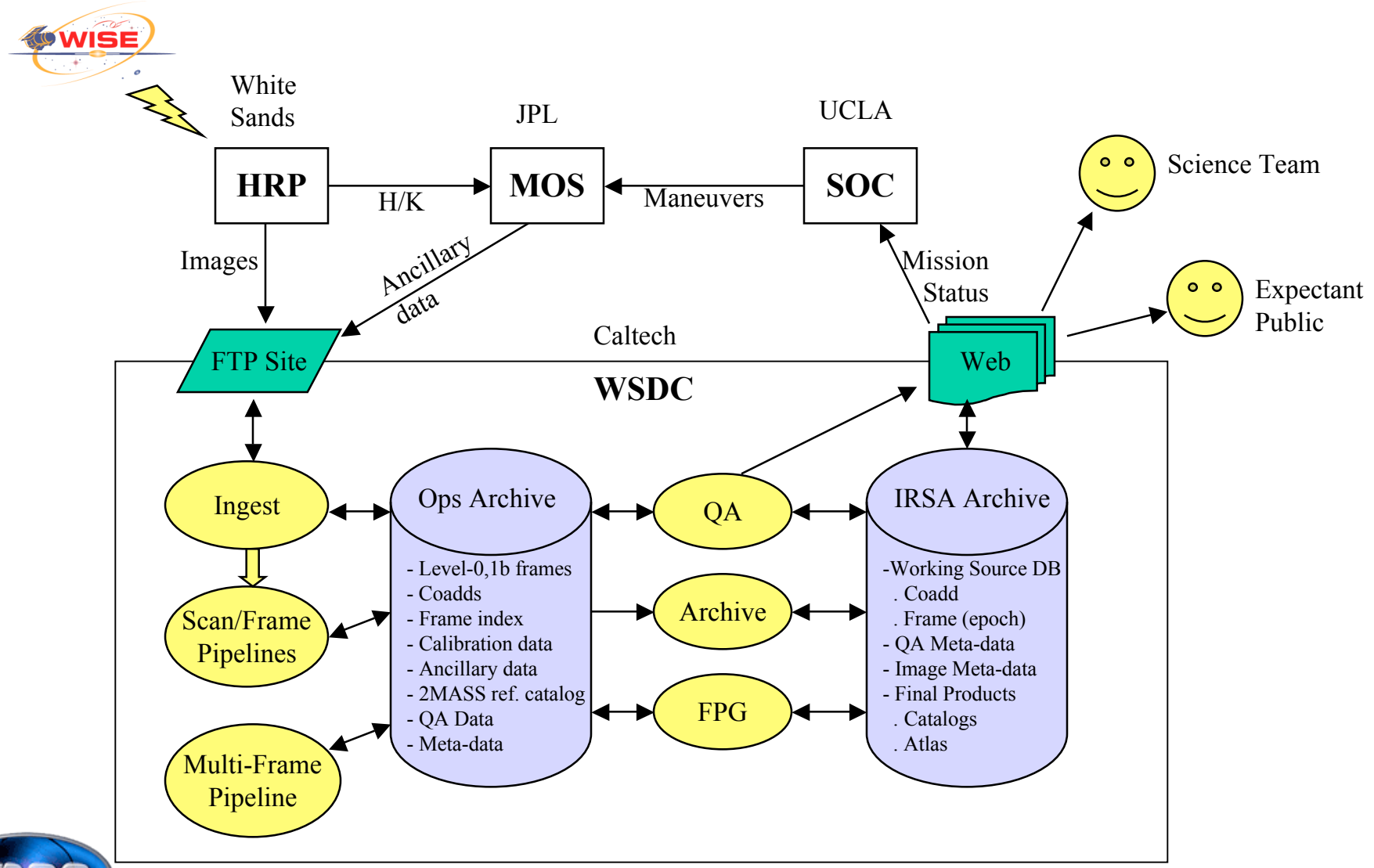
WISE Science Data System Design

Tim Conrow
IPAC

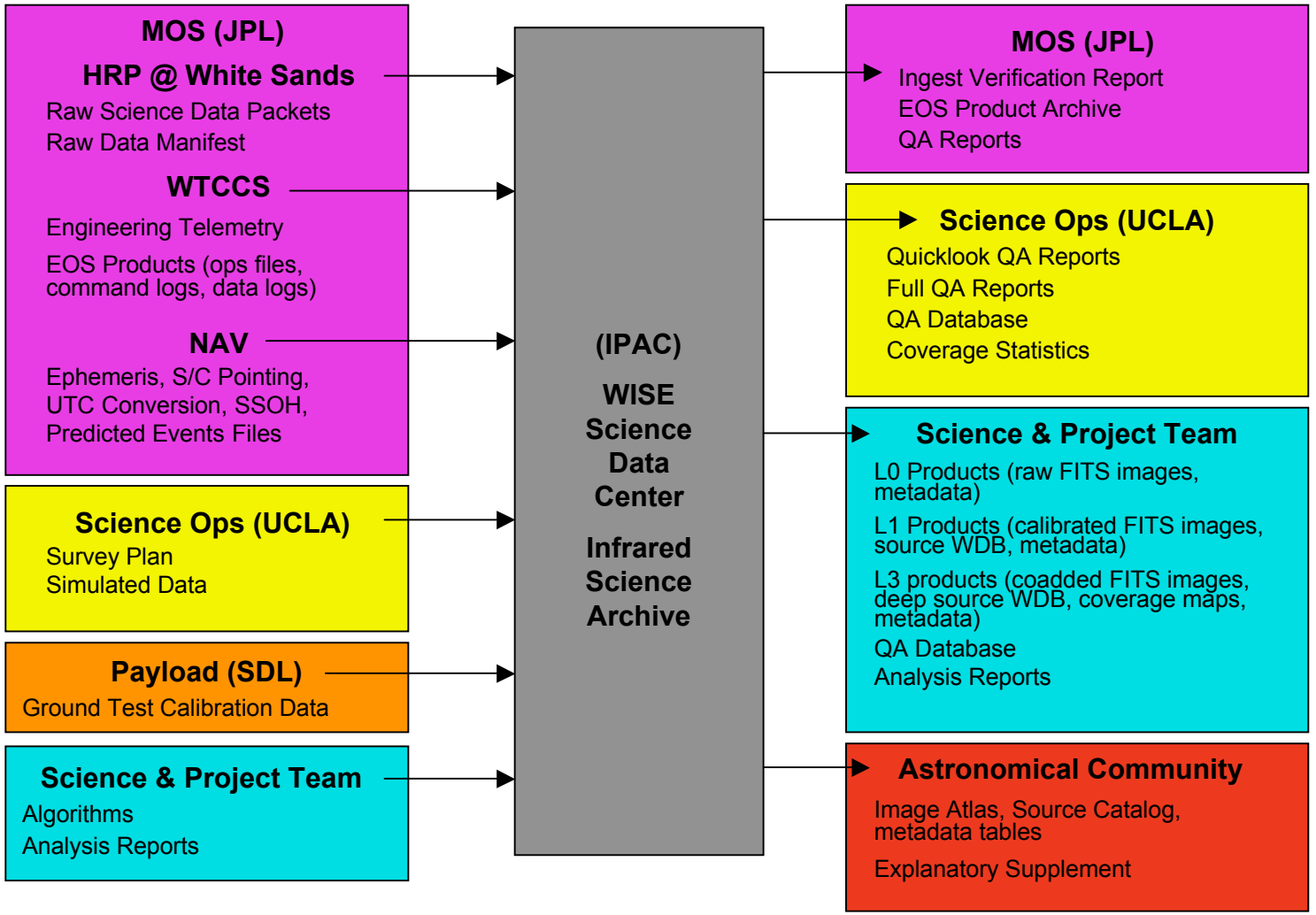
WSDC Functional Block Diagram



Jargon

- Data levels
  - Telemetry: CCSDS source packets
  - Raw: FITS, integer 2.75” pixels, band+time meta-data
  - Level-0: Raw + real pixels, add much meta data
  - Level-1: Level-0 + instrumental, astrometric/photometric calibration
    - Level-1a: Level-0 + instrumental calibration applied to pixels
    - Level-1b: Level-1a + astrometric/photometric calibration in header
  - Level-2: Level-1 + upsampled and undistorted (rubber-sheeted)
    - Level-2a: Level-1a + upsampled and undistorted
    - Level-2b: Level-1b + upsampled and undistorted
  - Level-3: Multiple frame coadds, upsampled, undistorted
  - Atlas images: Selected FPG Level-3 products
- WSDS hierarchy
  - Sub-system: One of 6 major functionality groupings making up the WSDS
  - Module: A smaller unit of functionality within a sub-system. E.g. a pipeline is made up modules, sometimes from different sub-systems
  - For historical reasons you will occasionally hear someone refer to a module as a sub-system

External Interfaces



External Interfaces

- From MOS/GDS ICD
- High Rate Processor (HRP)
  - **To WSDS:**
    - Data push using FastCopy
    - Manifest of files to be delivered
    - CCSDS source packet virtual channel (1A-1D) file for each band

      WIS_HRP_PKT_FE1A_YYYY_DDD_HH_MM_SS.bin
      WIS_HRP_PKT_FE1B_YYYY_DDD_HH_MM_SS.bin
      WIS_HRP_PKT_FE1C_YYYY_DDD_HH_MM_SS.bin
      WIS_HRP_PKT_FE1D_YYYY_DDD_HH_MM_SS.bin
    - HRP to JPL via trunked T1 lines, then to IPAC via Internet
    - 4 transfers/day totaling 25GB
    - 14 hours/day transfer time
      - HRP->JPL: 4Mbit/s
      - JPL->IPAC: timing test shows 80Mbit/s
  - **From WSDS:** Nothing

External Interfaces

- MOS
  - **To WSDC:**
    - H/K CSV: temperatures, rates, etc.

