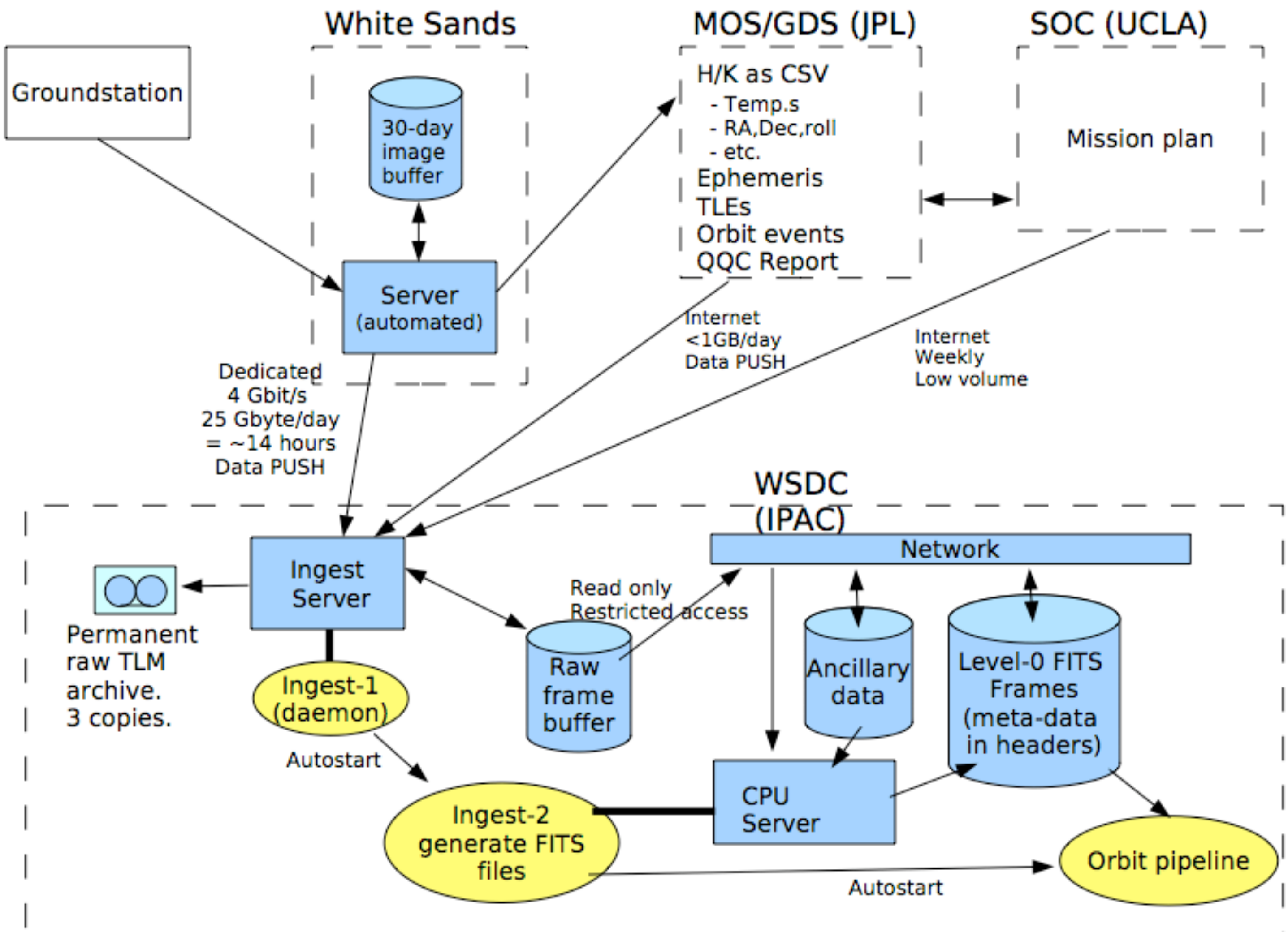
WISE Mission Operations System
Preliminary Design Review

WSDC Architecture

Tim Conrow - IPAC
WSDC Lead Engineer

Ingest Data Flow



Ingest Subsystem

- **Ingest, step 1**

Receive and verify high-rate Level 0 science data from White Sands. Write raw data tape archive.

- Runs on semi-isolated, dedicated server
  - External network has only read-only access
  - Limited remote access
- Watch for completion of transfers
- Move data to staging area for manual backup to permanent media (once per day) and downstream processing
- Copy ancillary telemetry to database and raw frame images to frame pool
- Auto-start Ingest step 2 on WISE CPU server

