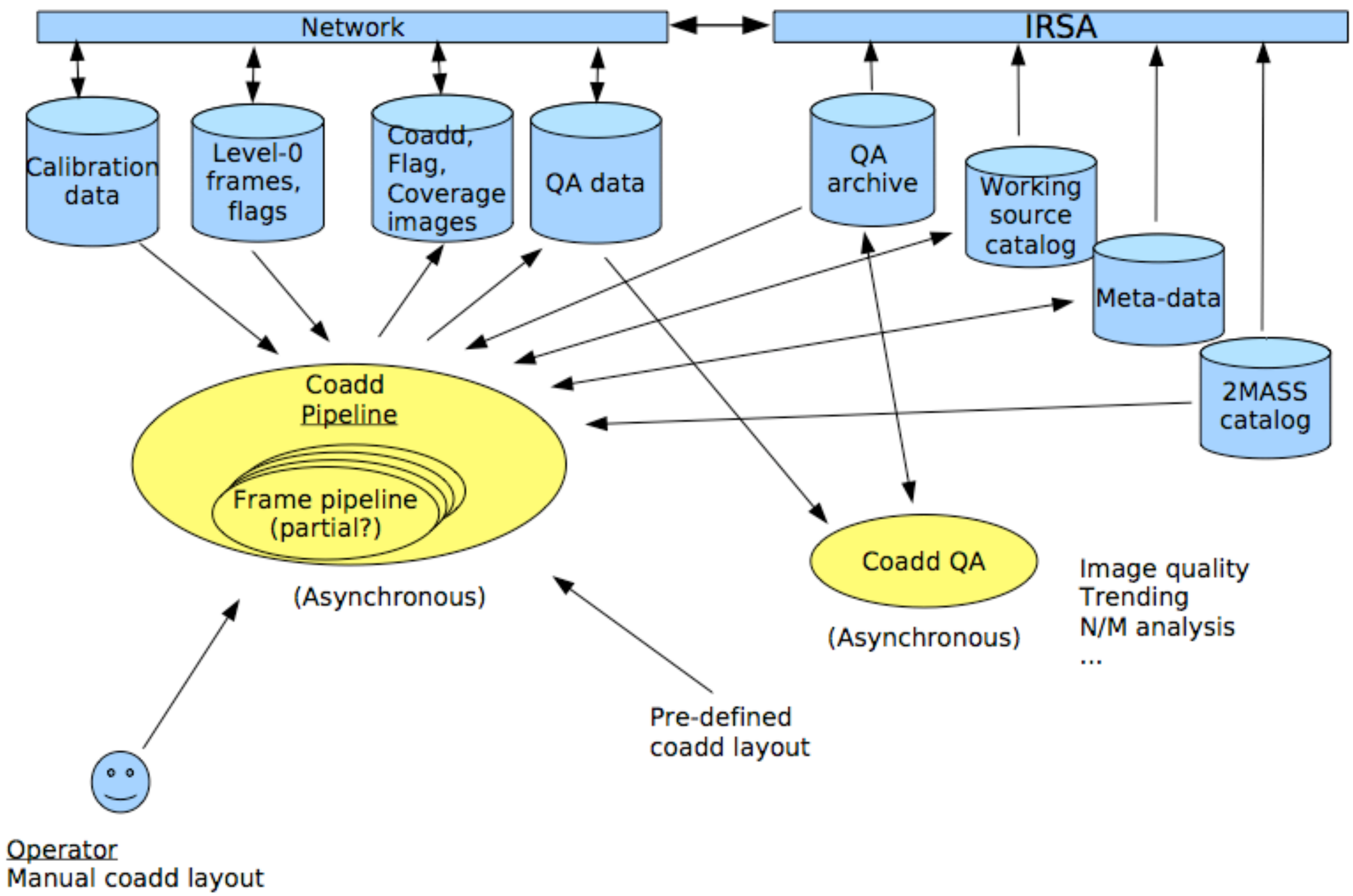
WSDS Subsystems: **Pipelines**

- **Coadd Pipeline**

Combine frame data covering the coadd geometry, extract sources, identify artifacts, Update calibration. Output coadd images and sources to the coadd Working Database.