      WIS_WTCCS_TYPE_YYYY_DDD_HH_MM_SS.csv
    - ADCS S/C attitude (C-kernel), S/C time (Clock-kernel), S/C Ephemerides (SP-Kernel), read with the JPL NAIF (Navigation and Ancillary Information Facility) toolkit

      WIS_EOS_clock_file_YYYY_DDD_HH_MM_XX.txt
    - Command Mnemonic File (CMF) - maneuver command quaternions

      WIS_SEQ_CMF_YYYY_DDD_HH_MM_SS.txt
    - Predicted Events File (PEF): N/SEP crossings (orbit #'s), SAA entry/exit, Science scan start/stop

      WIS_SEQ_SEQID_YYYY_DDD_HH_MM_SS.pef

0941934127:172 2009-310T00:22:07.673 GEV,S_POLE_CROSS,REQ_2$_1GEV; << Spacecraft south pole crossing >>;
0941935556:150 2009-310T00:45:56.584 GEV,NODE_CROSS,REQ_3$_1GEV; << Spacecraft ascending node crossing >>;
0941936985:123 2009-310T01:09:45.479 GEV,N_POLE_CROSS,REQ_4$_1GEV; << Spacecraft north pole crossing >>;
0941944815:069 2009-310T03:20:15.271 GEV,SAA,REQ_6$_1GEV,2009-310T03:21:03.236; << South Atlantic Anomaly >>;
0941944815:069 2009-310T03:20:15.271 OEF: IN_SAA=TRUE;
0941944863:060 2009-310T03:21:03.236 OEF: IN_SAA=FALSE;

- Mission, H/K data for deep archive

External Interfaces

- MOS (continued)
  - **From WSDC:** Deep archive of mission+H/K data, QA reports
- SOC
  - **To WSDC:** Survey plan

WIS_EOS_SURVEY_PLAN_YYYY_DDD_HH_MM_SS.txt

```
#TOGGLE          0.2200
#MOONAVOID       1.2300
#ECLIPSEBIAS     0.0000
#BIAS            0.0000
#BIASMAX         0.0000
#DIHEDRAL        0.0000
#SCANRATE        0.0000     3.8000     0.0000
#    Time now: 10/26/2007   13:51:11
# Start time: 12/03/2009   01:30:00
#    End time: 12/10/2009   01:30:00
0.5   1310522757.860   0.027640   0.190134 -0.966404 -0.170731   0.000000   3.800000   0.000000
1.0   1310525373.402  -0.960474   0.150126  0.101595  0.211258   0.000000   3.800000   0.000000
1.5   1310528228.944  -0.099569   0.210886 -0.962331 -0.139757   0.000000   3.800000   0.000000
2.0   1310531324.486  -0.964983   0.179824 -0.025653  0.189244   0.000000   3.800000   0.000000
2.5   1310534180.028   0.027908   0.190253 -0.966589 -0.169503   0.000000   3.800000   0.000000
3.0   1310536795.570  -0.960701   0.148895  0.101355  0.211215   0.000000   3.800000   0.000000
```

  - **From WSDC:** Survey progress, Quicklook, QA Report

External Interfaces

- Science and Project Team
  - **To WSDC:** Algorithms, ground test/cal data analysis
  - **From WSDC:** Archive access, Quicklook, QA results, images
- SDL, JPL
  - **To/From WSDC:** Ground test data
- Public
  - **From WSDC:** IRSA access to
    - Atlas images
    - Catalogs and meta-data
    - Explanatory Supplement

Driving Requirements

- Key WSDC System Level Functional Requirements
  - Design of Processing Capabilities
    - Capable of supporting a 13 month mission (L4WSDC-83)
    - Ingest (L4WSDC-29-36), Pipelines (L4WSDC-37-49), Archive (L4WSDC-50-59), QA (L4WSDC-62-66)
  - Throughput and Latency
    - Data volume: 25GB/day (L4WSDC-30), 50GB/day peak (L4WSDC-31)
    - 6 months of data acquisition (L4WSDC-82)
    - Quicklook: QA report within 24 hours (L4WSDC-32)
    - Scan/Frame Pipelines: Level-1 available within 3 days (L4WSDC-39)
  - Robustness
    - Disaster recovery (L4WSDC-54)
    - 50% Processing Margin (L4WSDC-70)
  - Schedule
    - Preliminary release: EOO+6 months (L4WSDC-4)
    - Final release: EOO+17 months (L4WSDC-8)
  - Archive: Public access through IRSA (L4WSDC-51,53,60,61,86,etc.)

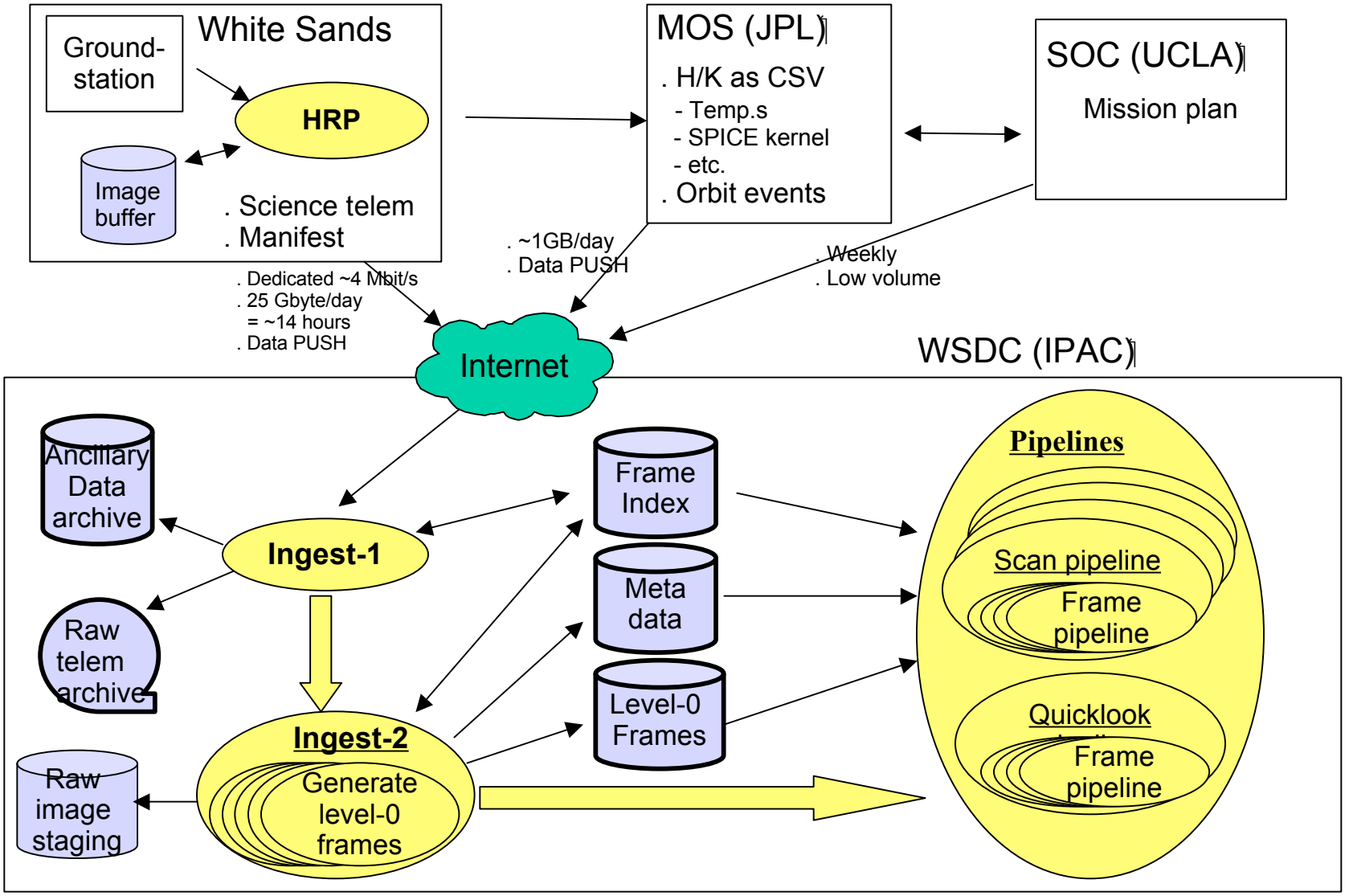
Derived Key Design Features

- A compute cluster composed of interchangeable inexpensive nodes
  - Capability, latency, robustness, schedule
- Maximal use of node-local storage to offload network
  - Latency, schedule
  - Medium cost storage and servers
  - Archive, robustness
- Minimal reliance on external services (IRSA, license servers, etc.) for ingest and pipelines
  - Latency, robustness
- Dedicated ops and QA staff
  - Latency, robustness, schedule
- Daily backup of critical data, off-site storage of critical data, disaster recovery plan
  - Schedule, latency, robustness
- Maximal leveraging of IRSA/IPAC expertise and infrastructure
  - Schedule, robustness