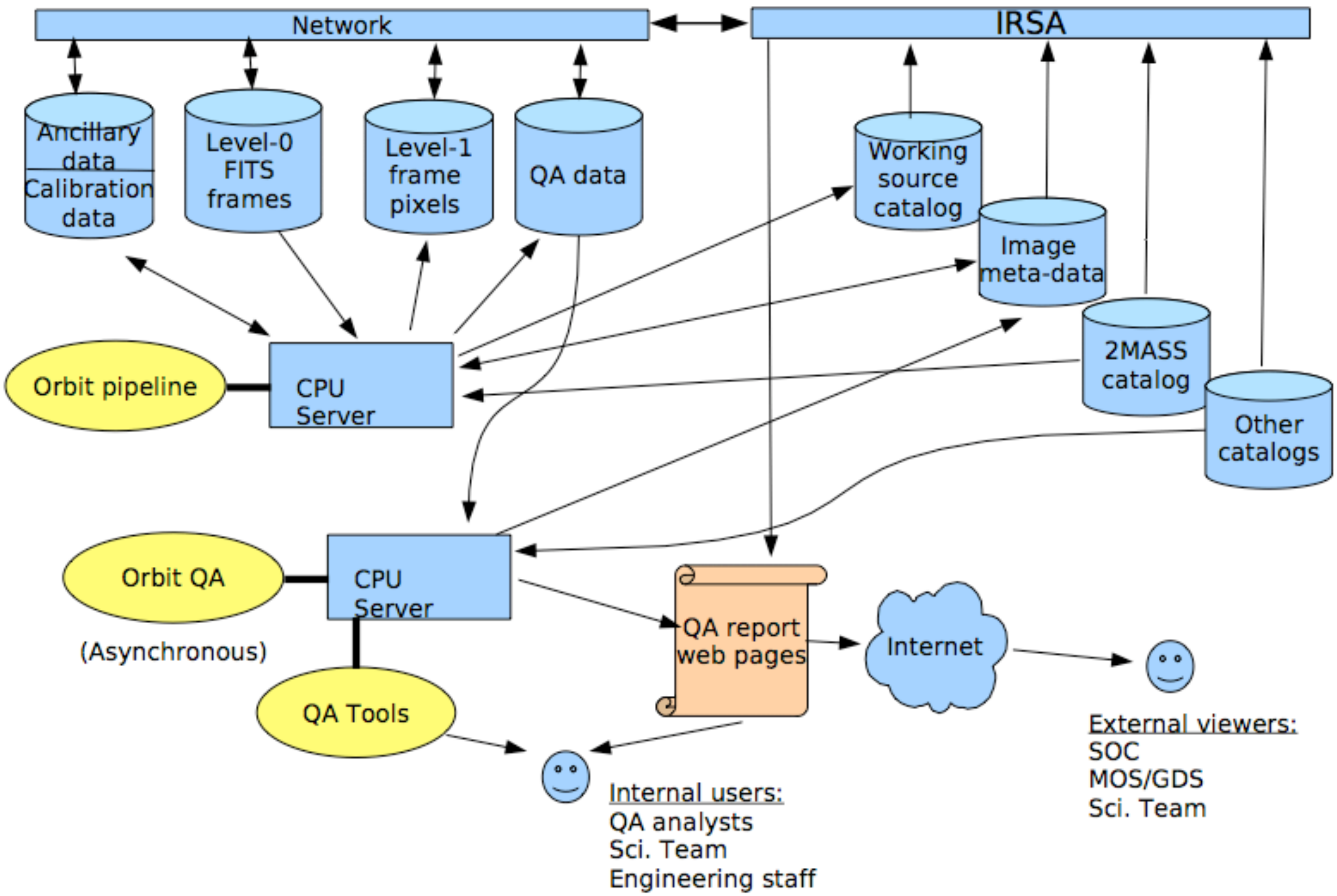
• Ingest, step 2

Decompress and assemble complete frames, integrate meta-data into FITS-format frame files. Assemble H/K and ephemeris data from EOS, science planning data from SOC.

- Runs on WISE CPU server on attached disk
- Establish which frames in buffer are complete
  - Frames may come out of order from multiple deliveries, so images and orbits may need to be stitched together using a directory of data in the frame pool
- Decompress and assemble complete frames
  - Insert blank pixels where QQC report says data are missing
- Collect meta-data for each frame
  - WCS info (RA,Dec of frame center, etc.)
  - Observation time range
  - Instrument telemetry, QQC report, etc.
  - Correlation to science plan
- Write losslessly-compressed level-1 FITS files with meta-data in headers
  - Image data
  - Flag map (artifacts, saturation)
- These will be the primary input to the orbit pipeline

Wide-field Infrared Survey Explorer (WISE)

Orbit Pipeline Data Flow



Data Reduction Pipelines

- **Orbit Pipeline**

Calibrate frames from a single orbit, write FITS frame image archive and extracted source Working Database.

- – Locate frames for one orbit when orbit is complete or 72 hours elapses since data cannot be recovered from S/C after 3 days (modulo re-transmissions from White Sands)
- – Generate flat field and illumination profile from orbit frame data and/or the calibration data library
- – Locate hot pixels from multi-frame pixel detection frequency histogram and ground calibration data
  - Update flag map
- – Apply flats, linearity correction (from ground calibration), and illumination profile (no darks for SUR data)

Data Reduction Pipelines

- **Orbit Pipeline**, continued
  - For each band, extract and characterize high S/N sources
    - Find regions of connected, above-threshold pixels (“segmentation”)
    - Find positions through centroiding and application of the preliminary astrometric solution
    - Compute multiple aperture fluxes, magnitudes, and curves of growth
    - Collect various quality statistics (S/N, uncertainties, flags, etc.)
  - Refine frame position/orientation from comparison of extracted positions with 2MASS Point Source Catalog
    - Cross-compare and unify solutions for four bands
  - Band merge; I.e. positionally associate extractions from different bands, resolve and flag conflicts, produce a single multi-band extraction with refined position

Data Reduction Pipelines

- **Orbit Pipeline**, continued
  - Identify and flag artifacts and saturation
  - Identify and flag known solar system objects
  - Derive and apply (provisional) photometric calibration
  - Save extraction list to the *Working Database*
  - Save image meta-data to database
  - Save calibrated, position-reconstructed (level-1) frames to archive
  - Write QA data