- Create coadds as often as once per delivery or as rarely as once per week
- Select pre-defined or manual coadd geometry (RA, Dec, size, epoch)
- Calibrate level-0 frames. resample to produce coadd pixels
- Construct intensity, flag and coverage images
  - Examine frame flag images and meta-data
    - Reject frames failed by QA
    - Apply flags (artifacts, saturation, etc.), flag (possibly mask) out-of-bed frame pixels (cosmic ray hits, reflections, other outliers)
  - Propagate modified flags to flag map
  - Record coverage (exposure? variance? throughput?) in supplementary image(s)
- Extract and characterize sources, band merge, re-derive astrometric solution
- Write coadds to coadd archive
- Write sources to coadd Working Database and meta-data to archive
- Write QA data

WSDS Subsystems: Coadd Pipeline Data Flow



WSDS Subsystems: **A Note About Concurrency**

- Meeting our throughput requirements demands a high level of concurrent processing
- Our unit of concurrence will be processing level-0 frames
- Will need to process perhaps 100-200 frames at once
- Since CPUs are easier and cheaper to scale than network capacity and disk access, we will emphasize on-the-fly processing over large-scale storage of intermediate results
- Consequently, **level-1 frame pixels will be mostly computed on demand**, though they may be cached
  - Avoids having to replace them when the processing or calibration changes
  - Concurrency means we can compute a **lot** of level-1 frames in about the same time it takes to create one (we hope)

WSDS Subsystems: **Quality Assurance**

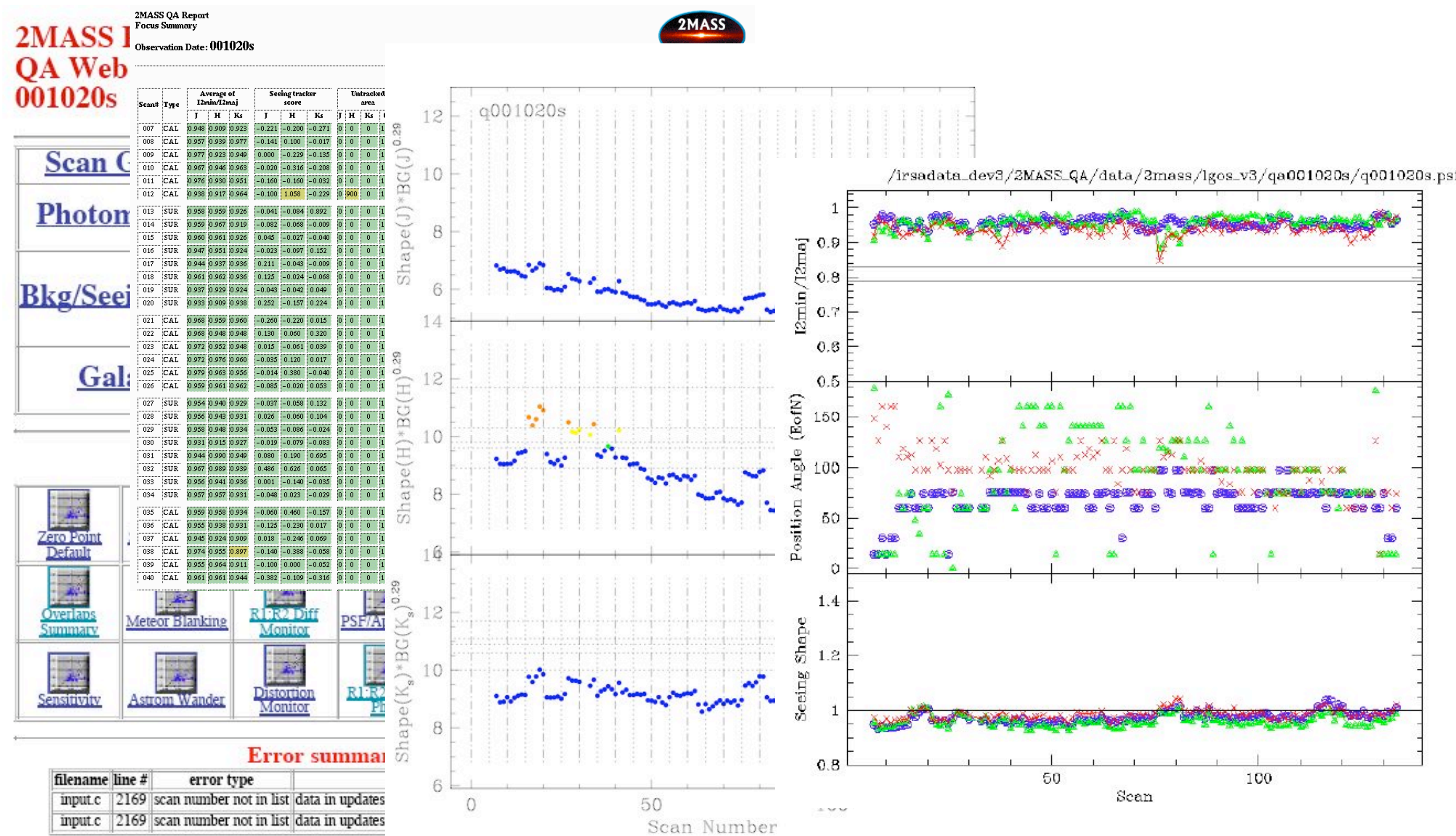
- **Frame and Multi-frame QA**

Generate concise, web-based reports summarizing science data quality. QA analysts generate a QA score for each frame. Key results are archived.