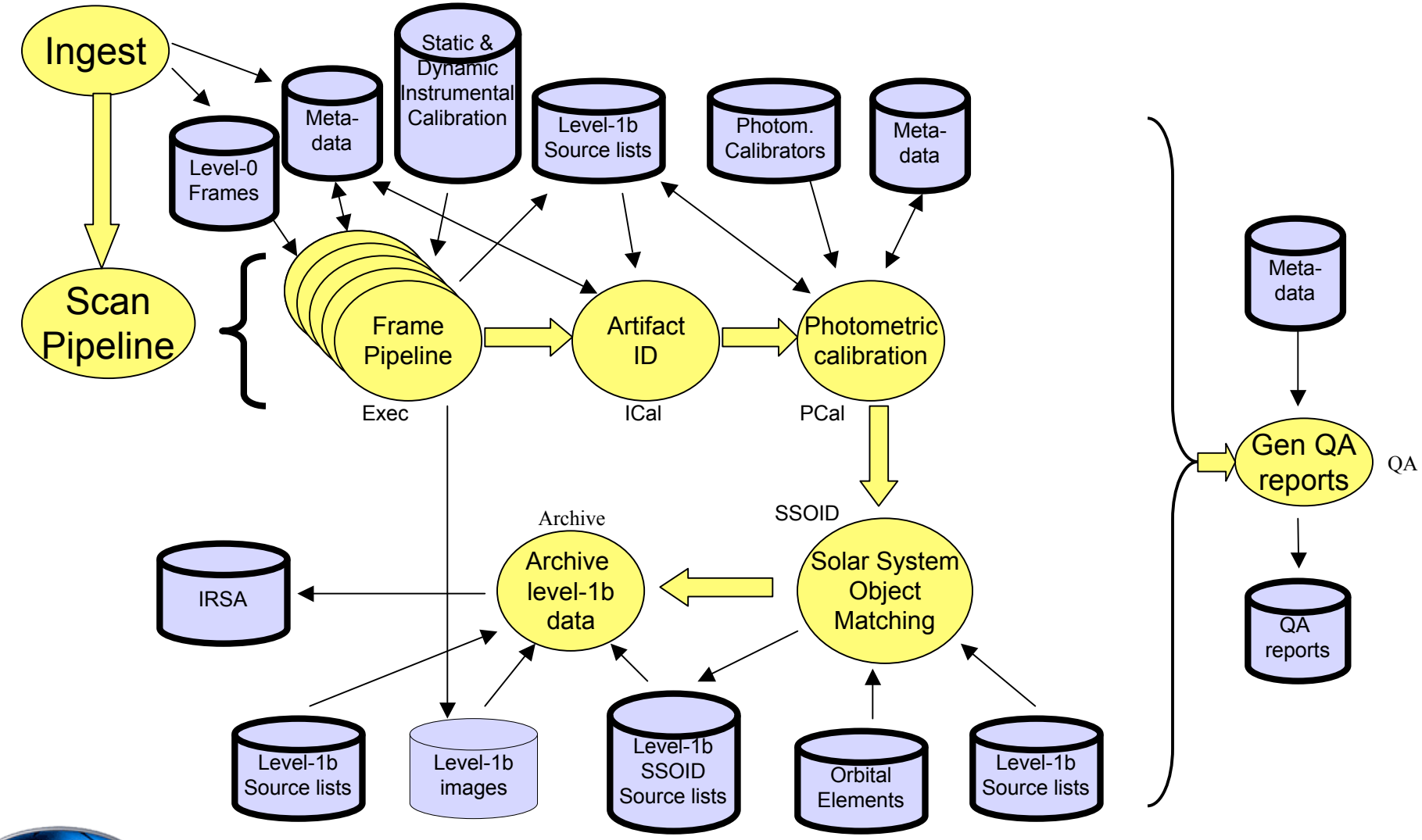
Sub-systems

- INGEST
  - Receives science data packets and engineering telemetry from MOS and assembles Level 0 FITS-format files. Stages Level 0 images and metadata for pipeline processing.
- Data Reduction PIPELINES
  - Converts Level 0 imaging data into calibrated images and extracted source *Working Databases*
  - **Frame/Scan pipeline** operates on individual frames within one "scan" (=1/2 orbit)
  - **Multi-frame pipeline** operates on data from multiple orbits
- Quality Assurance (QA)
  - Generates concise reports summarizing science data quality using summary outputs from other subsystems. Web-based report, with capability to drill-down to detailed image, graphical and tabular data
  - Reports reviewed by QA scientists at WSDC. Final quality assignment approved by PI or designee
- EXEC
  - Provides interface-related services to wrappers and pipelines
  - Mediates between external callers and applications, providing a uniform interface, binding execution units (modules) together into a unified pipeline
- ARCHIVE/Distribution System
  - Archives raw and processed mission data and metadata. Serves Image Atlas, Source Catalog and mission metadata to WISE project team and astronomical community. Integrated into Infrared Science Archive (IRSA) at IPAC.
- Final Product Generator (FPG)
  - Constructs WISE Preliminary and Final Image Atlases and Source Catalogs from *combined* image and source *Working Databases.* Includes validation, characterization and documentation.