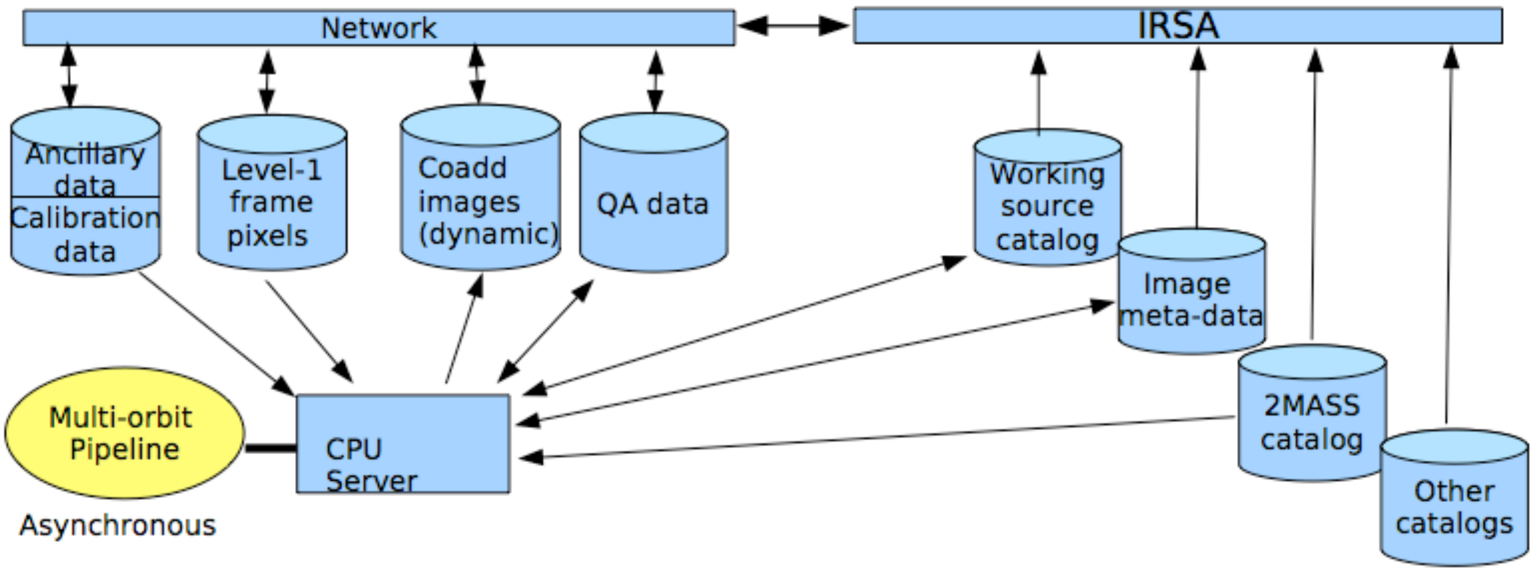
Data Reduction Pipelines

- **Quicklook Pipeline**

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

- Run on ~50 of frames for each orbit
- Same as the Orbit Pipeline with reduced functionality
  - Don't generate new calibration results
  - Less exacting astrometry and photometry
  - No archive output except QA-specific results
- Verify scan mirror synchronization from image shapes
- Compute other QA metrics
  - System throughput
  - Image noise
  - PSF shape
  - Scan placement
- QA output examined by mission planning at SOC (UCLA) and problems reported to MOS/EOS (JPL)

Multi-orbit Pipeline Data Flow



Data Reduction Pipelines

- **Multi-orbit Pipeline**

Combine frame data from multiple orbits, extract sources, identify artifacts, Update calibration. Output to image and source Working Databases.

- Run up to once per delivery or as little as once per week
- Construct images and flag and coverage maps from combined multi-frame data, producing a "coadd"
  - Select coadd geometry
  - Read overlapping calibrated frame images and flag maps; re-project and interpolatively up-sample frame pixels
  - Examine frame flag maps and meta-data
    - Reject pixels from frames failed by QA
    - Flag coadd pixels with out-of-bed frame pixels
      - » Some may be dropped from the coadd as radiation hits, moving objects, etc.
  - Propagate modified flags to new coadd flag map
  - Write coadd image, flag map, and mea-data to archive

- **Multi-orbit Pipeline**, continued
  - Extract and characterize sources
  - Band merge
  - Compute source detection frequency (N/M analysis) by comparison with frame extractions
  - Reconfirm or refine astrometric solution and photometric calibration
  - Write source data to *Working Database*
  - DBMS records must include meta-data sufficient to allow selection and removal of entries based on specific pipeline runs so working database can be purged of old coadd data
    - Selection criteria may include geometry, run and observation times, delivery ID, depth, calibration and pipeline versions
  - Write QA-specific data to meta-data archive

Quality Assurance

- **Frame QA**

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

- Draws from:
  - Multi-band Orbit Pipeline output, inc. specialized QA-oriented results
  - IRSA searches: 2MASS, extractions from overlaps, etc.
  - Ancillary-data: Ephemeris (SAA or moon proximity), QQC report (data gaps), temperatures, etc.
  - Boresite track
  - Results of manual examination of trend plots with data from every frame
- Most results are generated automatically at completion of the Orbit Pipeline
- Specialized tools allow interactive QA analysis

Quality Assurance

- **Frame QA**, continued
  - Evaluate
    - Scan mirror synchronization using PSF
    - Trending of PSF, astrometry and photometry
    - Efficacy of artifact flagging and/or removal
    - Flat performance
    - Orbit coverage (missing frames or parts of frames)
    - Overall image quality
    - Astronomical properties (logN vs. logS, color-color plots, etc.)
  - Automatic results are externally viewable on a web-based top-level summary. More detail accessible by drilling down through links
  - Analyst score written to frame meta-data and sent to Mission Planning for feedback to MOS/EOS