- Draws from:
  - Multi-band scan pipeline output, including specialized QA results
  - IRSA searches: 2MASS, extractions from overlaps, etc.
  - Ancillary-data: Ephemeris (SAA or moon proximity), temp.s, etc.
- Manual examination of trend plots with data from every frame
- Specialized tools allow interactive QA analysis
- Evaluate, scan mirror synchronization, PSF trend, efficacy of artifact removal, flat performance, scan coverage, astronomical properties (logN/logS, color-color plots, astrometry evaluation, etc.)
- Multi-frame analysis adds N/M results (C&R, photometric repeatability), artifact trending, coadd image quality
- Write QA report and key results to QA archive
  - Compare QA metrics to predetermined thresholds, generate QA score
  - Generate human (HTML) and machine-readable reports
  - Final QA disposition approved by ST designee

National Aeronautics and Space Administration
Jet Propulsion Laboratory
California Institute of Technology

WSDS Subsystems:
2MASS QA Example



WSDS Subsystems:
Final Product Generator

Constructs WISE Preliminary and Final Image Atlas and Source Catalog from coadd Image Archive and coadd Working Databases

- Commences after final all-sky coadd generation and source extraction
- **Manually executed and controlled process** primarily involving DBMS queries
- Use selected final coadded images on pre-defined sky tiling
- Examine QA and produce release-quality images, rejecting inferior frame data
- Create value-added columns in working database
- Create level-2 products in final format
- Analyze and validate final products (internal & Science Team)
- Iterate as necessary
- Distribute through IRSA

WSDS Subsystems: **Executive Functions**

Provide a uniform interface for execution and control of routinely-executed WSDC applications and utilities and interfaces aiding automation and resource management.

- **Application wrappers**
  - Standard parameter interface
  - Data dependency setup (moving, renaming, or massaging files, etc.)
  - Textual error, warning and informatory output management
  - Process status handling and error notification
  - Internal sub-process initiation and monitoring
- **Pipeline initiation and management**
  - Dependency-driven automatic start-up, e.g. ingest launches scan pipeline
  - Manual parameter-controlled CLI start-up, e.g. some coadd generation, special runs
  - Frame pipeline **concurrent** execution
- **Execution monitoring**
  - Web-accessible centralized process display
  - Controller notification of completion and failures
- **Resource monitoring**
  - Centralized monitoring of disk space, CPU and network load

WSDS Subsystems: **Archive**

Supply permanently saved data to both internal (pipelines and QA) and external (Sci. Team, public) users.

- **Multi-tiered approach to data archiving and access**
- **Raw mission telemetry archive (from INGEST)**
  - Copies to tape. On- and off-site storage
  - Live disk access
- **Level-0 frame data in distributed, multi-node filesystem**
  - Optimized for parallel access
- **Mission ancillary data from MOS**
- **Working databases, catalogs, meta-data, and QA results in DBMS integrated into IRSA infrastructure**
  - Catalog products are implicitly delivered to NASA-designated archive
  - Development minimized
  - IRSA/WISE web-based tools are primary Project interface to processed data
  - Public Interfaces are a subset of Project interfaces

Development Strategy: **Code Maturation**

Code evolves from early prototypes, which exercise end-to-end interfaces, to operational readiness through increasing functionality, adherence to documentation, coding and testing standards. A "Mayo-Smith's Pyramid" development approach is sought where the code is usable at all development stages but increases in functionality and rigor as it evolves.

- **New code starts at prototype level**
  - Establish interfaces, exercise critical code paths
  - Some functionality may be dummied out or otherwise not meet requirements
  - Code should be highly modular, but not necessarily fully up to coding standards
- **Leaves prototype stage at a subsequent major delivery**
  - Coding standards met
  - Major interfaces moderately stable
  - Interface documents complete
  - Code may continue to evolve for some time after leaving prototype
- **Code operationally ready**
  - Interfaces are mature and stable
  - Requirements are met in testing
  - RTB written, unit and end-to-end integrated ("in-situ") testing are complete
  - Can be re-opened for bug fixes, or modification if subsequent experience warrants

Development Strategy: **Capability Phase-in**

- **Parallel development of WSDS subsystems**
  - By end of FY08, several development tracks will be underway simultaneously
  - As existing code matures developers can pick up new tasks
  - New hires pick up new tasks; minimize code hand-offs
  - Limited simultaneous development by one developer gives schedule flexibility at a cost of small efficiency loss
- **Feature set at each version matched to ...**
  - Project activities, particularly instrument development and testing, and data production
  - Support for future development
  - Estimated development time and length of maturation period
  - Staffing profile