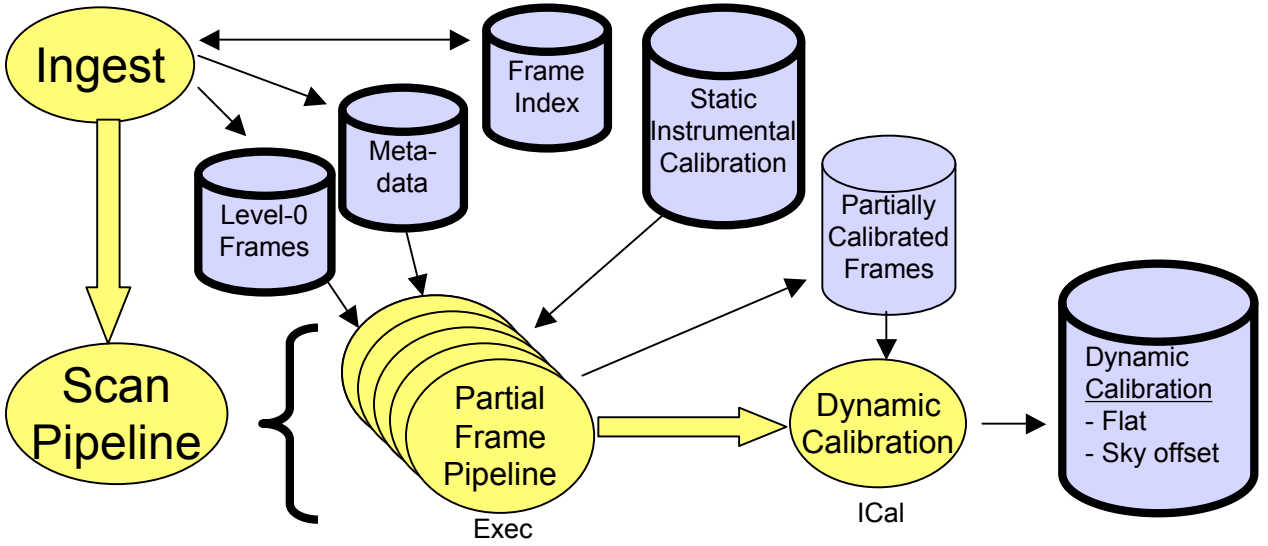
Sub-systems: Ingest



Sub-systems: Scan Pipeline

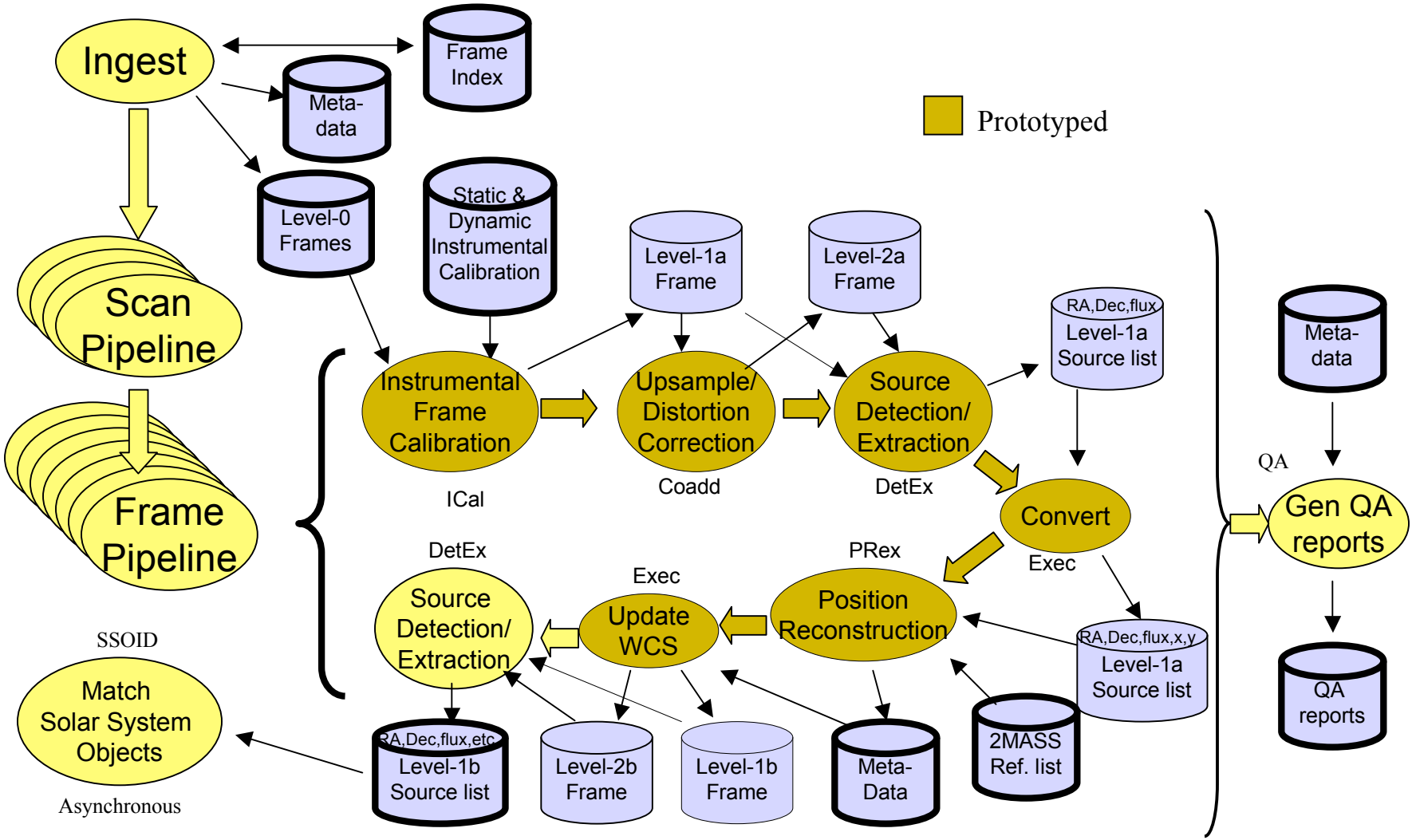


Sub-systems: Scan Pipeline, Dynamic Calibration

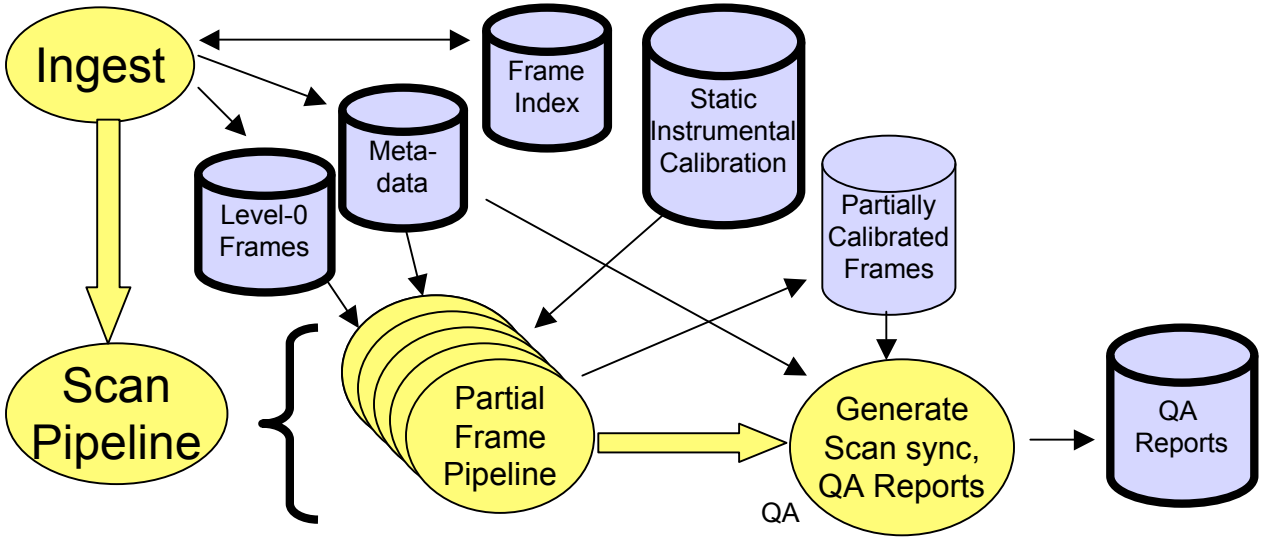


- Ingest runs dynamic calibration prior to the normal frame pipeline when …
  - … any of these conditions apply …
    - An anneal has occurred with no intervening dynamic calibration
    - Some other event which might modify the flat or sky offset
  - … and sufficient frames exposed after the last anneal (or other event) are available
- ~10 orbits of recent frames which meet certain constraints are selected
  - Out of confused areas, away from exciting background features
  - No known anomalies or saturated sources, few radiation hits
  - Etc.