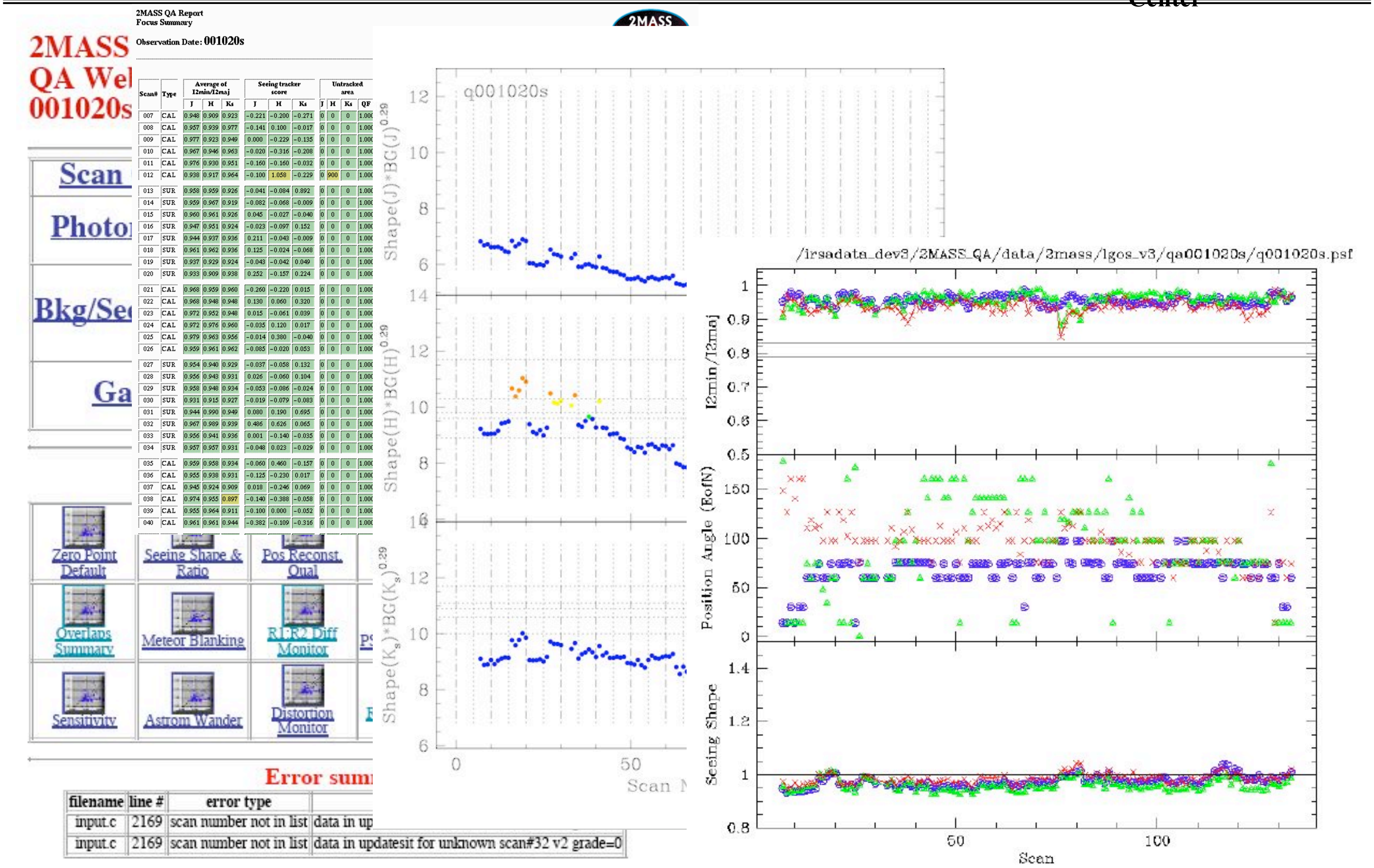
Quality Assurance

- **Quicklook QA**
  - Subset of Frame QA
  - Examine 5% of orbit data
  - Emphasizes anomalies/problems for which a rapid response is necessary
  - Loss of scan mirror synchronization is the main issue
  - Other instrument or spacecraft anomalies may be added

Quality Assurance

- **Multi-orbit QA**
  - Similar to frame QA, but add ...
  - Coverage analysis (e.g. create gap report)
  - Source reliability and stability stats from N/M analysis
  - Artifact trending (i.e. comparison of overlapping frame artifacts)
  - Trend calibration especially flats
  - Other trending analyses

Quality Assurance Subsystem



Final Product Generator

- **Final Product Generator**

Constructs WISE Preliminary and Final Image Atlas and Source Catalog from multi-orbit Image Archive and extracted source Working Databases

- **Manually executed and controlled process** primarily involving DBMS queries of extraction and image meta-data Working Databases
- Examine QA data and select final combined images of release-quality
- Create value-added columns in Working Database
- Perform source selection DBMS query
- Create products in final format
- Analyze and validate final products (internal & Sci. Team)
- Iterate as necessary

Pipeline Executive Functions

- **Pipeline Executive**

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
  - Textual error and informatory output management
  - Process status handling and error notification
  - Internal sub-process initiation and monitoring
- Pipeline initiation
  - Dependency-driven automatic start-up
  - Manual parameter-controlled CLI start-up (e.g. for testing or special-purpose analysis)
- Execution monitoring
  - Web-accessible centralized process display
  - Controller notification of completion and failures
- Resource monitoring
  - Centralized monitoring of disk space, CPU and network load