Development Strategy: **Schedule**

- **Version 0 :** 10/2007 (CDR - 3 months)
  Support data flow experiments, CDR preparation, survey planning
  - Hollow end-to-end frame pipeline prototype
  - Concurrency experiments
  - Early experiments on source extraction (astrometry, photometry) , pixel upsampling and coaddition
- **Version 1 :** 07/2008 **(**Instrument calibration - 3 months, E-to-E test)
  Support instrument characterization
  - QA tools, data display
  - Multi-origin (sim telemetry, raw images) data ingest, correlation with ancillary data
- **Version 2 :** 01/2009 (Mission scenario testing – 1 month)
  Support full telemetry I/O and WSDC product life cycle
  - Frame pipeline mature
  - Scan, coadd pipelines, QA, quicklook feature complete
  - Executive functions feature complete

Development Strategy: **Schedule**

- **Version 3 :** 08/2009 **(**Launch - 3 months)

  Support ORT, IOC, Early operations
  - All sub-systems mature except FPG
  - Begin strict change control (CCB review)
  - Used for parameter tuning prepatory to v3.5
- **Version 3.5:** 02/2010 **(**Launch + 3 months)

  Support routine operations and preliminary data release
  - Bug fixes and adjustments for in-orbit performance
- **Version 4 :** 10/2010 (End of on-orbit operations + 4 months)

  Support data reprocessing and final product delivery
  - Updated calibration
  - Code updates to incorporate in-orbit performance
  - QA improvements

Development Strategy: **Software Management**

- **Coding standards**
  - Language choice: C, C++, Fortran, Perl, IDL (not in pipelines!), shell
  - Code commenting, transparency and modularity
  - Error control and reporting, completion status
- **Revision control** (Subversion)
  - Software, parameters, and documents
  - Repository check-in/check-out structure
  - Mandatory check-in and tag at delivery time
- **Delivery control**
  - Directory based separation between ops, dev, integ, test, etc., deliveries
  - Makefile template-based builds
  - Change control
    - Initiation of change control at L-1 year (TBD)
    - Choose CCB members from WSDC and representatives from MOS and ST
    - Phased increase in code stability as launch approaches
- **Problem tracking (**Bugzilla?)
  - Developer or user initiated
  - Some issues elevated to project-level tracking

Hardware Architecture

- **Computation**
  - **~20-node, x64-based Linux Beowolf cluster** (~$3.5k/node currently)
  - Driven by scan pipeline and QA run on a day's deliveries in 8 hours
    - For 30,000 frames/day (all bands) / 8 hours = throughput of ~1 frame/sec
    - Comparison with 2MASS processing implies each frame will take ~120 seconds at 50% CPU utilization with 100% pad
    - **Implies ~120 cores required**
    - Additional cores required for QA, coadding, ingest, misc. analysis, etc.
- **Storage** (7 month mission)
  - **~30 TB distributed with cluster nodes** (~15 TB X2 for replication)
  - **~30 TB in NFS exported RAID-5**
  - Driven by 7 month mission archive
    - Raw frames: ~10TB
    - **Level-0 frames: ~30TB (compressed, on cluster nodes)**
    - Coadds: ~5TB
    - Frame cache and work space: ~5 TB
    - Source working databases, catalogs, metadata, etc.: ~5TB
    - Work space: ~5TB

Hardware Architecture

- **Network**
  - **2 parallel gigabit networks <u>minimum</u>**
  - Driven by level-0 frame access
    - Assume ~10 days-worth of level-0 pixels required in an 8 hour period
      - Hand-wavey: includes 1st runs, one rerun, coadding, QA, analysis
    - Implies **~200Mbit/s** on average, but bursty
    - Trade off complexity and cost vs. capacity
    - **Network usage is reduced if more frame processing is on local disk**
- **Backup**
  - **3-4 SDLTs per day**
  - Driven by frame backup
    - Raw telemetry: 25GB/day archived to 3 copies = 75 GB/day
    - Raw FITS frames: 50GB/day (uncompressed)
    - Level-0 frames: 130GB/day (uncompressed)
- **Security**
  - **IPAC border security is excellent**
  - Additional isolation for Ingest-1 servers

WSDS Architectural Issues

- **Inter-frame processing interactions in frame pipeline.** E.g. latent images. Are there others?
  - Interferes with concurrency
- **How well does frame processing time scale with CPU count?**
- **Latent image handling when frames are missing**
- **FPA surprises?**
- **Which bands are required for processing?**
- **Level-0 frame archive for 1-year mission; how do we scale storage on the cluster?**

Road to WSDS CDR

- **Version 0 delivery in fall**
  - Exercising key interfaces and data flow issues
  - Timing and data load studies
- **Processing time (concurrency) experiments**
  - Must understand how frame processing times scale with concurrency
  - Network load is key
  - Experiment with version 0 delivery
- **Detailed development milestones**
  - Matched to development versions, but more detailed
- **Updated versions of documents**
  - FRD, FDD, SMP