Sub-systems: Frame Pipeline

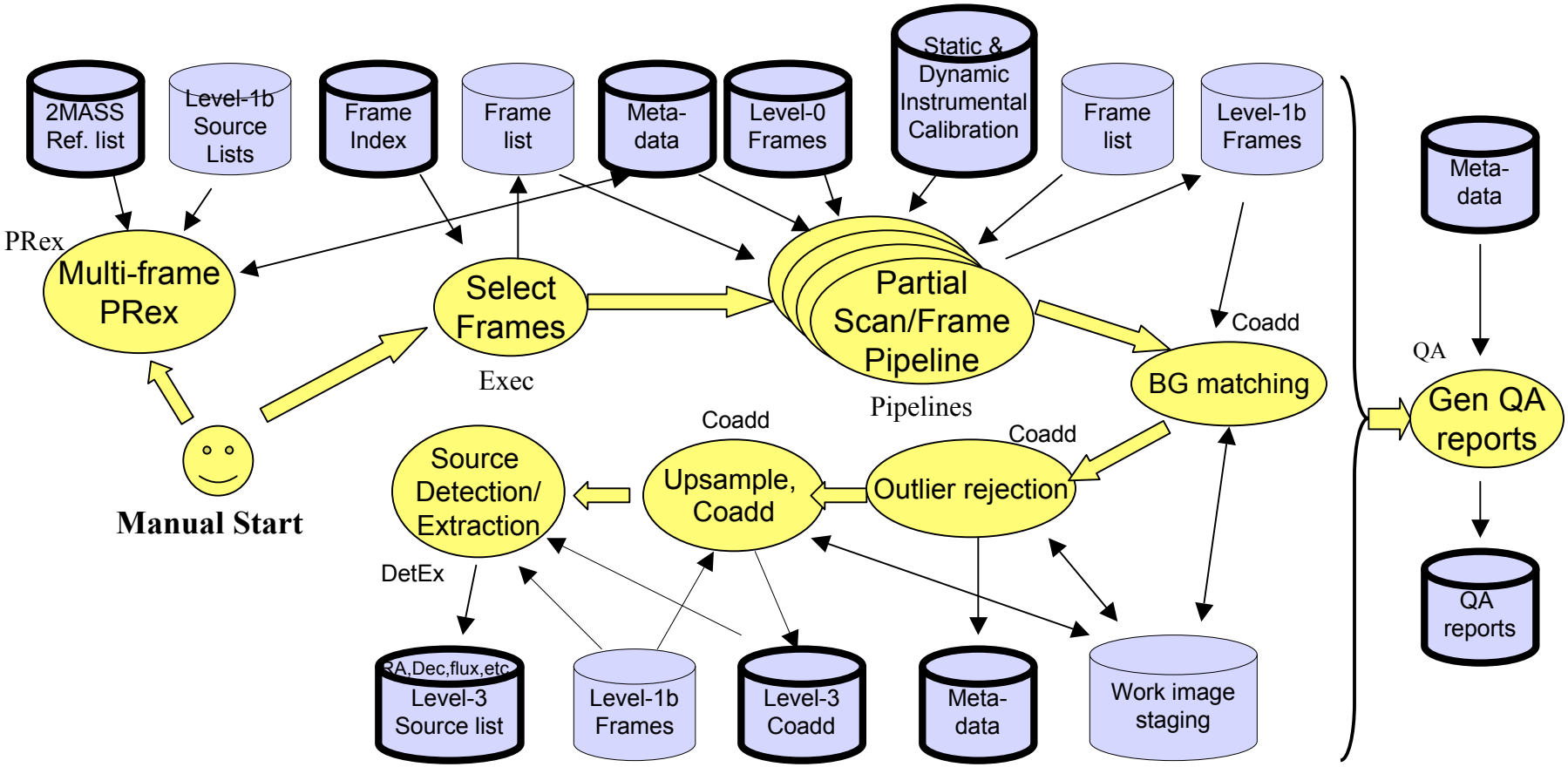


Sub-Systems: Quicklook Pipeline

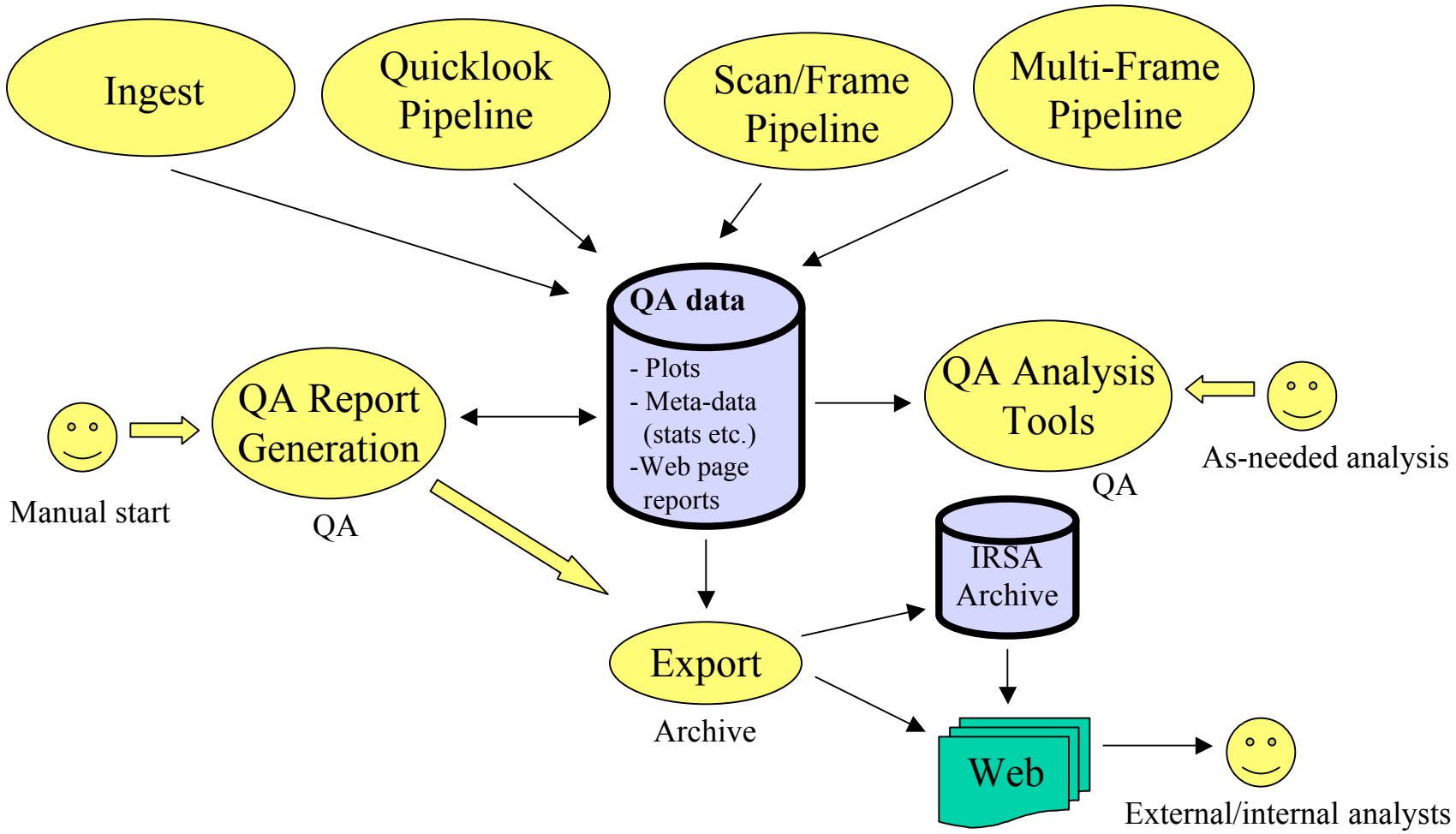


- Same as Scan/frame pipeline, except …
  - Only a select few frames processed
  - Processed for each delivery without delay, no waiting for pending frames
  - No dynamic calibration in frame processing; use static calibration only
  - Source catalogs not archived
  - Specialized QA: Scan frame sync
  - No Solar System object matching
  - Pipeline output in separate location from Scan/frame pipeline output