Pipeline Executive Functions

GALEX Job Status

Viewing Constraints Hide

Admin action none **Operator** unknown

Update Reset

Job list for /home/galex/lnk/slates//ops/pipe **17:42:29 PDT Thu Sep 01 2005 00:42:29 GMT Fri Sep 02 2005**

Select All None	ID	Name	State	Status	Elapsed Time (Total/step)	State Mod Time	Out	Log	Command	Host	PID	Directory
	auty	orbpipe (AISCHV3_374_44973_0004)	queued	?	8.6m	050902T003353Z	Out	OrbPipe	CmdLine	*	3094	11373-AISCHV3_37.../d/00-visits/0004-img/01-try/
	autx	orbpipe (GI1_114001_RXJ0122m7521_0011)	running	GDeltaPhot	9.6m/ 2.1m	050902T003251Z	Out	OrbPipe	CmdLine	pea#1	24717	21825-GI1_114001.../d/00-visits/0011-img/01-try/
	autw	orbpipe (ELAISS1_11_0007)	running	PhotMap	10.4m/ 1.9m	050902T003202Z	Out	OrbPipe	CmdLine	turnip#2	7083	06474-ELAISS1_11/d/00-visits/0007-img/01-try/
	autv	orbpipe (ELAISS1_11_0005)	running	SExtra	10.6m/ 1.4m	050902T003151Z	Out	OrbPipe	CmdLine	pea#2	24421	06474-ELAISS1_11/d/00-visits/0005-img/01-try/
	autu	orbpipe (ELAISS1_11_0004)	running	PhotMap	10.8m/ 4.7m	050902T003141Z	Out	OrbPipe	CmdLine	turnip#1	6987	06474-ELAISS1_11/d/00-visits/0004-img/01-try/
	autq	orbpipe (AISCHV3_422_39962_0001)	done	0/0(0)	5.1h	050901T193605Z	Out	OrbPipe	CmdLine	turnip#1	29695	11392-AISCHV3_42.../d/00-visits/0001-img/01-try/
	auts	orbpipe (WDST_LDS749B_GRID_10_20_0017)	done	0/0(0)	5.2h	050901T192822Z	Out	OrbPipe	CmdLine	turnip#2	8310	09340-WDST_LDS74.../d/00-visits/0017-img/01-try/
	autr	orbpipe (WDST_LDS749B_GRID_12_33_0032)	done	0/0(0)	5.9h	050901T184956Z	Out	OrbPipe	CmdLine	pea#2	8520	09385-WDST_LDS74.../d/00-visits/0032-img/01-try/
	auts	orbpipe (PAVO_GROUP_0014)	done	0/0(0)	8.4h	050901T162010Z	Out	OrbPipe	CmdLine	pea#1	23752	05678-PAVO_GROUP/d/00-visits/0014-img/01-try/
	autp	orbpipe (GI1_043001_AE_AQR_0031)	done	0/0(0)	9.4h	050901T152040Z	Out	OrbPipe	CmdLine	pea#1	8047	21279-GI1_043001.../d/00-visits/0031-img/01-try/
	auto	orbpipe (GI1_043001_AE_AQR_0030)	done	0/0(0)	9.6h	050901T150414Z	Out	OrbPipe	CmdLine	turnip#2	29179	21279-GI1_043001.../d/00-visits/0030-img/01-try/
	autm	orbpipe (PAVO_GROUP_0015)	done	0/0(0)	15.0h	050901T094316Z	Out	OrbPipe	CmdLine	turnip#2	18518	05678-PAVO_GROUP/d/00-visits/0015-img/01-try/
	autn	orbpipe (ELAISS1_11_0003)	done	65280/255(0)	15.1h	050901T093347Z	Out	OrbPipe	CmdLine	turnip#1	18640	06474-ELAISS1_11/d/00-visits/0003-img/01-try/
	autl	orbpipe (GI1_043001_AE_AQR_0029)	done	0/0(0)	1.0d	050831T233951Z	Out	OrbPipe	CmdLine	pea#1	19515	21279-GI1_043001.../d/00-visits/0029-img/01-try/
	autk	orbpipe (GI1_043001_AE_AQR_0028)	done	0/0(0)	1.1d	050831T232342Z	Out	OrbPipe	CmdLine	turnip#2	19935	21279-GI1_043001.../d/00-visits/0028-img/01-try/
	autj	orbpipe (GI1_043001_AE_AQR_0027)	done	0/0(0)	1.1d	050831T231725Z	Out	OrbPipe	CmdLine	turnip#1	19717	21279-GI1_043001.../d/00-visits/0027-img/01-try/
	auth	orbpipe (GI1_043001_AE_AQR_0025)	done	0/0(0)	1.1d	050831T230048Z	Out	OrbPipe	CmdLine	pea#2	15075	21279-GI1_043001.../d/00-visits/0025-img/01-try/
	auti	orbpipe (GI1_043001_AE_AQR_0026)	done	65280/255(0)	1.1d	050831T225505Z	Out	OrbPipe	CmdLine	pea#1	15186	21279-GI1_043001.../d/00-visits/0026-img/01-try/
	autf	orbpipe (GI1_043001_AE_AQR_0023)	done	0/0(0)	1.6d	050831T110306Z	Out	OrbPipe	CmdLine	pea#1	24157	21279-GI1_043001.../d/00-visits/0023-img/01-try/
	autg	orbpipe (GI1_043001_AE_AQR_0024)	done	0/0(0)	1.6d	050831T105803Z	Out	OrbPipe	CmdLine	turnip#1	6875	21279-GI1_043001.../d/00-visits/0024-img/01-try/
	aute	orbpipe (GI1_043001_AE_AQR_0022)	done	0/0(0)	1.6d	050831T103419Z	Out	OrbPipe	CmdLine	pea#2	13848	21279-GI1_043001.../d/00-visits/0022-img/01-try/
	auta	orbpipe (GI1_043001_AE_AQR_0018)	done	0/0(0)	1.6d	050831T100801Z	Out	OrbPipe	CmdLine	turnip#1	20316	21279-GI1_043001.../d/00-visits/0018-img/01-try/
	autb	orbpipe (GI1_043001_AE_AQR_0019)	done	0/0(0)	1.6d	050831T095612Z	Out	OrbPipe	CmdLine	turnip#2	20665	21279-GI1_043001.../d/00-visits/0019-img/01-try/