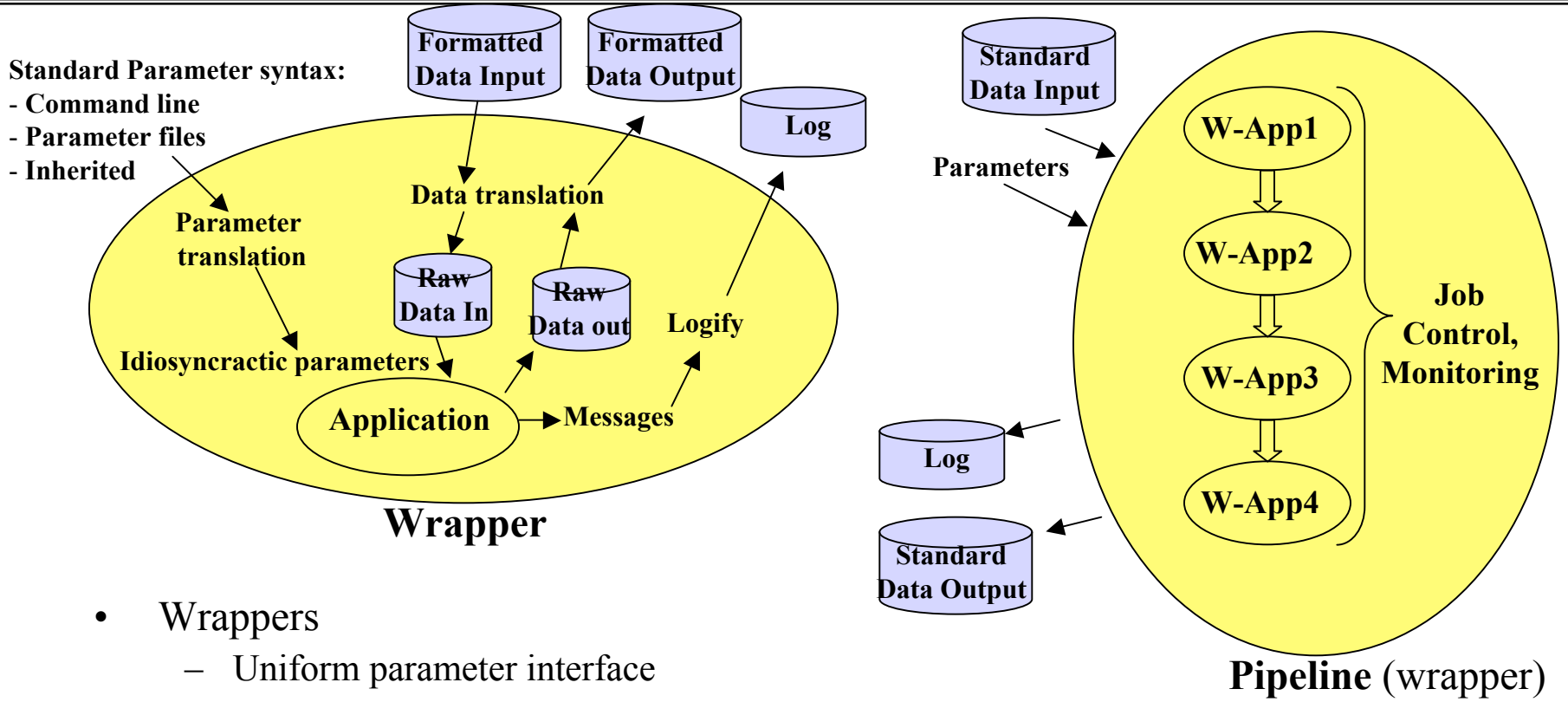
Sub-systems: Multi-frame Pipeline



Sub-systems: Quality Assurance

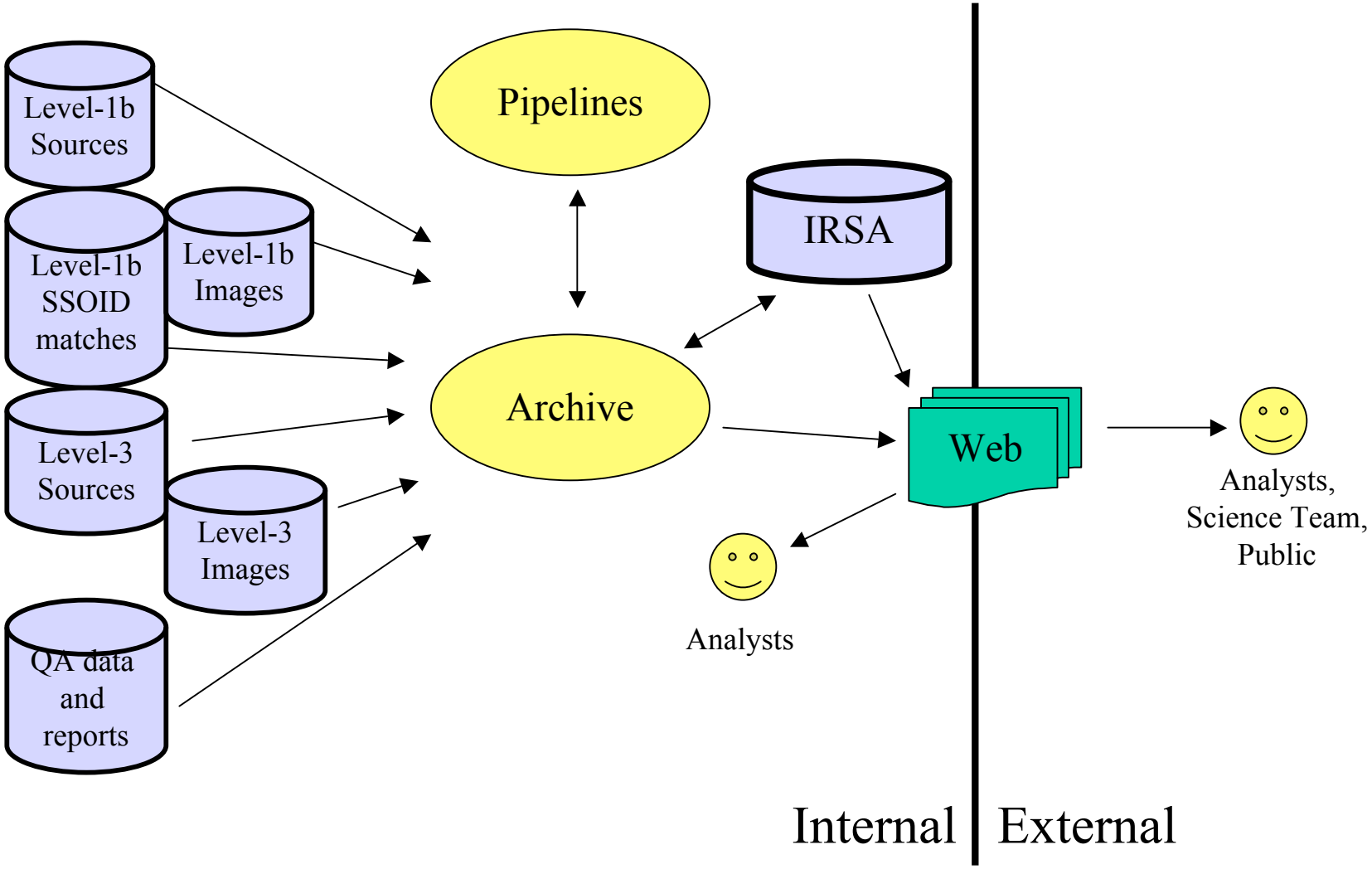


Sub-systems: Executive

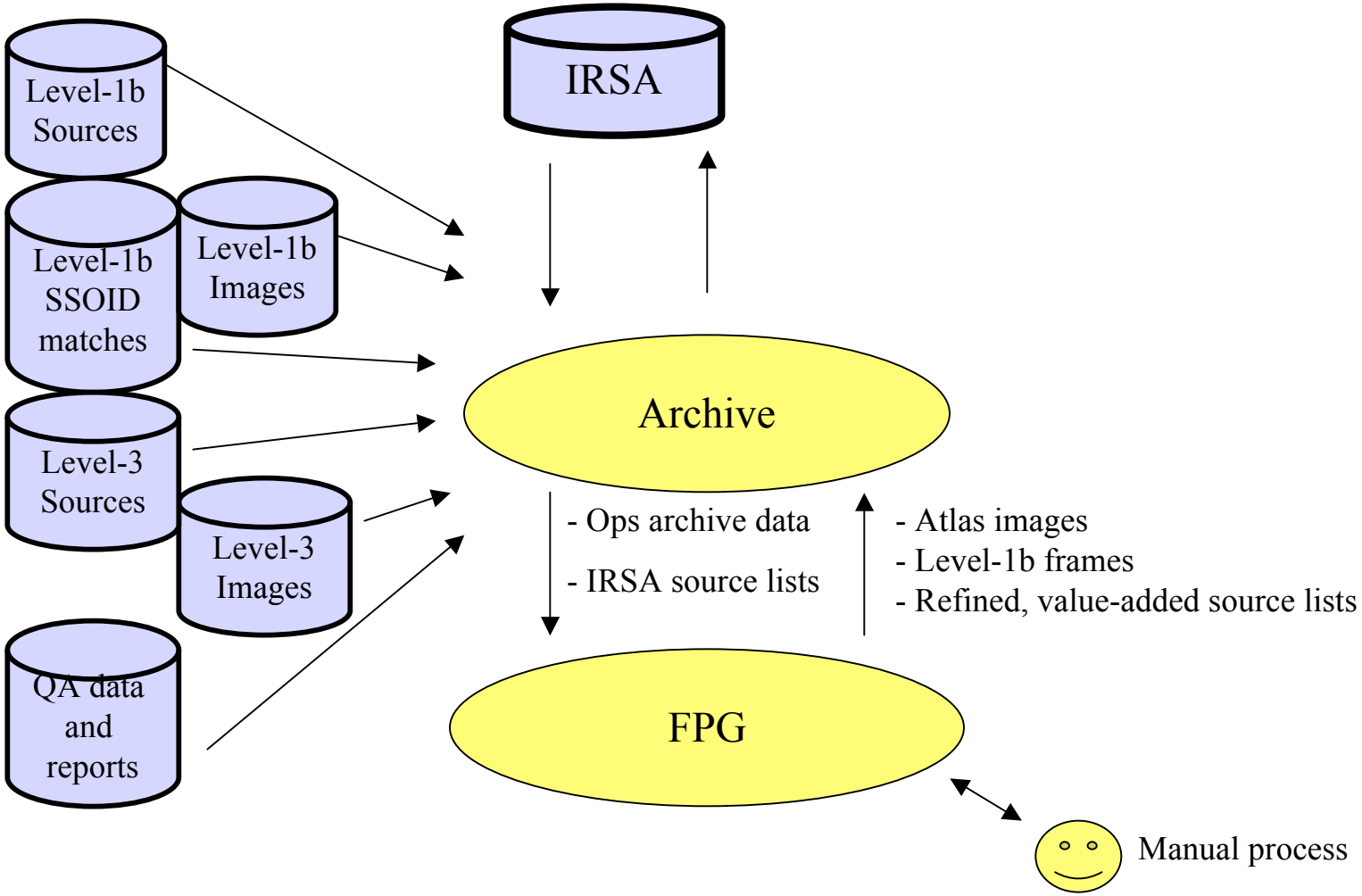


- Wrappers
  - Uniform parameter interface
  - Input/Output Data massaging
  - Stdout/Stderr message logging
  - Job Control and monitoring
    - UNIX exec/fork+IPC, or Condor
  - Pipelines are wrappers too
  - Parameter handling
    - Parameter type/constraint checking
    - Read from and/or save to parameter files
    - Inherit parameters from parent processes
  - Print context (host, release, etc.)

Sub-systems: Archive



Sub-systems: Final Product Generator



Development Status

- Exec: Parameters, stdout/err I/O logging, job control
- Ingest: Depacketize, decompress, create FITS files
- ICal: Static instrumental calibration with dummy cal data
  - Bias, flats, darks, mask, sky offset, linearity
- DetEx: Detection and simple aperture photometry and centroiding
- Prex: Pattern match, tie to 2MASS, inter-band corrections
- Coadd: Upsampling and distortion correction
- QA: Dynamic web page construction with Catalyst
- Cluster: Four nodes, Condor (job dispatch)
- Network: Planned topology in place
  - Com test measured 80Mbit/s between JPL and IPAC
- Backup: Participating in IPAC-wide backup process