Pipeline Executive Functions

GALEX CPU Status

17:26:54 PDT Thu Sep 01 2005 (00:26:54 GMT Fri Sep 02 2005)

Host	Alias	CPU Load	Mem Load	System Info	1st Cmd	1st CPU	1st User	GALEX Users
Angerboda		0.09/0	0.00	1x0.5GHz 0.7GB linux	X	39.2%	root	dme
Balderblom	glxg2	0/1	0.00	1x2.3GHz 0.9GB linux	init	0.0%	root	dme,krl
Berserk	glxg3	0.07/1	0.00	1x1.8GHz 0.9GB linux	init	0.0%	root	
Beyla	glxp1	0.515/1	0.00	2x1.9GHz 2.0GB linux	convert	7.9%	gop	gop
Bil		1.17/3	0.31	1x1.3GHz 0.9GB linux	X	95.9%	root	raymond
Cabbage		0.515/3	0.11	4x3.0GHz 3.9GB linux	poissonbg	18.7%	gop	gop
Esne	glxp4	0/0	?	2x1.9GHz linux	256	!!!%	/home/galex/ops/bin/topd:	
Fennel		0.0225/1	0.00	4x3.1GHz 5.6GB linux	init	0.0%	root	gop
Fezzik	glxa1	0/0	?	2x1.1GHz linux	8	!!!%	/home/galex/ops/bin/topd:	
Forsete	glxwww	0/1	0.00	2x1.9GHz 2.0GB linux	init	0.0%	root	
Garm	glxp3	0/0	?	2x1.9GHz linux	256	!!!%	/home/galex/ops/bin/topd:	
Goths	glxbkup	1.1/2	0.00	1x2.3GHz 0.5GB linux	cksum	34.7%	dme	dme
Heywood		0/2	0.00	1x3.5GHz 2.0GB linux	init	0.0%	root	tab
Kale		0.3275/1	0.00	4x3.1GHz 5.6GB linux	orbpipe	0.9%	gop	gop
Lorax		0.02/1	0.00	1x1.8GHz 0.9GB linux	init	0.0%	root	krl
Ora		0/1	0.00	1x2.3GHz 0.9GB linux	init	0.0%	root	krl,mcneill,patrick
Pea		0.0125/0	0.00	4x3.1GHz 5.6GB linux	ginit	2.9%	gop	gop,min,mseibert
Radish		0.51/0	0.00	4x3.1GHz 5.6GB linux	sex	36.5%	gop	gop
Skuld		0/0	0.00	1x3.5GHz 2.0GB linux	init	0.0%	root	krl,mcneill,pherrera
Spinach		0/1	0.00	4x3.0GHz 3.9GB linux	init	0.0%	root	
Syn	glxdb	0.545/1	0.00	2x0.7GHz 3.8GB linux	postmaster	80.5%	postgres	postgres
Turnip		0.01/0	0.00	4x3.0GHz 3.9GB linux	init	0.0%	root	gop
Vanaheim	glxg1	0/1	0.00	1x2.3GHz 0.9GB linux	init	0.0%	root	dme,patrick,pherrera

Pipeline Executive Functions

GALEX Disk Status

Current time: 17:37:26 PDT Thu Sep 01 2005 (00:37:26 GMT Fri Sep 02 2005)
File=/home/galex/topd/dutree.garm.ready

Disk	Alias	%-Used	Used	Free	Total	Host	User 1	% User 1
/home/galex		78%	50G	14G	64G	runes	gop	33.2%
/home/mani		46%	40G	47G	91G	mani	gop	28.6%
/home/slates	slates	9%	173M	1.8G	2.0G	runes	gop	6.7%
/home/lin	ops4	75%	346G	117G	467G	lin	gop	66.9%
/home/runes2	dev	93%	553G	48G	606G	eisa	krl	39.3%
/home/syn	misc	55%	209G	178G	407G	syn	galex	0.0%
/home/forsete2	www	96%	2.7T	114G	2.8T	svalin	tim	6.9%
/home/ops1	ops1	84%	788G	161G	958G	lin	tim	32.8%
/home/ops2	ops2	96%	2.6T	116G	2.8T	svalin	tim	6.9%
/home/ops3	ops3	89%	283G	39G	324G	lin	gop	86.9%
/home/lin5	ops5	73%	655G	254G	918G	lin	gop	63.9%
/home/lin6	ops6	90%	325G	39G	367G	lin	gop	86.8%
/home/lin7	ops7	91%	826G	83G	918G	lin	gop	89.7%
/home/dbms	dbms	60%	607G	419G	1.1T	syn	root	0.0%
/home/runes3	ops8	83%	789G	164G	962G	runes	tim	25.8%
/home/runes4	ops9	93%	320G	25G	348G	runes	gop	91.6%
/home/runes5	ops10	88%	811G	115G	936G	runes	gop	85.1%
/home/runes6	ops11	92%	836G	76G	921G	runes	gop	90.5%
/home/runes7	ops12	87%	787G	126G	921G	runes	gop	85.3%
/home/runes8	ops13	86%	778G	135G	921G	runes	gop	58.9%
/home/lin8	ops14	96%	2.6T	138G	2.8T	lodur2	gop	90.7%