Ops Archive Layout

Standard file naming scheme

Standard path structure

Local vs. global storage

Symlinks

Versioning

Ops Archive Layout

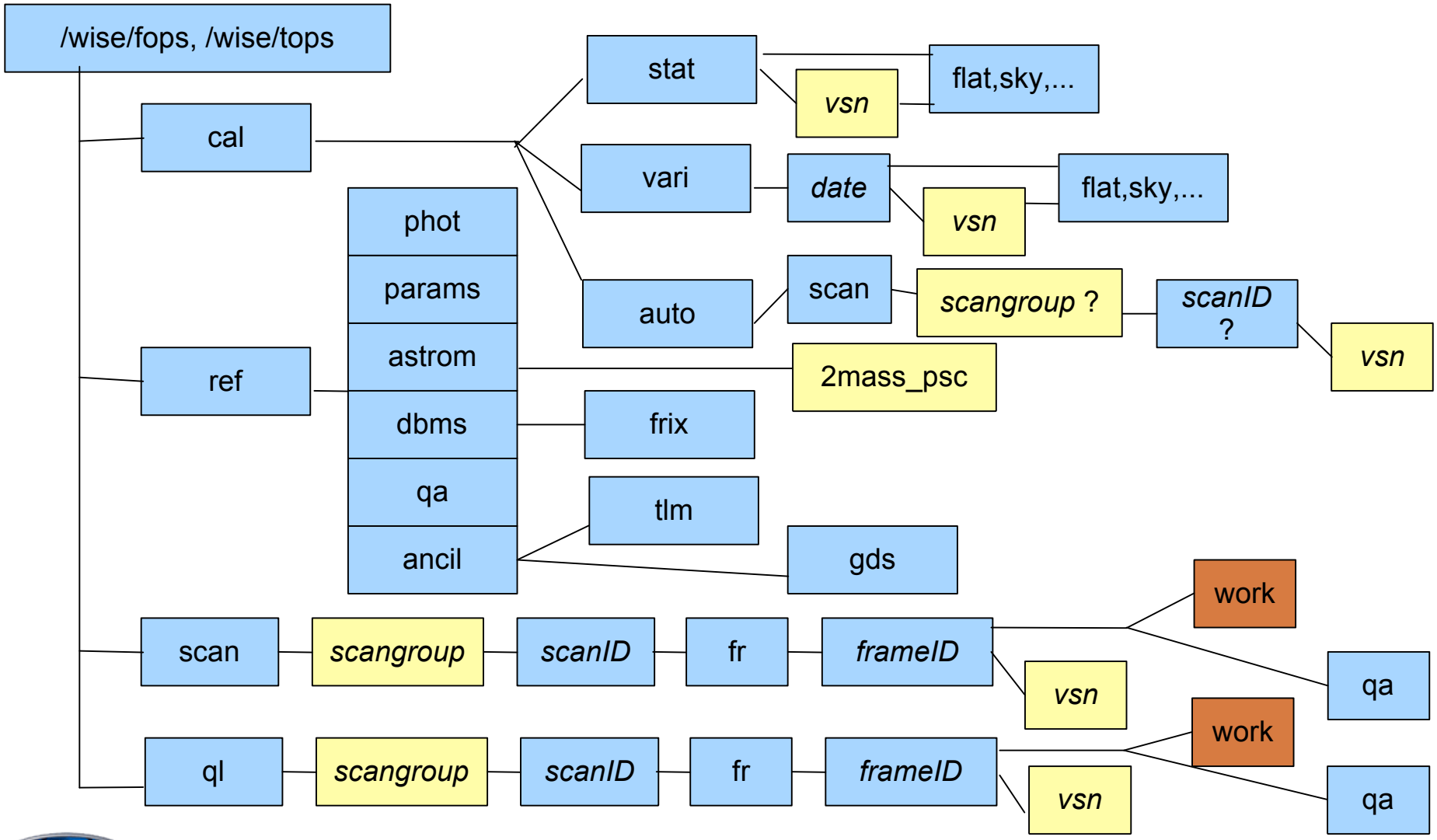
- Standard file naming scheme

  /Standard path / base name - band - type [*- variety*] *. format*

  - Standard path: see next
  - Base name
    - Frame ID: scanID+frame no., e.g. “12345a123”
      - » Scan ID equals name of half-orbit where most of the scan occurs. Provided by MOS meta-data.
      - » Frame number is computed by ingest based on scan start/stop times in MOS meta-data
    - Coadd ID: TBD
    - Calibration images: ifrcal
    - Etc.
  - Band: “w1”-”w4”
  - Type
    - “int” = intensity image
    - “unc” = uncertainty image
    - “posref” = Prex position ref. catalog
    - Etc.
  - Variety: optional sub-type specification, often used for image levels
  - Format: mostly “.tbl”, “.fits”

  – E.g. /wise/fops/scan/5a/12345a/fr/123/12345a123-w2-int-2b.fits

Ops Archive Layout



Ops Archive Layout

- Work directories are node-local but globally accessible
  - The work symlink in the Ops Archive frame pipeline output directory points to NFS-exported node-local storage. E.g. *these all represent the same location on disk*:

    Ops Archive symlink: /wise/fops/scans/1a/00101a/fr/007/work
    Global NFS Export: /compute/wcnode01/wise/fops/scans/1a/00101a/fr/007/work
    Visible only on wcnode01: /local/wise/fops/scans/1a/00101a/fr/007/work
  - Access through /local will often be faster than using NFS and will reduce LAN traffic
  - Users will access the data through the /wise/…/work symlink
- Scan group symlinks provide scaling and load balancing
  - Scan group symlinks may point to different automounted NFS-exported partitions
  - Allows for scan/frame pipeline data to be spread over up to 20 devices. E.g.

    /wise/fops/scans/**1a** → server1:/exports/wise/fops/scans/1a
    /wise/fops/scans/**1b** → server2:/exports/wise/fops/scans/1b
    /wise/fops/scans/**2a** → server3:/exports/wise/fops/scans/2a
- Frame pipeline output may be versioned
  - Data from previous runs which we may wish to save moved into a version
    /wise/fops/scans/1a/00101a/fr/007/**v1**
  - Currrent, up-to-date data always available in the parent directory, which is also given a version symlink

    /wise/fops/scans/1a/00101a/fr/007/**v2** → **.**

Data Throughput and Sizes

Sizing assumptions

Ops network, archive and backup

Network loading

Archive volume

CPU times

Data Throughput and Volume

- Sizing assumptions (rounded)
  - Level-0 frameset (including meta data): 14 MB
  - Level-1 frameset = L0 * 3: 42 MB
  - Ops coadd set = L1 * 4: 168 MB
  - Ops coadds tiling sky: 70,000
  - New ops coadds per day: 500
  - Framesets per ops coadd: 20
  - Framesets per day: 7100
  - 6 month mission framesets: 1,300,000
  - Ingest per frameset: 5 seconds
  - Scan/frame pipeline per frameset: 4 minutes
  - Partial scan/frame pipeline per frameset: 20 seconds
  - Coadd per frameset: 3 minutes
  - Processing cores: 200
  - Processing nodes: 25
  - Deliveries per day: 4
  - Compressed raw telemetry per day: 25 GB
  - Average compression ratio: 2
  - Source list table, frameset: 5 MB
  - Source list table, coadd: 50 MB

Data Throughput and Volume

- Ops Network, Archive and Backup Volume GB/Day

	Network		Archive	Backup	
	In	Out			
Ingest:	30	120	150	150	raw+level-0
Scan/Frame:	120	400	350	40	mainly level-1b images
Ops Coadd:	500	100	1	1	transient only
QA and Misc:	10	10	1	20	mainly jpegs and docs
Daily Total:	660	630	502	181	
6 Month Total:			90,000	30,000	uncompressed

- Network bandwidth
  - Assume ops go dark for 8 hours/day, so a processing day is 16 hours
    ~1,400 Gbytes/day = ~200 Mbits/sec average
  - Assume sustained peak load is 5 times the average, implies we need to support ~1 Gbit/sec.
  - Assume a network can be no more than 50% loaded to sustain a given rate, implies **we need ~2 Gbit/sec rated network capacity**
  - Achievable with careful design and cheap hardware

Data Throughput and Volume

- 6 Month Permanent Archive Volume (uncompressed)
  - Raw + level-0: **30 TB**
  - Level-1b: **60 TB**
  - Atlas: **20 TB** (2048*2048 1.375")
  - Catalogs, meta-data: **10 TB**
- Ops Processing Times
  - Ingest: 7100 framesets/day * 5 s/frameset / 4 cores
  - = **~2.5 hours/day**
  - Scan/frame: 240s/frameset * 7100 framesets/day / 200 cores
    = **~2.5 hours/day**
  - Ops Coadd: 500 coadd/day * 20 framesets/coadd * 180 s/frameset / 200 cores
    = **~2.5 hours/day**
  - 100% pad implies processing can be done in 16 hours

Software Management

Coding Standards

Issue Tracking

Revision Control

Builds and Deliveries

Software Management

- Coding standards
  - Revision number in code, binary, and output messages
  - Error and warning message format
  - Error and warning to stderr
  - Termination status
  - FITS images/tables via CFITSIO, IPAC table files
  - Languages
    - Fortran 95 (g95 v0.91+)
    - Perl (v5.10)
    - C95 plus GNU extensions (gcc v3 or 4)
    - IDL (v6.2+)
      - Prototype, analysis code and QA, prefer not to have in ops pipeline code
      - Difficult to meet coding standards
      - Requires license server
  - Coding practices
    - Commenting, structure, variable naming, etc.
    - Functionality isolation
    - Reduce code duplication

Software Management

- Issue tracking with Roundup
  - FOSS, SQLite backend, Python implementation
  - Simple web interface, customizable catagories, resolution tags, etc.
  - Track s/w and doc bugs, feature requests
  - Elevate some issues to project level

List of issues

Your Queries (edit)

Issues
Create New
Show Unassigned
Show All
Search
Show issue:

Keywords
Create New

Administration
User List

Hello, demo
My Issues
My Details
Logout

Help
Roundup docs

ID	Activity	Title	Status	Creator	Assigned To
		critical			
468	2 minutes ago	Donec consequat convallis quam.	unread	admin	epsilon
288	2 minutes ago	Vivamus tincidunt.	done-cbb	admin	demo
228	2 minutes ago	Donec consequat convallis quam.	done-cbb	admin	beta
136	2 minutes ago	Suspendisse et turpis.	testing	admin	epsilon
99	2 minutes ago	Donec consequat convallis quam.	deferred	admin	epsilon
		urgent			
477	2 minutes ago	Vestibulum gravida.	deferred	admin	admin
472	2 minutes ago	Sed convallis vehicula felis.	deferred	admin	beta
355	2 minutes ago	Fusce pede enim, nonummy sit amet, dapibus a, blandit eget, metus.	done-cbb	admin	admin
289	2 minutes ago	Aenean non felis.	testing	admin	epsilon
282	2 minutes ago	Nam egestas eros.	unread	admin	alpha
196	2 minutes ago	Integer tellus quam, mattis ac, vestibulum sed, egestas quis, mauris.	in-progress	admin	admin
181	2 minutes ago	Nam odio mauris, dignissim vitae, eleifend eu, consectetuer id, risus.	in-progress	admin	epsilon
175	2 minutes ago	Integer tellus quam, mattis ac, vestibulum sed, egestas quis, mauris.	deferred	admin	beta

Software Management

- Revision control with Subversion (SVN)
  - Widely used
  - CL and web interface
  - Code and document version tracking
  - Tag releases for easy recall of complete code state
  - Branches for bug fixes in operational code
- Builds and Deliveries with Make
  - Idiosyncratic, but widely used and well known
  - Makefile templates make user makefiles simple

    EXECLIST = sfprex

    FLIST = sfprex.f
  - Links in standard libraries, include files, etc.
  - Maintains build isolation
  - Build and install is “make install CFG=ops”

Software Management

- Build Isolation
  - Want to be able to run with multiple independent build configurations
  - Associate a configuration with a target directory

Ops	operational	/wise/base/deliv/ops
Dev	global development area	/wise/base/deliv/dev
Int	integration for next release	/wise/base/deliv/int
Tim/dev	personal development area	/wise/base/deliv/tim/dev

  - Build/link code to resolve dependencies (libraries, modules, include files) strictly within the target configuration directory hierarchy
  - Makefile templates enforce isolation
  - User configuration controlled by environment variables
  - "Newcfg" utility allows users to switch between configurations
  - Wrappers report configuration in use in output messages

Software Management

- Build version tracking: "What version am I running?"
  - Wrappers print out configuration and release tags