Low	Moderate	High	Full	Out

Development Strategy

- **Development process**
  - The IPAC Way
    - Algorithm development drawing on extensive institutional know how
    - Close frequent contact between cognizant Science Team members and the development staff
    - Rapid, low-overhead code production
  - The cycle of (software) life
    - Algorithm development
      - Cog. Sci. > Developer > Algorithm development > Prototype > Test > Repeat
    - Refinement, Maturation and Delivery
      - Code > Unit test > Delivery > In situ test > RTB > Repeat
    - Deliveries have varying degrees of formality
      - Developer deliveries. Used in early development and between other releases
        - » Asynchronous, frequent, informal, no RTB
      - Intermediate deliveries. Starting before version 1
        - » Coordinated, ~monthly, semi-formal, RTB phased in
      - Release deliveries. After version 1
        - » Coordinated, matched to mission milestones, formal, RTBs

Development Strategy

- **Testing**
  - Testing will be a routine, integrated part of code development
  - Developing code will also see frequent use as a analysis tools, thus testing in use
  - Test results examined by developers, instrument engineers, Science Team members as appropriate on a frequent basis
  - Test data will increase in completeness and fidelity
    - Pre-version 1
      - Simulated frame data of increasing sophistication
        - » Delivered by UCLA, informed by array data as it becomes available
      - Preliminary array output
    - Post-version 1
      - Instrument frame data
    - Post-version 2
      - S/C and instrument housekeeping telemetry incorporated

Development Strategy

- **Code maturation**
  - New code starts at prototype level
    - Establish interfaces
    - Some functionality may be dummied out or otherwise not meet requirements
    - Code quality not fully up to coding standards
  - Leaves prototype stage at a subsequent major delivery
    - Coding standards met
    - Interfaces moderately stable
    - Interface documents complete
    - Used in-situ
    - Code may continue to evolve for some time after leaving prototype
  - Code complete prior to operational use
    - Interfaces are mature and stable
    - Requirements are met
    - Unit and realistic integrated testing are complete
    - Can be re-opened for bug fixes, or for further modification if subsequent experience warrants

Development Strategy

- **Capability phase-in**
  - Parallel development
    - By 2007, 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
  - Feature set at each version matched to …
    - Project activities, particularly instrument development and data production
    - Estimated development time and length of maturation period
    - Staffing profile

Development Strategy

- **Capability Phase-in,** continued
  - **Version 0**: April 2007
    - Project activity: Simulation and early instrument data flow
    - Support basic data analysis and algorithm development
    - FITS image data I/O, conversion of raw image data to FITS
    - Elementary pixel upsampling, interpolation and coadding
    - Use of IRSA
  - **Version 1:** April 2008 (instrument cal. -1 month)
    - Project activity: Instrument characterization
    - Support detailed instrument data analysis
    - Correlation with instrument H/K and ancillary data
    - Orbit pipeline feature complete
    - Prototype frame-QA

Development Strategy

- **Capability Phase-in,** continued
  - **Version 2**: September 2008 (end-to-end test -1 month)
    - Project Activity: End-to-end testing
    - Support full telemetry I/O and WSDC product life cycle
    - Ingest step 1 prototype
    - Ingest step 2 feature complete
    - Frame QA feature complete
    - Multi-orbit pipeline feature complete
    - Multi-orbit pipeline QA feature complete

Development Strategy

- **Capability Phase-in,** continued
  - **Version 3:** Feb. 2009 (Launch -3 months)
    - Project activity: Final ground tests, IOC and operations
    - Support full product lifecycle and preliminary data release
    - All modules required for operations complete
    - Beginning of strict change control.
    - Final product generation prototype
  - **Version 4:** April 2010 (end of on-orbit ops. +4m)
    - Project activity: Preliminary data release
    - Support final data processing and final product delivery
    - Final product generation feature complete

Development Strategy

- **Software management**
  - Documents
    - Functional Design Document
    - System Interface Specifications
    - Sub-system Design Specifications
  - Coding standards
    - Fixed set of languages
    - Parameter handling
    - Commenting and transparency
    - Error control and reporting
    - Isolation of I/O and architecture/OS dependencies
  - Software, parameter, and document revision control
    - CVS or better
    - Repository check-in/check-out workflow
    - Regular developer-initiated check-ins
    - Coordinated test builds after version 2
    - Mandatory check-in and tag at delivery time

Development Strategy

- **Software management**, continued
  - Delivery control
    - Directory-based isolation of all delivery products and their dependencies and required parameters
    - Code state is tagged in the revision control system at delivery time and delivered from fresh checkout
    - Dependency-based builds
  - Problem tracking
    - GNATS or better (please!). Web accessible
    - Some problem levels elevated to project tracking
  - Change control
    - Initiation of change control between versions 2 and 3 using a WSDC-based Change Control Board
    - Phased increase in code stability as launch approaches