    E.g. "CFG=**ops**, Release=**ops-v1.1**"
  - Subversion ties a release tag to all constituent code revisions. Any tagged release can be regenerated from the repository
  - Delivered code source or binaries have SVN revision number embedded

    version => '$Id: wsfpipe 428 2008-01-08 02:19:48Z tim $'
  - For important release builds, "make" starts with repository checkout

    make install CFG=ops RELEASE=ops-v1.1

Software Management

- Formal builds
  - Procedure
    - **CCB declares release.** Candidate release code in dev.
    - **Code freeze declared.** Further changes require CCB permission.
    - **Time passes.** Informal use and testing.
    - **Integration build.** "make install RELEASE=ops-v1.0 CFG=int"
    - **RTB's run on int.**
    - **CCB approves build.**
    - **Operations halted.** Current ops build saved.
    - **Ops build.** "make install RELEASE=ops-v1.0 CFG=ops"
    - **RTB's run on ops.**
    - **CCB approves return to operations.**
  - Adherence increases until full compliance prior to release 3
  - Code changes following a release require CCB approval
  - CCB comprised of local astronomers and engineers

Software Management

- CCB
  - Composed of WSDC key astronomers and engineers
    - Tim, Roc, Davy, ???
    - Add project-level delegate for key builds/releases
  - Evaluate
    - Changes appropriate for inclusion in ops code?
    - Build ready for delivery?
    - Hardware changes properly planned and appropriate?
  - Formed prior to version 2 release
  - Strictness increases over internal releases between versions 2 and 3 until strict change control in place prior to version 3

Development Schedule

The IPAC Advantage

Code Maturation

Phase-in Strategy

Project Milestones

Capability Phase-in Schedule

Development Schedule

- The IPAC Advantage
  - Close interaction between paired cogE's and cogSci's
  - Heritage
    - Many astronomers and engineers with experience in large, automated surveys
    - Large existing code base to use as model for new development
  - IPAC System Group (ISG)
  - Infrared Science Archive (IRSA)

Development Schedule

- Code Maturation
  - Prototype
    - Almost anything goes
    - Support data flow, key interfaces and downstream functionality
  - Preliminary
    - Early production code base
    - Meets most coding standards
    - Under revision control
    - Probably not feature complete nor ready for requirements verification
  - Complete
    - Feature complete
    - Aheres to coding standards
    - Ready for requirements verification and RTB's
  - Mature
    - In use in the "complete" state for application-specific interval

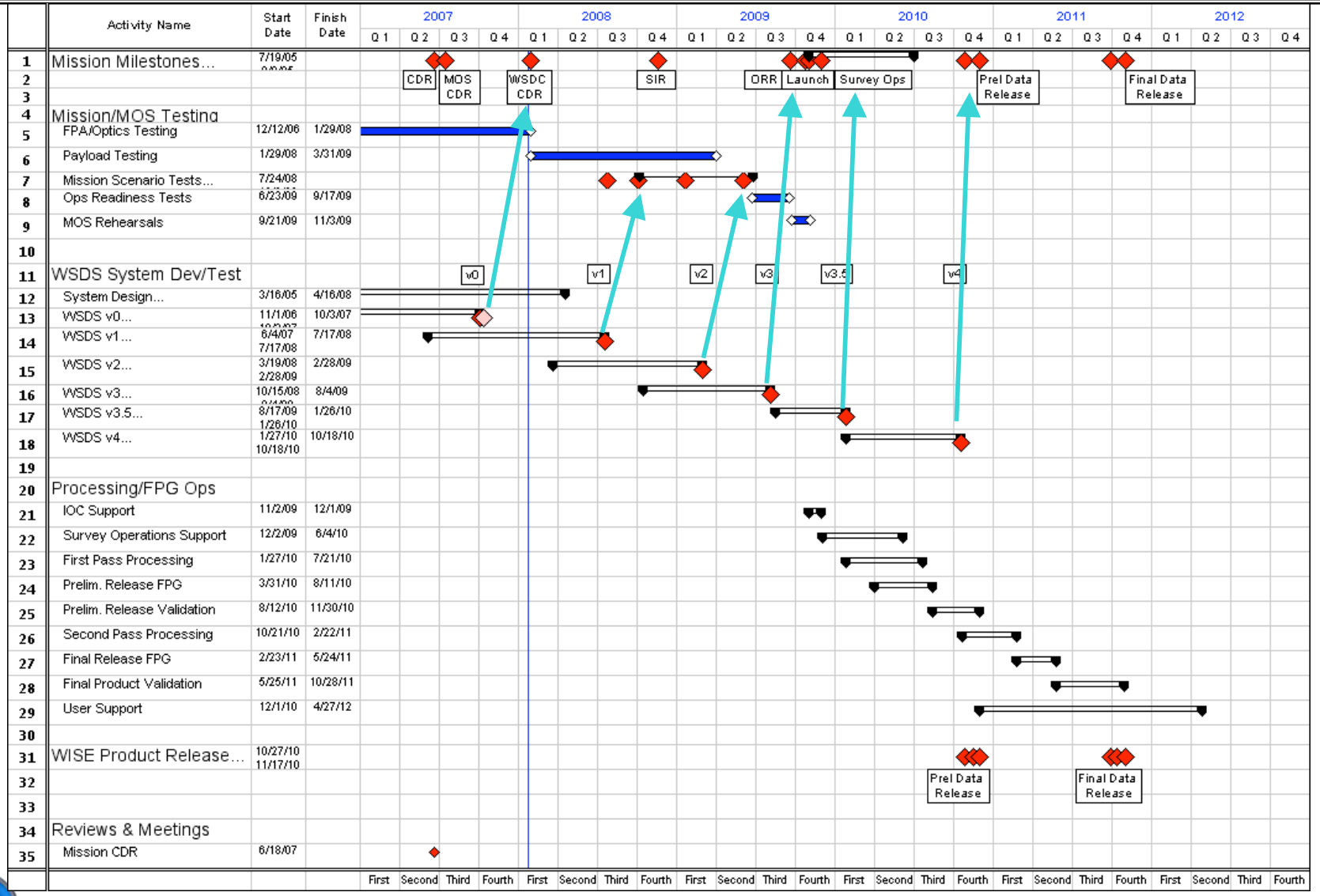
Development Schedule

- Phase-in Strategy
  - Feature set at each version matched to ...
    - Project activities, particularly instrument development and testing, and Mission Scenario testing
    - Support for future development of dependent apps
    - Estimated development time and length of maturation period
    - Staffing profile
  - 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

Development Schedule

- Project milestones most important to the WSDC
  - Simulation data deliveries: Summer ‘07 - Fall ‘09
  - WSDC CDR: Jan. 29-30 ‘08
  - Ground detector/payload characterization: Spring ‘08 - Fall ‘08
  - Mission scenario testing: Oct. ‘08 - June ‘09
  - ORR: Sept. 18 ‘09
  - Launch, IOC: Nov. 1 ‘09 - Dec. 1 ‘09
  - Survey ops: Dec. 2 ‘09 - June 1 ‘10
  - Preliminary data delivery: Dec. 1 ‘10
  - Final data delivery: Nov. 2 ‘11

Mission Schedule



Development Schedule

- Capability Phase-in Schedule
  - **Version 0 Oct. 15 '07**
    - Supports: Throughput testing, CDR
    - Capabilities
      - Frame pipeline: proto
      - Ingest: proto
      - Cluster: proto
      - Exec: prelim
      - SFPRex: prelim
      - ICal: proto
      - Coadd: proto
      - DetEx: proto

Development Schedule

- Capability Phase-in Schedule (continued)
  - **Version 1** **July 17, '08**
    - Supports: MST 2 - End-to-end testing
    - Capabilities:
      - Cluster: prelim
      - Storage: proto
      - QA: proto
      - Multi-frame pipeline: proto
      - Archive: proto
      - Ingest: prelim
      - ICal: prelim
      - Pcal: proto
      - Coadd: prelim
      - DetEx: prelim
      - Frame pipeline: complete
      - Scan pipeline: proto

Development Schedule

- Capability Phase-in Schedule (continued)
  - **Version 2 Feb. 28, '09**
    - Supports MST 6, 7 - Load Volume Tests
    - Capabilities:
      - QA: prelim
      - Archive: prelim
      - SSOID: prelim
      - Exec: complete
      - Multi-frame pipeline: complete
      - Ingest: complete
      - ICal: complete
      - PCal: prelim
      - Coadd: complete
      - DetEx: complete
      - Frame pipeline: mature

Development Schedule

- Capability Phase-in Schedule
  - **Version 3** **Aug. 10 '09**
    - Supports ORR, Launch, IOC
    - Capabilities:
      - QA: complete
      - SSOID: complete
      - PCal: complete
      - Cluster: mature
      - Storage: mature
      - Archive: mature
      - Exec: mature
      - Multi-frame pipeline: mature
      - Ingest: mature
      - ICal: mature
      - Coadd: mature
      - DetEx: mature
      - FPG: prelim

Development Schedule

- Capability Phase-in Schedule
  - **Version 3.5 Jan. 26, '10**
    - Support: Response to IOC, survey operations, preliminary data release
    - Capabilities:
      - QA: mature
      - SSOID: mature
      - PCal: mature
      - FPG: complete
  - **Version 4 Oct. 18, '10**
    - Support: Final product release
    - Capabilities:
      - FPG: mature

Design Issues and Concerns

- Hardware scaling
- Coadd run times vs. duty-cycle
- On-orbit detector behavior
- Loss of bands during ops
- Parallel runs required during ops to prepare for prelimary release