Hardware Architecture

- **Facilities**
  - WISE will have its own dedicated hardware integrated into the IPAC facilities and network infrastructure
- **Security**
  - IPAC border security is excellent
    - Compliant with JPL D-7155E; "Automated Information Security Requirements for Computer System Administrators" 11/96
  - Additional isolation for ingest (step 1) servers
- **CPU Load Requirements**
  - A day's orbit and multi-orbit pipelines plus another day's data running in parallel in 8 hours (~4X 2MASS data rate); 33% margin
  - Support ongoing interactive analysis and QA with good response time
- **Disk Requirements**
  - Keep 7 mo.s of frame images and meta-data
  - Space for coadds sufficient to tile sky
  - Support required network access to the images and source databases

Hardware Architecture

- **CPU and Memory**
  - Homogeneous architecture/OS
  - 2 CPUs supporting ingest step 1 plus hot spare
  - 4 CPUs on each of 8 CPU servers, supporting
    - Ingest step 2 (2)
    - 1 CPU per orbit pipeline per ingest (8)
    - Parallel orbit pipeline processing (8)
    - Multi-orbit pipeline (4)
    - QA, analysis (2)
    - 2 servers detailed to IRSA for dedicated WISE use
  - 2 GB RAM per CPU, 8GB per machine
  - ~20 workstations of modest capability

Hardware Architecture

- **Disk**
  - RAID-5 disk for a 7 month mission
  - All images stored losslessly compressed
  - Level 1 frame pixels: ~50TB
  - Coadds: ~30TB
  - Source databases, catalogs, metadata, etc.: ~4TB
  - Additional disk needed for extended mission

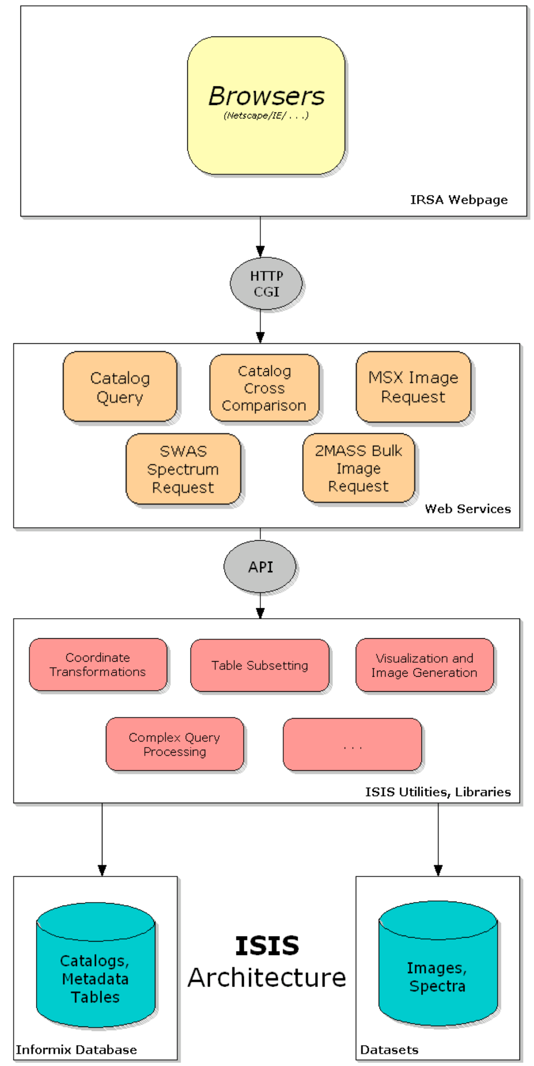
Hardware Architecture

- **Network**
  - Assume 10 days-worth of pixels required in an 8 hour period (1st runs, reruns, coadding)
  - ~60MB/s on average, but **very** bursty
  - 2 parallel gigabit networks should be sufficient
  - Trade off complexity of network topology and cost vs. capacity
- **Backup and permanent archiving**
  - Raw telemetry: 76GB/day archived (25GB x 3 copies, compressed)
  - Level 1 FITS frames: 100GB/day archived
  - 2 or 3 SDLTs per day will handle this

Archive/Distribution Subsystem

- **Multi-tiered approach to Archive system**
  - Raw mission telemetry archive (from INGEST)
    - 3 copies to tape. On- and off-site storage.
  - Mission metadata, engineering data in small, flat-file systems
    - Rapid access for processing control (from INGEST)
  - Image data in file system (from PIPELINES)
  - Extracted source information, image metadata, orbit metadata, QA results in DBMS integrated into IRSA infrastructure (from PIPELINES and QA)

- **IRSA Architecture**
  - Highly extensible and reusable component-based architecture:
    - Each component is a module with a standard interface that communicates with other components and fulfills one general function
    - New modules added to fulfill requirements of WISE data sets.
    - Components plugged together to make user services
  - Optimized for astronomical spatial searches and complex, general queries
  - Unrestricted by wavelength and type of mission



Development Strategy

- **Testing**, continued
  - Unit testing
    - Specific and local to each developer, before delivery
    - Confirm basic operation of coding units
  - In-situ testing
    - Code built and delivered
      - "Wild west" development deliveries
      - Formal, controlled intermediate and final deliveries
    - Confirm interfaces to and interoperability with other modules
  - RTB (regression test baseline)
    - Part of the formal delivery process
    - Enforce consistency and delivery quality
    - Increase code coverage with time